University of Coimbra
Faculty of Sciences and Technology
Department of Electrical and Computer Engineering

Jérôme Amaro Pires Mendes

Computational Intelligence Methodologies for Control of Industrial Processes

Tese de Doutoramento em Engenharia Electrotécnica e de Computadores, ramo de especialização em Automação e Robótica, orientada pelo Prof. Dr. Rui Alexandre de Matos Araújo e apresentada ao Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

February 2014

Universidade de Coimbra

University of Coimbra
Faculty of Sciences and Technology
Department of Electrical and Computer Engineering

Computational Intelligence Methodologies for Control of Industrial Processes

by

Jérôme Amaro Pires Mendes

Tese de Doutoramento em Engenharia Electrotécnica e de Computadores, ramo de especialização em Automação e Robótica, orientada pelo Prof. Dr. Rui Alexandre de Matos Araújo e apresentada ao Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Coimbra
February 2014

Agradecimentos

Gostaria de agradecer às várias pessoas que, de diferentes formas, contribuíram para a realização desta tese. O meu reconhecimento ao Professor Dr. Rui Araújo, pela sua orientação e conhecimentos transmitidos ao longo desta tese. O meu profundo agradecimento ao Eng. Pedro Sousa, Eng. Luís Alves da AControl e ao Eng. Filipe Apóstolo, pelo apoio, ajuda, amizade e confiança que depositaram em mim; e por permitirem e apoiarem a realização desta tese. Agradeço ao Instituto de Sistemas e Robótica (ISR-UC) e à empresa Acontrol por apoiarem a realização desta tese.

Agradeço à Fundação para a Ciência e a Tecnologia (FCT) que suportou esta tese através da bolsa de doutoramento com a referência SFRH/BD/63383/2009. Este trabalho foi também suportado pelo Projecto SInCACI "Sistemas Inteligentes de Controlo, Aquisição e Comunicação Industrial" (referência: SInCACI/3120/2009), e pelo Projecto SCIAD "Sistemas de Controlo Industrial Auto-Aprendizes através de Dados do Processo" (referência: SCIAD/2011/21531), ambos co-financiados pelo QREN, no âmbito do "Mais Centro - Programa Operacional Regional do Centro", e pela União Europeia através do Fundo Europeu de Desenvolvimento Regional (FEDER), bem como pela Agência de Inovação (AdI) no caso do projecto SInCACI. Agradeço a estes projectos.

Aos meus pais que permitiram que tirasse um curso superior. Um especial agradecimento à minha mãe e ao meu irmão que sempre se mostraram disponíveis para me ajudar e me deram força nos momentos de fraqueza. Aos meus falecidos avós José Amaro e Idalina Sousa pela amizade, carinho e ajuda, que sem o qual não seria possível tirar um curso superior. Ao meu grande amigo e colega de curso Eng. João Salgueiro, que desde infância mutuamente nos ajudamos e percorremos o mesmo caminho até finalizarmos o mestrado, e que mesmo durante o meu doutoramento me apoiou e me orientou com a sua experiência e sabedoria.

Aos meus amigos pelos bons momentos passados durante o meu percurso

académico. Aos meus colegas e amigos do ISR, pela união e amizade, em especial ao Eng. Ricardo Maia, Eng. Francisco Alexandre, Eng. João Loureiro, Eng. Luís Maricato, Eng. Sérgio Sousa, Eng. Teresa Sousa e Dr. Omar Mahjoubi. Gostaria de salientar o apoio prestado pelo Eng. Ricardo Maia e Eng. Francisco Alexandre durante todas as fases desta tese, e ainda ao Eng. Ricardo Seco e Eng. Tiago Matias que também foram elementos importantes para parte do trabalho elaborado nesta tese.

Um obrigado especial, à minha mulher Teresa Mendes pelo seu apoio, amor, carinho e compreensão da minha ausência causada pelo tempo dedicado a este percurso académico, e pela força que me deu nos momentos de fraqueza.

Abstract

Industrial processes have faced major changes in the market during the past decades, due to the increasing world competition, and the environmental legislation, which resulted in hard constraints that increase the process complexity and the costs of production equipment. Many industrial systems exhibit nonlinear behaviors and frequently have many complex characteristics, such as unknown and time-varying dynamics, constraints, and disturbances. In the future, worldwide industrial competition in general, and specifically in industrial plants, will require high levels of efficiency, flexibility, reliability and a control performance that can cover a wide process operation range and process variations. Thus, more advanced control systems will therefore be required to overcome these changes and features of the industrial processes.

This thesis addresses identification and control problems on nonlinear industrial processes using Fuzzy Logic theory. The control of Nonlinear processes, which have widespread presence in industry applications, as well as modeling and control difficulties, non-modeled dynamics, time-varying parameters, and presence of disturbances, are important problems that will be addressed. Three main research objectives and research directions are considered.

The first objective is to design automatic methodologies to identify a model of nonlinear process through a numerical data set and using fuzzy logic and genetic algorithms. The learning of a Takagi-Sugeno (T-S) fuzzy model is performed from input/output data to approximate unknown nonlinear processes by a coevolutionary genetic algorithm (GA). The proposed method is an automatic tool since it does not require any prior knowledge concerning the structure (e.g. the number of rules) and the database (e.g. antecedent fuzzy sets) of the T-S fuzzy model, and concerning the selection of the adequate input variables and their respective time delays for the prediction setting. The GA approach is composed by five hierarchical levels and

has the global goal of maximizing the prediction accuracy. The first level consists of the selection of the set of input variables and respective delays for the T-S fuzzy model. The second level considers the encoding of the membership functions. The individual rules are defined at the third level, the population of the set of rules is treated in fourth level, and a population of fuzzy systems is handled at the fifth level.

The second objective of the thesis is to design automatic methodologies to control a nonlinear process, by learning/designing all fuzzy parameters of a Fuzzy Logic Controller (FLC) from data extracted from a given process while it is being manually controlled. The learning of the FLC is performed by a hierarchical genetic algorithm (HGA) composed by a five level structure. The selection of an adequate set of input variables, and the definition of the antecedent and consequent membership functions, individual rules, set of rules, and fuzzy operators (t-norm, implication, aggregation, and defuzzifier operators) which constitute the FLC, are all performed.

Finally, the third objective of the thesis is the design of methodologies for fuzzy model predictive control of nonlinear time-varying systems without knowledge about the mathematical model of the plant. The fuzzy systems learned by the methodologies proposed in the work related to the first objective were incorporated into the control methodology, namely on the Generalized Predictive Control (GPC) algorithm.

To validate and demonstrate the performance and effectiveness of the proposed methodologies, they are applied on the identification of a model for the estimation of the fluoride concentration in the effluent of a real-world wastewater treatment system; and on the control of a simulated continuous stirred tank reactor (CSTR), on the control of the dissolved oxygen in an activated sludge reactor within a simulated wastewater treatment plant, and on the control of a real-world experimental setup composed of two coupled DC motors.

Resumo

Os processos industriais, durante as últimas décadas, têm enfrentado grandes mudanças no mercado devido ao aumento da concorrência mundial e da legislação ambiental, o que resultou em constrangimentos rígidos que aumentam a complexidade do processo e os custos dos equipamentos de produção. Muitos sistemas industriais exibem comportamentos não-lineares e, frequentemente, possuem muitas características complexas, tais como dinâmicas desconhecidas e variantes no tempo, restrições e perturbações. No futuro, a concorrência industrial a nível mundial, e mais especificamente nas plantas industriais, irá exigir altos níveis de eficiência, flexibilidade, e confiança bem como um bom desempenho de controlo que abranja uma ampla gama de regiões de operação, e de variações no processo. Assim, devido a estas mudanças e características dos processos industriais, serão necessários sistemas de controlo mais avançados.

Esta tese aborda problemas de identificação e controlo em processos industriais não-lineares utilizando lógica difusa. O controlo de processos não-lineares, que têm uma ampla presença em aplicações industriais, bem como dificuldades de identificação e controlo, dinâmicas não-modeladas, parâmetros variantes no tempo e presença de perturbações são problemas importantes que serão abordados. São abordados três principais objectivos de investigação e direcções de pesquisa.

O primeiro objectivo é o desenvolvimento de metodologias para identificação de um modelo de um dado processo não-linear de forma automática, através de um conjunto de dados e usando lógica difusa e algoritmos genéticos (AG). A aprendizagem de um modelo difuso de Takagi-Sugeno (T-S) é realizada a partir de um conjunto de dados de entrada/saída para aproximar processos não-lineares desconhecidos por um algoritmo genético co-evolutivo. O método proposto é uma ferramenta automática, uma vez que não exige qualquer conhecimento prévio sobre a estrutura (por exemplo, o número de regras) e base de dados (por exemplo, conjuntos difusos das

antecedentes) do modelo difuso T-S, e acerca da selecção de variáveis de entrada adequadas e respectivos atrasos para a definição da predição. A abordagem dos AG é composta por cinco níveis hierárquicos e tem o objectivo global de maximizar a precisão da predição. O primeiro nível consiste na seleção de um conjunto de variáveis de entrada e respectivos atrasos para o modelo difuso T-S. O segundo nível considera a codificação das funções de pertença. As regras individuais são definidas no terceiro nível, a população do conjunto das regras é tratada no quarto nível e a população dos sistemas difusos é manipulada no quinto nível.

O segundo objectivo da tese é o desenvolvimento de metodologias automáticas para o controlo de um dado processo não-linear, através da aprendizagem de todos os parâmetros difusos de um controlador difuso (CD) a partir de dados extraídos de um determinado processo sob controlo manual. A aprendizagem do CD é realizada por um algoritmo genético hierárquico, composto por uma estrutura de cinco níveis. A seleção de um conjunto adequado de variáveis de entrada, da definição das funções de pertença das antecedentes e da consequente, das regras individuais, do conjunto de regras e dos operadores difusos (operadores de t-norma, implicação, agregação e desdifusão) que constituem o CD são realizadas.

Finalmente, o terceiro objectivo da tese é o desenvolvimento de metodologias para controlo predictivo com modelo difuso para sistemas não-lineares variantes no tempo sem o conhecimento sobre o modelo matemático da planta. Os sistemas difusos aprendidos pelas metodologias propostas no trabalho relacionado com o primeiro objectivo foram incorporados nesta metodologia de controlo, nomeadamente no controlador "Generalized Predictive Control" (GPC).

Para validar e demonstrar o desempenho e eficácia dos métodos propostos, estes são aplicados na identificação de um modelo para a estimação da concentração de "fluoreto" no efluente de um sistema real de tratamento de águas residuais, e no controlo de um reactor contínuo do tipo tanque agitado simulado, no controlo do oxigénio dissolvido num reactor de lamas activadas dentro de uma estação simulada de tratamento de águas residuais, e no controlo de um sistema experimental real composto por dois motores DC acoplados.

Symbols and Abbreviations

General Abbreviations

ACO	Ant Colony Optimization
AFC	Adaptive Fuzzy Control
AFGPC	Adaptive Fuzzy Generalized Predictive Control
AHGA-Control	Adaptive Hierarchical Genetic Algorithm for Control
AHGA-FCM	Adaptive Hierarchical Genetic Algorithm with Fuzzy c-Means
ANFIS	Adaptive Neuro-Fuzzy Inference System
ARMAX	Autoregressive Moving Average eXogenous
ARX	AutoRegresive model with eXternal input
BSM1	Benchmark Simulation Model n.1
COST	European Cooperation in Science and Technology
CSTR	Continuous Stirred Tank Reactor
DC	Direct Current
DDSS	Data-Driven Soft Sensors
DO	Dissolved Oxygen
ELM	Extreme Learning Machine
FCM	Fuzzy c-Means
FCRM	Fuzzy c-Regression Model
FCS	Fuzzy Control Systems
FGA	Fuzzy Genetic Algorithm
FLC	Fuzzy Logic Control
FLC-BSM1	Fuzzy Logic Controller implemented on the Benchmark Simulation Model n.1
FLS	Fuzzy Logic System

FMMGPC	Fuzzy Multiple Models Generalized Predictive Control
FNN	Fuzzy Neural Network
GA	Genetic Algorithm
GPC	Generalized Predictive Control
HGA	Hierarchical Genetic Algorithm
HGA-Control	Hierarchical Genetic Algorithm for Control
HGA-FCRM	Hierarchical Genetic Algorithm with Fuzzy c-Regression Model
HVAC	Heating, Ventilation and Air Conditioning
I/O	Input/Output
ILLSA	Incremental Local Learning Soft Sensing Algorithm
INICONTROL	Initialization algorithm for Hierarchical Genetic Algorithm for Control
INICONTROL-FCM	Initialization algorithm for Hierarchical Genetic Algorithm for Control with Fuzzy c-Means
LQR	Linear Quadratic Regulator
MIMO	Multiple-Input Multiple-Output
MLP	MultiLayer Perceptron
MPC	Model Predictive Control
MSE	Mean Square Error
NARMAX	Nonlinear Autoregressive Moving Average eXogenous
NARX	Nonlinear AutoRegresive model with eXternal input
NMPC	Nonlinear Model Predictive Control
NN	Neural Network
NTU	Nephelometric Turbidity Unit
PC	Personal Computer
PID	Proportional-Integral-Derivative
ppm	Parts-Per-Million
PSO	Particle Swarm Optimization
RAM	Random Access Memory
RLS	Recurvive Least Squares
RLS-ADF	Recurvive Least Squares with Adaptive Directional Forgetting
RFNN	Recurrent Fuzzy Neural Network

RPLS	Recursive Partial Least Squares
SISO	Single-Input Single-Output
T-S	Takagi-Sugeno
TCU	True Colour Unit
WWTP	Wastewater Treatment Plant

General Symbols

$\boldsymbol{\theta}_i$	Parameter vector with adjustable parameters of the i-th fuzzy rule.
$\boldsymbol{\Theta}$	Matrix that contain all adjustable parameter vector $\boldsymbol{\theta}_i$.
$\boldsymbol{\Theta}^*$	Optimal parameters of the matrix $\boldsymbol{\Theta}$.
$\dot{\boldsymbol{\Theta}}$	Derivative of the matrix $\boldsymbol{\Theta}$.
$\mathbf{C}_i$	Covariance matrix of the i-th fuzzy rule.
$\mathbf{e}$	Closed loop error.
$\mathbf{I}$	Identity matrix.
$\mathbf{Q}$	Arbitrary positive-definite matrix.
$\mathbf{P}$	Positive-definite matrix.
$\mathbf{p}$	Last column of matrix $\mathbf{P}$.
$\mathbf{t}_5$	A t-norm method.
$\mathbf{U}$	Fuzzy partition matrix.
$\mathbf{V}$	Matrix of cluster centroid vectors.
$\mathbf{v}_i$	Cluster centroid vectors.
$\mathbf{X}$	Data matrix.
$\mathbf{x}$	Input variable vector.
Δ	Difference operator.
η	Overlapping factor or the fuzziness parameter.
γ	Gain of the adaptation law.
$\boldsymbol{\Lambda}$	Matrix to attain the desired error dynamics.
$\lambda(z^{-1})$	Weighting polynomial.
μ_i	Fuzzy partition of fuzzy subsets i.
$\mu_{A_j^i}$	Antecedent fuzzy membership functions of the i-th fuzzy rule of the variable j.
μ_{B_i}	Consequent fuzzy membership functions of the i-th fuzzy rule.

ν_i	Parameter of the i-th fuzzy rule of RLS-ADF.
ρ	Positive constant parameter of RLS-ADF.
σ_{ij}	Gaussian dispersion of membership functions i of the variable j.
τ_i	Parameter of the i-th fuzzy rule of RLS-ADF.
θ_{ij}	Adjustable parameter of the i-th fuzzy rule and of the input variable j.
φ_i	Forgetting factor of the fuzzy rule i of RLS-ADF.
A_j^i	Linguistic term characterized by fuzzy membership functions $\mu_{A_j^i}(x_j)$.
B_i	Linguistic term characterized by fuzzy membership functions $\mu_{B_i}(u)$.
$b_{1,w}(v_h)$	Parameter of the membership function w of the variable v_h.
$b_{2,w}(v_h)$	Parameter of the membership function w of the variable v_h.
b_i	Center of membership function B_i.
$C_{1,w}(v_h)$	Parameter of the membership function w of the variable v_h.
$C_{2,w}(v_h)$	Parameter of the membership function w of the variable v_h.
$d_i(l)$	Euclidean distance (l^2-norm).
$e(k+p)$	An p-step ahead prediction of tracking error.
i	i-th fuzzy rule.
i_{max}	Maximum number of chromosomes at Level 5.
j	j-th input variable.
J_1^m	Fitness function of individual m at Level 1.
J_2^t	Fitness function of individual t at Level 2.
J_3^k	Fitness function of individual k at Level 3.
J_4^j	Fitness function of individual j at Level 4.
J_5^i	Fitness function of individual i at Level 5.
j_{max}	Maximum number of chromosomes at Levels 4.
K_j	Number of membership functions of the input variable j.
k_{max}	Maximum number of chromosomes at Level 3.
L	Number of observations.
l	An observation from data.
$L_w(v_h)$	Parameter of the membership function w of the variable v_h.
$m_{1,w}(v_h)$	Parameter of the membership function w of the variable v_h.

$m_{2,w}(v_h)$	Parameter of the membership function w of the variable v_h.
$m_{2,0}(v_h)$	The first value of the universe of respective variable v_h.
m_{max}	Maximum number of chromosomes at Level 1.
$m_w(v_h)$	The average between $m_{1,w}(v_h)$ and $m_{2,w}(v_h)$.
n	Number of input variables.
N	Number of fuzzy rules.
N_p	Output horizon.
N_u	Control horizon.
p	p-step ahead prediction.
p_m	Mutation probability.
p_t	Parameter of $\mathbf{t}_5$ t-norm.
$r(k+p)$	Future reference trajectory.
R_r	Random point of crossover.
R_i	The i-th fuzzy rule.
$R_w(v_h)$	Parameter of the membership function w of the variable v_h.
t_{max}	Maximum number of chromosomes at Level 2.
$T_w(v_h)$	Parameter of the membership function w of the variable v_h.
$u(\cdot)$	Process input.
$u(k+p)$	An p-step ahead prediction of the controller.
v_{ij}	Center of the membership function i of the input variable j.
w	w-th membership function.
x_j	Input variable j of the fuzzy system.
$y(\cdot)$	Process output.
$\hat{y}(k+p)$	An p-step ahead prediction of the system.
$\hat{y}^i(l)$	i-th regression model output for the data l.
$y_d(\cdot)$	Desired output patterns.

Hierarchical Genetic Algorithm

i	i-th fuzzy system at Level 5.
j	j-th set of fuzzy rules at Level 4.
k	k-th individual fuzzy rule at Level 3.
m	m-th set of input variables and delays at Level 1.

t	t-th partition set individual at Level 2.
v_h	v_h-th variable of the fuzzy system.

Contents

Agradecimentos **i**

Abstract **iii**

Resumo **v**

Symbols and Abbreviations **vii**

Contents **xiii**

List of Figures **xvii**

List of Tables **xxiii**

List of Algorithms **xxv**

1 Introduction **1**

1.1 Motivation 1

1.2 Model Predictive Control Motivation 2

1.2.1 Industrial MPC Technology 3

1.2.2 The Nonlinear System Identification Procedure / Problems . . 4

1.3 Fuzzy Control Motivation 6

1.4 Thesis Contributions 7

1.5 Thesis Organization 8

2 Overview of Fuzzy Control for Industrial Processes **9**

2.1 Fuzzy Logic 9

2.2 Takagi-Sugeno Fuzzy Systems 11

2.3 Adaptive Fuzzy Control 12
2.4 Neural-Fuzzy Control 14
2.5 Fuzzy Genetic Algorithms 16
2.6 Fuzzy Model Predictive Control 17
2.6.1 MPC with Linear Models 20
2.6.2 MPC with Nonlinear Models 21
2.7 Summary / Conclusions 22

3 Concepts of Fuzzy Systems and Genetic Algorithms 23
3.1 Fuzzy Systems 24
3.1.1 Concept 24
3.1.2 Knowledge-Base 24
3.1.3 Fuzzifier 26
3.1.4 Fuzzy Inference Engine 27
3.1.5 Defuzzifier 28
3.1.6 Direct Adaptive Fuzzy Control 30
3.2 Modelling Using T-S Fuzzy Models 32
3.3 Genetic Algorithms 33
3.3.1 Concept 33
3.3.2 Chromosome 34
3.3.3 Initialization 34
3.3.4 Fitness Function 35
3.3.5 Selection 35
3.3.6 Crossover 35
3.3.7 Mutation 36
3.3.8 Replacement 37
3.3.9 Why to Use GAs to Design Fuzzy Systems 37

4 Hierarchical Genetic Fuzzy System for Identification 39
4.1 Introduction / State of the Art 40
4.2 Overview 43
4.3 Hierarchical Genetic Fuzzy System 44
4.3.1 T-S Fuzzy Model 44
4.3.2 Least Squares Method 45

4.3.3 Hierarchical Structure 46
4.4 Initialization Methods 53
4.4.1 Fuzzy c-Means Clustering Algorithm 53
4.4.2 Fuzzy c-Regression Model Clustering Algorithm 56
4.4.3 Recursive Least Squares Method With Adaptive Directional Forgetting 58
4.5 Hierarchical Genetic Fuzzy System With Initialization Methods . . . 60
4.6 Experimental Results 64
4.6.1 Wastewater Treatment System 65
4.6.2 Continuous-Stirred Tank Reactor 71
4.6.3 Real-World Setup of Two Coupled DC Motors 78
4.7 Conclusion 83

5 Hierarchical Genetic Fuzzy System for Control 85
5.1 Introduction / State of the Art 86
5.2 Hierarchical Genetic Fuzzy System 88
5.2.1 Fuzzy Control Rules 88
5.2.2 Hierarchical Structure 89
5.3 Initialization Methods 94
5.4 Hierarchical Genetic Fuzzy Systems With Initialization Methods . . . 94
5.4.1 HGA-Control Algorithm 98
5.4.2 AHGA-Control Algorithm 99
5.5 Experimental Results 102
5.5.1 Dissolved Oxygen Control 103
5.5.2 Real-World Control of Two Coupled DC Motors 111
5.6 Conclusion 117

6 Fuzzy Predictive Control 123
6.1 Introduction / State of the Art 123
6.2 Fuzzy Predictive Control Frameworks 125
6.2.1 Modelling Using T-S Fuzzy Models 127
6.2.2 Multiple Models 128
6.2.3 Predictive Control Law 129
6.2.4 Fuzzy Predictive Control Framework Algorithms 133

6.3 Experimental Results 134
6.3.1 Continuous-Stirred Tank Reactor 134
6.3.2 Real-World Control of Two Coupled DC Motors 139
6.4 Conclusion 142

7 Conclusions **143**

Bibliography **147**

List of Figures

1.1 Distribution of MPC applications versus the degree of process nonlinearity [Qin and Badgwell, 2000]. 4

2.1 Architecture of a generic adaptive fuzzy control scheme. 13
2.2 An example of a Neural Network. In this case, a Neuro-Fuzzy system is illustrated. 15
2.3 MPC strategy. 18
2.4 Block diagram of a predictive controller. 19

3.1 Basic configuration of a fuzzy logic system. 24
3.2 Examples of membership functions: a) Trapezoidal b) Triangular c) Gaussian. 25
3.3 Input variable, *Speed*, membership functions. 26
3.4 Output variable, $\Delta Force$, membership functions. 26
3.5 Mamdani minimum *implication* example. 29
3.6 Maximum rule *aggregation* example. 29
3.7 Example of the *defuzzification* methods of Table 3.2. 29
3.8 Basic genetic algorithm flowchart. 33
3.9 Example of the genetic structure used by a GA. 34
3.10 Roulette wheel selection. 36
3.11 Single point crossover of parents A and B to form offsprings C and D. 36
3.12 Uniform mutation of parent A, forming offspring B. 37
3.13 Weakest individuals replacement technique. 37

4.1 Encoding and hierarchical relations among the individuals of the different levels of the genetic hierarchy of the HGA approach. 47

4.2 Membership functions encoding in Level 2 of the HGA approach. . . 48
4.3 Flowchart of the HGA approach for learning the T-S fuzzy system. . 53
4.4 Encoding and hierarchical relations among the individuals of the different levels of the genetic hierarchy for the HGA-FCRM and AHGA-FCM approaches. 61
4.5 Plot of the variables of the WWTP data set:(a) y, (b) u_1, (c) u_2, (d) u_3, (e) u_4, (f) u_5, (g) u_6, (h) u_7, (i) u_8, (j) u_9, (k) u_{10}, and (l) u_{11}. . . 67
4.6 Modeling performance on the wastewater treatment system data set by the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies. The first half of the time interval of the graph corresponds to the training data set, and the second half corresponds to the test data set. 68
4.7 Membership functions of the proposed HGA method and of its best initial individual (HGA-INI) on Level 2, for all the delayed versions of the input variables (a) u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, and u_{11}; and (b) u_9, and u_{10}, for the WWTP process. 68
4.8 Membership functions of the proposed HGA-FCRM method and of their initialization method (HGA-FCRM-INI), for all the delayed versions of the input variable (a) u_1; (b) u_2; (c) u_3; (d) u_4; (e) u_5; (f) u_6, (g) u_7; (h) u_8; (i) u_9; (j) u_{10}; and (k) u_{11}, for the WWTP process. . . 70
4.9 Membership functions of the proposed AHGA-FCM method and of their initialization method (AHGA-FCM-INI), for all the delayed versions of the input variable (a) u_1; (b) u_2; (c) u_3; (d) u_4; (e) u_5; (f) u_6, (g) u_7; (h) u_8; (i) u_9; (j) u_{10}; and (k) u_{11}, for the WWTP process. . . 71
4.10 Evolution of the best fitness function value on Level 5 for all generations in the WWTP experiment by the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies. 72
4.11 Control signal used to compile the data set for the CSTR process. . . 73
4.12 Modeling performance on the CSTR data set by the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies. The first half of the time interval of the graph corresponds to the training data set, and the second half corresponds to the test data set. The output of the plant is $y(k) = C_A(k)$. 74

4.13 Membership functions of the proposed HGA, HGA-FCRM and AHGA-FCM methods and of their best initial individual (HGA-INI, HGA-FCRM-INI, and AHGA-FCM-INI, respectively) on Level 2, for all the delayed versions of the HGA input variables (a) C_A, and (b) q_c; the HGA-FCRM input variables (c) C_A, and (d) q_c; and the AHGA-FCM input variables (e) C_A, and (f) q_c, for the CSTR process. 75
4.14 Evolution of the best fitness function value on Level 5 of the proposed HGA and HGA-FCRM methodologies for all generations in the CSTR experiment (training data set). 76
4.15 Evolution of the best fitness function value on Level 5 of the proposed AHGA-FCM methodology for all generations in the CSTR experiment (training data set). 76
4.16 First data of the modeling performance on the CSTR train data set given by Figure 4.12. 77
4.17 The experimental scheme of the two coupled DC motors. 78
4.18 Control signal used to compile the data set on the DC motors process. 80
4.19 Modeling performance of the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies for the Motors data set. The first half of the time interval of the graph corresponds to the training data set, and the second half corresponds to the test data set. 81
4.20 Membership functions of the proposed HGA, HGA-FCRM, and AHGA-FCM methods and of their best initial individual (HGA-INI, HGA-FCRM-INI, and AHGA-FCM-INI, respectively), for all the delayed versions of the HGA input variables (a) y, and (b) u; the HGA-FCRM input variables (c) y, and (d) u; and the AHGA-FCM input variables (e) y, and (f) u, for the DC motors process. 81
4.21 Evolution of the best fitness functions value on Level 5 for the HGA, HGA-FCRM, and AHGA-FCM for all generations on the real DC motors process. 82

5.1 Encoding and hierarchical relations among the individuals of different levels of the genetic hierarchy of the HGA-Control approach. 90
5.2 Encoding and hierarchical relations among the individuals of different levels of the genetic hierarchy of the AHGA-Control approach. 100

5.3 General overview of the BSM1 plant [Belchior *et al.*, 2012]. 104
5.4 Performance of the FLC-BSM1 for the $DO_{ref}(t)$ trajectory (5.17) used to compile the learning data set for the BSM1 plant. 106
5.5 FLC target $\Delta KLa5$ commands (FLC-BSM1), and the corresponding command signals learned by the proposed HGA-Control and AHGA-Control methodologies for the BSM1 plant. 106
5.6 Evolution of the best individual fitness function value along the generations for the proposed HGA-Control and AHGA-Control methodologies for the BSM1 plant. 107
5.7 Membership functions of the proposed HGA-Control and AHGA-Control methods and of their initialization method (INICONTROL and INICONTROL-FCM, respectively), for all the delayed versions of the HGA-Control input variables (a) E; (b) ΔE, (c) $\Delta KLa5$; and the AHGA-Control input variables (d) E; (e) ΔE, (f) $\Delta KLa5$, for the BSM1 process. 108
5.8 (a) DO results of the proposed HGA-Control and AHGA-Control methodologies, of the respective initialization methods INICONTROL and INICONTROL-FCM, and of the AHGA-Control method with the adaptation law turned off until $t = 10\,[\text{days}]$, a method named as AHGA-Control-OFF, for a reference different from the one used for obtaining the BSM1 data set used for training the HGA methodologies; and (b) the respective applied $KLa5$ command signals. The red dashed line represents the time when the AHGA-Control methodology turns on the adaptation law (at $t = 10\,[\text{days}]$), for the AHGA-Control-OFF case. 109
5.9 Performance of the AFGPC for the $r(k)$ trajectory (5.18) used to compile the learning data set of the DC motor process. 114
5.10 AFGPC target command signal, and command signals learned by the proposed HGA-Control and AHGA-Control methodologies on the DC motor process. 114
5.11 Evolution of the best individual fitness function value along the generations for the proposed HGA-Control and AHGA-Control methodologies for the DC motors process. 115

5.12 Membership functions of the proposed HGA-Control and AHGA-Control methods and of their initialization methods (INICONTROL and INICONTROL-FCM, respectively), for all the delayed versions of the HGA-Control input variables (a) E; (b) ΔE, (c) Δu; and the AHGA-Control input variables (d) E; (e) ΔE, (f) Δu, for the DC Motor process. 116

5.13 (a) Results of the proposed HGA-Control and AHGA-Control methodologies, of the respective INICONTROL and INICONTROL-FCM initialization methods, and of the AHGA-Control method with the adaptation law turned off until $k = 880$, a method named as AHGA-Control-OFF, for a reference signal different from the one used for obtaining the DC motors data set used for training, and in the presence of load disturbances (lamps switched on for $360 \leqslant k \leqslant 680$) on the DC motors process; and (b) the respective applied command signals. The red dashed line represents the time when the AHGA-Control methodology turns on the adaptation law (at $k = 880$), for the AHGA-Control-OFF case. 119

6.1 A generic schematic diagram of the proposed FMMGPC control architecture. 126

6.2 A generic schematic diagram of the proposed AFGPC control architecture. 126

6.3 Performances of combinations among the M best models, $M = 1, \ldots, 10$, obtained by the HGA-FCRM method in the CSTR process, in Subsection 4.6.2 in Chapter 4. 135

6.4 (a) Results of the proposed FMMGPC and AFGPC frameworks in the presence of disturbance in the process flow rate q (for $13\,[\text{hours}] \leqslant t \leqslant 17\,[\text{hours}]$) in the CSTR process for the study Case 1; and (b) the respective applied command signal. 137

6.5 (a) Results of the proposed FMMGPC and AFGPC frameworks in the presence of disturbances on the feed concentration C_{A0} (for $6\,[\text{hours}] \leqslant t \leqslant 17\,[\text{hours}]$) and on the inlet coolant temperature T_{c0} (for $12\,[\text{hours}] \leqslant t \leqslant 20\,[\text{hours}]$) in the CSTR process for the study Case 2; and (b) the respective applied command signal. 138

6.6 Performances of combinations among the M best models, $M = 1, \ldots, 10$, obtained by the HGA-FCRM method in the real DC motors process, in Subsection 4.6.3 in Chapter 4. 139
6.7 (a) Results of the proposed FMMGPC and AFGPC frameworks and the GPC controller, in the presence of load disturbances (lamps switched on for $360 \leqslant k \leqslant 680$) in the real-world DC motors process; and (b) the respective applied command signal. The units for $y(k)$ and $r(k)$ are [pp/(0.25 sec)]. 141

List of Tables

3.1 Fuzzy inference operators: a) t-norms b) Mamdani implication methods c) Aggregation methods. 28

3.2 Defuzzification methods. 30

4.1 Variables of the wastewater treatment plant data set. 66

4.2 Comparison results of the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies with three state of the art non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and with two state of the art adaptive approaches: RPLS and ILLSA; for the WWTP test data set. 69

4.3 Variables of the continuous stirred tank reactor (CSTR) [Morningred *et al.*, 1992]. 73

4.4 Comparison results of the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies with three state of the art non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and with two state of the art adaptive approaches: RPLS and ILLSA; on the CSTR test data set. 75

4.5 Comparison of results of the proposed methodologies HGA, HGA-FCRM and AHGA-FCM with three state of the art non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and with two state of the art adaptive approaches: RPLS and ILLSA; for the DC Motors test data set. 80

5.1 Comparison of results of the proposed HGA-Control and AHGA-Control methodologies on the BSM1 train data set. 107

5.2 Fuzzy rule structure obtained with the initialization methods: (a) INICONTROL (used in HGA-Control); and (b) INICONTROL-FCM (used in AHGA-Control), for the BSM1 process. 110
5.3 Fuzzy rule structure obtained/learned by the HGA-Control method for the BSM1 process. 111
5.4 Fuzzy rule structure obtained/learned by the AHGA-Control method for the BSM1 process. 112
5.5 Comparison of results of the proposed HGA-Control and AHGA-Control methodologies, of the respective initialization methods INICONTROL and INICONTROL-FCM, and of the AHGA-Control method with the adaptation law turned off until $t = 10\,[\text{days}]$, a method named as AHGA-Control-OFF, for the BSM1 test. Note that the $MSE = \frac{1}{T}\sum_{t=1}^{T}\left(DO_{ref}(t) - DO(t)\right)^2$ is related with the tracking error instead of estimation error that is used in the train. T is the total number of sample on the test. 113
5.6 Comparison of results of the proposed HGA-Control and AHGA-Control methodologies on the Motor training data set. 115
5.7 Fuzzy rule structure obtained with the initialization methods: (a) INICONTROL (used in HGA-Control); and (b) INICONTROL-FCM (used in AHGA-Control), for the DC Motor process. 117
5.8 Fuzzy rule structure obtained/learned by the HGA-Control method for the DC Motor process. 118
5.9 Fuzzy rule structure obtained/learned by the AHGA-Control method for the DC Motor process. 118
5.10 Comparison of results of the proposed HGA-Control and AHGA-Control methodologies, of the respective initialization methods INICONTROL and INICONTROL-FCM, and of the AHGA-Control method with the adaptation law turned off until $k = 880$, a method named as AHGA-Control-OFF, for the DC motors process test. Note that the $MSE = \frac{1}{T}\sum_{k=1}^{T}\left(r(k) - y(k)\right)^2$ is related with the tracking error instead of estimation error that is used in the train. T is the total number of sample on the test. 120

List of Algorithms

2.1 Generic algorithm of model predictive control strategy. 19
4.1 Proposed HGA algorithm. 52
4.2 FCM algorithm. 56
4.3 FCRM algorithm. 58
4.4 Proposed initialization of the HGA methodology for the HGA-FCRM and HGA-FCM algorithms. 62
4.5 Proposed HGA-FCRM algorithm. 63
4.6 Proposed AHGA-FCM algorithm. 64
5.1 INICONTROL - Initialization algorithm of the antecedent and consequent membership functions to be used on the HGA-Control methodology. 95
5.2 INICONTROL-FCM - Initialization algorithm of the antecedent and consequent membership functions to be used on the AHGA-Control methodology. 96
5.3 Proposed initialization of the HGA-Control and AHGA-Control methodologies. 97
5.4 Proposed HGA-Control algorithm. 98
5.5 Proposed AHGA-Control algorithm. 103
6.1 Proposed FMMGPC algorithm. 133
6.2 Proposed AFGPC algorithm. 134

Chapter 1

Introduction

Contents

1.1 Motivation **1**

1.2 Model Predictive Control Motivation **2**

1.2.1 Industrial MPC Technology 3

1.2.2 The Nonlinear System Identification Procedure / Problems 4

1.3 Fuzzy Control Motivation **6**

1.4 Thesis Contributions **7**

1.5 Thesis Organization **8**

1.1 Motivation

Industrial processes have faced major changes in the market during the past decades, due to the increasing world competition (globalization of the market), and the environmental legislation that has been severely tightened in what concerns to the consumption and degradation of natural resources. The changes of the market resulted in hard constraints imposed by the need to reduce the consumption of energy and materials, and the environmental constraints imposed by legislation have in addition resulted in a significant increase of process complexity and costs of production equipment. Thus, more advanced process support systems will therefore be required due to these changes in process operation, and they can improve prod-

uct quality, reduce energy consumption and environmental emissions, and increase process safety/capacity.

Many industrial systems exhibit nonlinear behaviors and frequently have many complex characteristics, such as unknown and time-varying dynamics, constraints, and disturbances. In the future, due to worldwide industrial competition, industrial plants will require high levels of efficiency, flexibility, reliability and a control performance that can cover a wide process operation range and with load variations. Thus, it is desirable to develop efficient control strategies for uncertain nonlinear processes. Important problems to deal with include robustness, optimality, disturbance rejection and explicit constraint handling capabilities in controller synthesis, and, simultaneously, to enforce output tracking performance of the closed-loop control system.

Motivated by these problems, several works with model predictive control (MPC) and fuzzy logic control (FLC) for nonlinear industrial systems have recently been developed. In this context, this thesis focuses and proposes methodologies for control of uncertain nonlinear systems, in particular in the areas of FLC and MPC.

1.2 Model Predictive Control Motivation

The term model predictive control (MPC) does not designate a specific control method but an ample range of control methods which use a model of the process to predict the future output, and to obtain the control signal by minimizing an objective function that depends on the future behavior of the process. MPC originated in the late seventies and has been used successfully in industrial control applications [Richalet, 1993]. MPC quickly became popular particularly in chemical processes. However, most plants where MPC has been applied were linear. The reasons for this were that the identification of a linear model based on process data is relatively easy, and the quadratic optimization problem involved in MPC is easily solved for the linear prediction case. MPC is considered to be a mature control technique for linear systems, however, MPC for complex systems, such as nonlinear systems, is a field under current research.

1.2.1 Industrial MPC Technology

Some MPC algorithms are available commercially. A survey about MPC industrial applications with linear and nonlinear models is presented in [Qin and Badgwell, 2003], where the authors found, in 2003, more than 4600 MPC applications, over twice the number in their previous survey in 1997 [Qin and Badgwell, 1997]. Taking into account this number, it is safe to conclude that the usage of MPC technology is growing. MPC technologies have important desirable characteristics because they have the capability of controlling multivariable processes with a large number of inputs and outputs, and cope with constraints, complex dynamics, model uncertainty, and predict and take into account the future behavior of a plant [Qin and Badgwell, 2003].

The majority of applications (67%), mentioned in [Qin and Badgwell, 2003], are in the area of refining industries, one of the original application fields of MPC, where it has a solid background. The number of applications in petrochemicals (18.7%) and chemicals (4.9%) is also significant. Remaining applications are in areas such as pulp and paper, food processing, aerospace, automotive, furnaces, mining and metallurgy. The distribution of MPC applications versus the degree of process nonlinearity can be seen in Figure 1.1 (from [Qin and Badgwell, 2000]).

There exists several MPC control packages for industry applications, which highlights the interest of the industry for the application of MPC algorithms. In [Qin and Badgwell, 2003] ten commercial MPC control packages are referred, however, while some of these technologies integrate nonlinear MPC methodologies, most of them support only linear MPC. In the more than 4600 MPC applications before mentioned only 93 (2%) are nonlinear MPC applications. As a next generation of MPC technology, in [Qin and Badgwell, 2003] it is essentially suggested the adaptive MPC, the robust MPC, and that the next-generation MPC technology will allow nonlinear models combining process knowledge with operating data.

The next natural step in this area is the development of predictive control based on nonlinear models. The use of controllers that take into account the nonlinearities of the plant implies an improvement in the performance of the control system by reducing the impact of the disturbances and by improving the tracking capabilities of the control system.

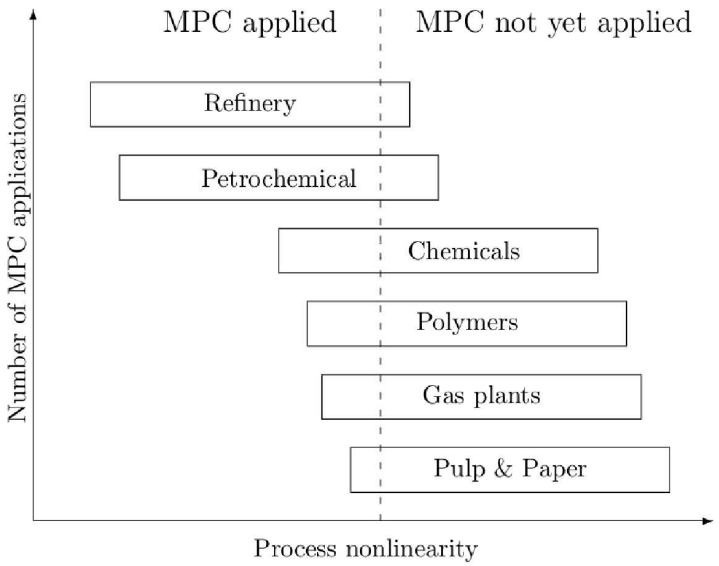


Figure 1.1: Distribution of MPC applications versus the degree of process nonlinearity [Qin and Badgwell, 2000].

1.2.2 The Nonlinear System Identification Procedure / Problems

As mentioned throughout this thesis, many industrial systems exhibit nonlinear behaviors and frequently have many complex characteristics, such as the multivariable nature of the systems, unknown and time-varying dynamics, constraints, and disturbances. A model of the process is used in MPC as a tool for controller design, where good controller performance can be achieved with good model performance (the best MPC controller certainly results from the "perfect" model) [Nelles, 2000]. The model must be accurate and the next step in MPC methodologies is to allow it to deal with nonlinear plants and models.

There are several steps, choices, factors/problems that have to be considered for successful system identification design [Nelles, 2000] as presented in the following list:

1. **Choice of the model inputs**: Some strategies can be used to choose the input variables: use all inputs, try all inputs combinations, unsupervised input selection (e.g. Principal Component Analysis (PCA) algorithm which discards

non-relevant inputs) and supervised input selection (the selection is related with the highest possible model accuracy (e.g. in linear models a standard tool is correlation analysis));

2. **Choice of the excitation signals**: This step requires prior knowledge about the process. It is important that the training data covers all operating conditions;

3. **Choice of the model architecture**: This step is mainly determined by the prior knowledge of the studied systems and the intended use of the model. The following factors are important when choosing appropriate model architecture (e.g. ARX, NARX, ARMAX, NARMAX): intended use of the model, dimensionality of the problem, quality and availability of data, training time, memory restrictions, and offline or online identification, etc. The advantages and drawbacks of different nonlinear dynamical representations are still under research;

4. **Choice of the model order, structure, and complexity**: This step is typically carried out by a combination of prior knowledge and a trial-and-error approach. The choice of higher order models increases the dimensionality of the problem and its complexity. Commonly, the model complexity is related to the number of parameters that the model possesses. The objective of this step is to simplify the model while keeping the capability of capturing the process behaviors. Order-reduction is one of the often met problems;

5. **Choice of the model parameters**: This step is usually carried out by the application of linear and nonlinear optimization techniques;

6. **Model validation**: The validation tests involve some procedures to assess how well the model relates to observed data, to prior knowledge, and to its intended use.

A common factor of all MPC methods is their assumption of the availability of an accurate model. This assumption may present problems, because many complex plants are difficult to be modelled mathematically based in physical laws, or have large uncertainties and strong nonlinearities, and as can be seen by the six

steps above, a successful system identification can be difficult to achieve in complex industrial processes.

The use of fuzzy models together with the concept of predictive control is a promising technique because both techniques can be explained in simple terms to operators, and it is theoretically supported by the fact that fuzzy logic systems are universal approximators. Motivated by these problems, this PhD work will also investigate the identification/design of the prediction models using fuzzy logic theory, and MPC methods using fuzzy models (Fuzzy Model Predictive Control).

1.3 Fuzzy Control Motivation

Fuzzy control systems have been used for a wide variety of industrial systems and consumer products, attracting the attention of many researchers. A major application of fuzzy logic theory has been in control of nonlinear systems, which are typically difficult to model and control. If the model of the plant is linear and known, then in many cases conventional control may be used to provide a solution. On the other hand, fuzzy logic control (FLC) should be used in situations where the, possibly nonlinear, mathematical model is poorly understood or is unknown, and where expert human knowledge (e.g. from experienced operators) is available and can describe the control of the plant. However, there still exist many difficulties in designing fuzzy systems to solve certain complex nonlinear problems, such as the fact that it is not easy to determine the most suitable fuzzy rules and membership functions to control the output of a plant, when the only available knowledge concerning the process is the empirical information transmitted by a human operator. Thus, a major challenge in current fuzzy control research is translating human empirical knowledge into FLCs. A possible candidate to meet this challenge is the application of the genetic algorithm (GA) approach to data extracted from a given process while it is being manually controlled.

GAs have been successfully applied to a wide variety of applications over the years, because GAs provide a robust search approach with the ability to find near optimal solutions in complex and large search spaces [Cordón *et al.*, 2001; Herrera, 2008]. Motivated by the above problems, this PhD work will also investigate the automatic design/extraction of all fuzzy parameters of a fuzzy logic controller from control data in order to control nonlinear industrial processes.

1.4 Thesis Contributions

This thesis has the following fundamental contributions:

1. *[Chapter 4], [Mendes* et al.*, 2012], [Mendes* et al.*, 2013a], [Mendes* et al.*, 2013b]:* Design of a new methodology for identification of industrial processes based on a Takagi-Sugeno (T-S) fuzzy model and on genetic algorithms (GAs). The learning of the T-S model is performed offline from input/output data to approximate unknown nonlinear processes by a hierarchical genetic algorithm (HGA). The proposed methodology is an automatic tool since it does not require any prior knowledge concerning the structure (e.g. the number of rules) and the database (e.g. antecedent fuzzy sets) of the T-S fuzzy model, and concerning the selection of the adequate input variables and their respective time delays for the prediction setting. Three approaches are proposed.

2. *[Chapter 5], [Mendes* et al.*, 2014]:* Design of a new methodology to automatically extract all fuzzy parameters of a fuzzy logic controller in order to control nonlinear industrial processes. The main objective is the extraction offline of a FLC from data gathered from a given process while it is being manually controlled. The learning of the FLC is performed by a HGA, from a set of input/output data obtained from a process under control. The selection of an adequate set of input variables, and the definition of the antecedent and consequent membership functions, the individual rules, the set of rules, and the fuzzy operators (t-norm, implication, aggregation and defuzzifier operators) which constitute the FLC, are performed.

3. *[Chapter 4], [Chapter 6], [Mendes* et al.*, 2013b]:* Design of a new methodology for fuzzy model predictive control of nonlinear time-varying systems without the knowledge about the mathematical model of the plant. The fuzzy system learned by the methodology proposed in Contribution 1 (see above) was incorporated into the control methodology. An initialization method was used on the hierarchical evolutionary approach developed in Contribution 1, and a combination of multiple learned T-S fuzzy models was used as the predictive model.

4. *[Chapter 4], [Chapter 6], [Mendes* et al.*, 2013a]:* Design of a new methodology for adaptive fuzzy model predictive control of nonlinear time-varying systems

without the knowledge about the mathematical model of the plant. The fuzzy system learned by the methodology proposed in Contribution 1 was incorporated into the control methodology. The proposed methodology includes an initialization method used for the hierarchical evolutionary approach developed in Contribution 1, and an adaptive approach for online tuning the fuzzy model consequent parameters.

1.5 Thesis Organization

The Thesis is organized as follows:

1. Chapter 2 gives an overview of fuzzy control for industrial applications. A description is given concerning standard fuzzy control and Takagi-Sugeno fuzzy control and their applications. Also, adaptive fuzzy control, and hybrid systems such as neuro-fuzzy control and genetic fuzzy control are mentioned in this overview. Finally, fuzzy model predictive control is described.

2. Chapter 3 provides an overview about the main concepts of fuzzy systems and genetic algorithms.

3. Chapter 4 describes the methodologies proposed for identification of industrial processes based on a T-S fuzzy model and on genetic algorithms.

4. Chapter 5 describes the methodology proposed to automatically extract all fuzzy parameters of a fuzzy logic controller, from data extracted from a given process while it is being manually controlled, in order to control nonlinear industrial processes.

5. Chapter 6 describes the methodologies proposed for fuzzy model predictive control for industrial processes.

6. Finally, Chapter 7 presents concluding remarks.

Chapter 2

Overview of Fuzzy Control for Industrial Processes

Contents

2.1 Fuzzy Logic 9

2.2 Takagi-Sugeno Fuzzy Systems 11

2.3 Adaptive Fuzzy Control 12

2.4 Neural-Fuzzy Control 14

2.5 Fuzzy Genetic Algorithms 16

2.6 Fuzzy Model Predictive Control 17

2.6.1 MPC with Linear Models 20

2.6.2 MPC with Nonlinear Models 21

2.7 Summary / Conclusions 22

2.1 Fuzzy Logic

The principles of fuzzy sets and fuzzy logic were developed by Lotfi A. Zadeh in 1965 [Zadeh, 1965]. In the late 1960s and early 1970s, fuzzy theory has grown to become a major scientific domain. Several new developments such as fuzzy algorithms, fuzzy decision making, etc, were proposed, and fuzzy logic and fuzzy systems grew as an independent field. In this period, Zadeh proposed fundamental concepts in fuzzy

logic theory. Zadeh proposed the concepts of fuzzy algorithms in 1968 [Zadeh, 1968], fuzzy decision making in 1970 [Bellman and Zadeh, 1970], and fuzzy ordering in 1971 [Zadeh, 1971]. An important paper published by Zadeh was [Zadeh, 1973], where he introduced the concepts of linguistic variables and proposed the use of fuzzy IF-THEN rules to formulate human knowledge, which established the foundation for fuzzy control [Wang, 1997].

The first fuzzy logic control (FLC) system was developed by Mamdani and Assilian [Mamdani, 1974; Mamdani and Assilian, 1975], to be used in a small steam engine. They found that the fuzzy controller was very easy to construct and worked with a good performance. The foundations of fuzzy theory were established in the 1970s, and the initial applications like the fuzzy steam engine controller, and the fuzzy cement kiln controller, have shown that the field was promising [Mamdani, 1974; Holmblad and Østergaard, 1995]. Since then, FLC has been extensively applied in a wide variety of industrial systems and consumer products and has attracted the attention of many researchers. A major application of fuzzy theory has been in control of nonlinear systems. Nonlinear processes are typically difficult to model and control. A lot of theoretical research has been developed in this field. If the model of the plant is linear and known, then in many cases conventional control may be used to provide a solution. On the other hand, fuzzy logic control should be used in situations where the, possibly nonlinear, mathematical model of the plant is poorly understood or unknown, and where expert human knowledge (e.g. by experienced operators) is available and can describe the control of the plant.

In the 1980s, several applications/researches have been done in the field of fuzzy control. In 1983, Sugeno began the pioneering work on a fuzzy robot, a self-parking car that was controlled by calling out commands [Sugeno and Nishida, 1985]. In the early 1980s, Yasunobu and Miyamoto [1985] began to develop a fuzzy control system for the Sendai subway. They finished the project in 1987 and created the most advanced subway system on earth [Wang, 1997]. In [Lim, 1995], a real-time experimental study of a fuzzy PID control of a DC motor is presented. More recently, more industrial applications have been developed and proposed. In [Ramírez *et al.*, 2004], a fuzzy controller is proposed for controlling the temperature of a multiple hearth furnace plant where fast and extensive changes in operating conditions occur, complicated by nonlinear and time-varying behavior of the process and interaction between the different variables. In [Wakabayashi *et al.*, 2009] a fuzzy control in a

semi-batch reactor for the production of nylon is presented. In [Cao *et al.*, 2013] a fuzzy control is applied on a ball mill pulverizing system. Fuzzy control has also been applied to a variety of servo systems and actuators in mechatronics [Kalyoncu and Haydim, 2009; Kim and Jeon, 2011; Zhao *et al.*, 2011].

Fuzzy logic systems are also used for modelling nonlinear plants, and may be used to approximate unknown nonlinear functions that compose the plant model. This approach is theoretically supported by the fact that fuzzy logic systems are universal approximators [Wang and Mendel, 1992; Kosko, 1994]. Several researches have been done is fuzzy identification [Babuska and Verbruggen, 1995, 1996; Gómez-Skarmeta *et al.*, 1999; Pishvaie and Shahrokhi, 2006; Sadrabadi and Zarandi, 2011].

2.2 Takagi-Sugeno Fuzzy Systems

An important type of fuzzy system is the Takagi-Sugeno (T-S) fuzzy system [Takagi and Sugeno, 1985]. Such type of fuzzy system has gained much popularity because of its rule consequent structure. The main difference between T-S fuzzy systems and other fuzzy systems is that the consequent of a T-S fuzzy system is a real-valued function, rather than a fuzzy set. Since their rule consequents are usually local linear or affine models corresponding to different operating points, T-S systems allow the designers to take advantage of conventional linear systems methodologies to analyze and design nonlinear systems. Local linear controllers can be easily designed in each operating region defined by fuzzy rule base and then produce the global nonlinear control by fuzzy weighted integration with several rules. T-S fuzzy models with the simplified linear rule consequent are also universal approximators capable of approximating any continuous nonlinear function, or any nonlinear system with continuous constituent functions [Ying, 1997]. However, when compared to the "standard" (Mamdani) fuzzy systems, in T-S fuzzy systems the power of interpretation decreases, but the consequents have a larger number of parameters, which offers more flexibility.

Several works using T-S fuzzy systems on identification and/or control have been done, where the identification of T-S fuzzy models is of great importance generally for FLC designs. In [Precup *et al.*, 2012] an improvement of T-S fuzzy control system using iterative feedback tuning (IFT) is applied on a laboratory setup composed by three tank systems. In [Al-Hadithi *et al.*, 2012] it is presented an approach to improve

the local and global modelling capability of a T-S fuzzy model. This method is also incorporated into a global fuzzy controller based on Linear Quadratic Regulator (LQR). An inverted pendulum and Van der Pol system are chosen to evaluate the robustness and performance of the proposed method. In [Lam and Lauber, 2013] the stability of a fuzzy model based control system, formed by a T-S fuzzy model and a fuzzy controller connected in a closed loop, is studied.

The above mentioned works, similarly to several of the other works in fuzzy control do not consider controller online auto-adaptation mechanisms to take into account and overcome complex and/or unknown time-varying plant behavior and system disturbances, and in particular to improve performance under such conditions. In off-line training algorithms, the discrete-time fuzzy system can be obtained from input-output data collected from a plant. However, such offline collected data set can be limited (including limited scope of coverage of the several plant operating situations) and the obtained fuzzy systems may not provide adequate accuracy. This motivates the introduction of adaptive control methodologies to solve the problem.

2.3 Adaptive Fuzzy Control

As mentioned in Sections 2.1 and 2.2, non-adaptive fuzzy control has been proving its performance in many applications. However, it is sometimes difficult to specify the rule base for plants that have unknown and/or time-varying model and parameters. In order to take into account these problems, adaptive fuzzy control (AFC) techniques have been investigated to make use of control structures whose parameters are unknown and/or time-varying, focusing on automatic on-line synthesis and tuning of fuzzy controller parameters. The main advantages of adaptive fuzzy control over nonadaptive fuzzy control are that adaptive fuzzy control has a better performance because it can adjust itself to changing environments, and less information about the plant is required because the adaptation law can help to learn and/or tackle the dynamics of the plant [Wang, 1997].

Adaptive fuzzy controllers can be classified into two categories [Wang, 1997]: direct and indirect adaptive controllers. In direct adaptive fuzzy control, the parameters of the controller are initially constructed from human control knowledge, and then iteratively adjusted to reduce the output error between the plant and a desired reference. In indirect adaptive fuzzy controllers, the parameters of the model of

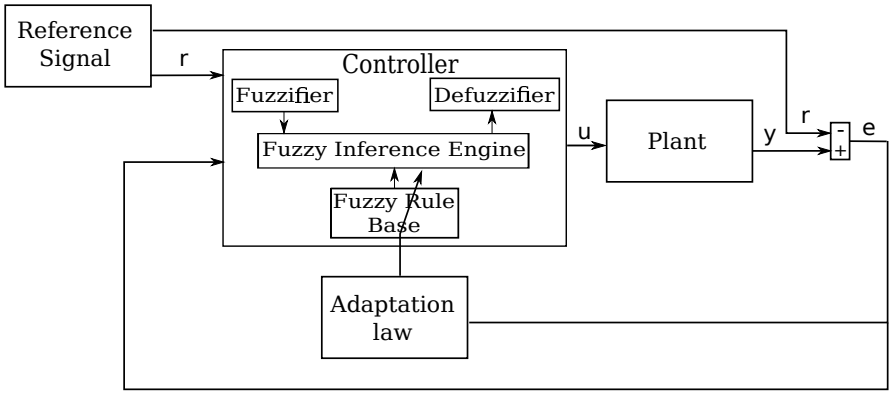


Figure 2.1: Architecture of a generic adaptive fuzzy control scheme.

the plant are initially constructed from some human knowledge about the unknown plant, and then iteratively adjusted to reduce the output error between the plant and an estimated model, while the current model parameters are used to indirectly adapt the controller. A generic adaptive fuzzy control scheme is represented in Figure 2.1. As can be seen, the scheme consists of the plant, the controller and the adaptation law. Adaptive fuzzy controllers are typically composed of fuzzy systems which have adjustable parameters to be adjusted by an adaptation law.

The first adaptive fuzzy controller was called the linguistic self-organizing controller and was introduced in [Procyk and Mamdani, 1978]. Later, the fuzzy model reference learning controller (FMRLC) was introduced in [Layne and Passino, 1993], and has shown to be successful by several studies in simulation [Layne and Passino, 1993; Kwong *et al.*, 1994; Passino *et al.*, 1995; Layne and Passino, 1996; Kwong and Passino, 1996; Lennon and Passino, 1999], and in implementation studies [Moudgal *et al.*, 1995; Zumberge and Passino, 1998].

The most studied stable adaptive fuzzy control (AFC) schemes are based on feedback linearization [Wang, 1996, 1992; Spooner *et al.*, 2002], to control nonlinear plants. They use fuzzy systems to approximate the unknown nonlinear plant in indirect schemes, or to approximate the unknown control law in direct schemes. They consider the parameters of the antecedent membership functions to be fixed, and the consequent parameters are adapted based on the tracking error by stable adaptive laws derived through Lyapunov synthesis. As an improvement, Hojati and Gazor [2002] have shown that composite adaptation laws can improve the perfor-

mance and the parameters convergence. Composite adaptive laws are based on both the tracking error and the modeling error to adjust the consequent parameters.

More recently, adaptive fuzzy control has been used in industry applications. Rubaai *et al.* [2007] proposed an embedded adaptive fuzzy control structure that was implemented for trajectory tracking control of a brushless servo drive system. Salehia *et al.* [2009] presents an adaptive control scheme for a *pH* neutralization process, where, to experimentally evaluate the performance of the proposed scheme, a bench-scale *pH* setup was used. The process consists of acid and base streams, being fed into the neutralization process tank. In [Mendes *et al.*, 2011], an architecture for adaptive fuzzy control of industrial systems is presented. The control methodology can integrate *a priori* knowledge about the control and/or about the plant, with on-line control adaptation mechanisms to cope with time-varying and/or uncertain plant parameters. The paper presents the fuzzy control software architecture that can be integrated into industrial processing and communication structures. An experimental benchmark composed of two mechanically coupled electrical DC motors has been employed to study the performance of the presented control architectures. In [Dounis *et al.*, 2013] it is proposed a methodology for designing a maximum power point tracking controller for photovoltaic systems using a fuzzy gain scheduling (FGS) methodology for a PID controller with adaptation of scaling factors for the input signals of FGS.

2.4 Neural-Fuzzy Control

Another form of computational intelligence structure is the neural network (NN). Network computing is a class of computational and learning methodologies inspired by networks or circuits of biological neurons. Advances in computer technology allow the construction of powerful artificial neural networks that approximate some features and capacities of biological systems. NNs are powerful data modelling tools.

Hybrid architectures were proposed to combine neural networks and fuzzy inferencing, the fuzzy neural networks (FNN) (Figure 2.2).

Lin and Lee [Lin and Lee, 1991, 1994] implemented a self-learning fuzzy logic system embedded in a five layer neural network. A modified version of the error backpropagation algorithm is used to train the network. FNN combine the advantage of these two methods: neural networks have the advantage in learning while

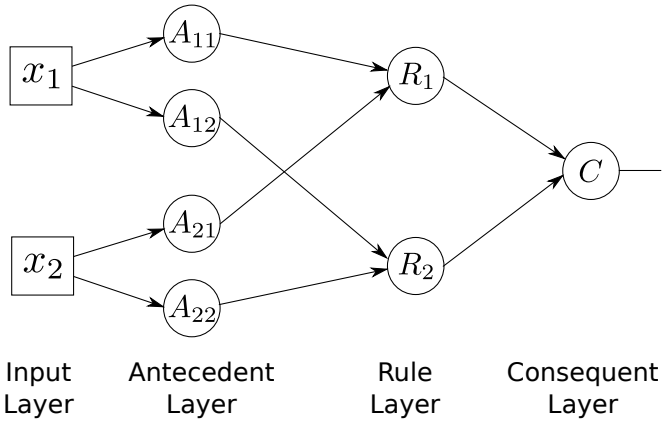


Figure 2.2: An example of a Neural Network. In this case, a Neuro-Fuzzy system is illustrated.

fuzzy systems have the advantage in inferencing, and human interpretation.

As an improvement of FNNs, recurrent fuzzy neural networks (RFNNs) were proposed. RFNNs include feedback connections, and internal memories [Juang and Lin, 1999; Lin and Wai, 2001; Zhou and Xub, 2001], and connections between neurons may form directed cycles. RFNNs have been shown to be more suitable for describing dynamic systems than FNN, because they can deal with time-varying inputs or outputs through their own natural temporal operation. Moreover, such networks can be functionally interpreted using fuzzy inference mechanisms, and they have been shown to be applicable to many engineering fields, such as image processing, control, signal processing, robotics, speech recognition, etc.

More recently, fuzzy neural networks have been used in industry applications. In [Waewsak *et al.*, 2010], the developed neuro-fuzzy control system was used with the available operational input variables (pH, total volatile acids (TVA), and alkalinity (Alk)) for controlling the influent feed flow rate into the anaerobic hybrid reactor in a wastewater treatment and biogas production. In [Wang *et al.*, 2010] a fuzzy adaptive back-propagation neural network is applied in a thermal conductivity gas analyzer. Taylan and Darrab [2011] introduce a systematic approach for the design of an adaptive neuro-fuzzy inference system (ANFIS) [Jang, 1993] for latex weight control of level loop carpets. The ANFIS inference architecture combines an adaptive neural network structure and fuzzy inferencing. In [Zhou *et al.*, 2013] is presented the development and application of four ANFIS models for solving four real-life

problems encountered in operation of the CO_2 capture process system.

2.5 Fuzzy Genetic Algorithms

Other type computational intelligence systems inspired on biological systems is the genetic algorithms (GAs). GAs are optimization methods, founded on "Survival of the fittest" Darwin's evolutionary theory [Darwin, 1859] consisting in having biological systems self-organize and adapt to their environments. Evolutionary computing was introduced in the 1960s by I. Rechenberg in the work "Evolution strategies" [Rechenberg, 1973]. Furthermore, GAs were proposed by John Holland, which developed this idea in his book "Adaptation in Natural and Artificial Systems" [Holland, 1975].

GAs are composed by a population of potential solutions to a problem. Each individual of the population represents a particular solution to the problem, generally expressed in some form of genetic code. The genetic algorithm system uses feedback from the interaction with the environment to find adequate solutions to problems. The population is evolved, over generations, to produce better solutions to the problem. GAs have proven to be useful in solving a variety of search and optimization problems.

Integration of fuzzy logic and genetic algorithms has been accomplished by the following two different approaches: 1) the application of GAs in optimization and search problems related with fuzzy systems (e.g. learning fuzzy systems), and 2) the use of fuzzy tools for modeling different GA components or adapting GA control parameters, with the goal of improving performance. Since a major challenge in current fuzzy logic research is learning good fuzzy systems for nonlinear systems, when expert knowledge may be not accurate or complete, in this thesis only approach 1 is considered. This approach is named as fuzzy genetic algorithm (FGA).

The typical parameters adjusted in fuzzy systems, or in fuzzy control, by GAs are the type and the parameters of the antecedent and consequent membership functions. In [Cazarez-Castro *et al.*, 2010], a hybrid architecture is presented, which combines a Type-1 (standard fuzzy logic) or Type-2 fuzzy logic system (FLS) and GAs for the optimization of the FLS membership function parameters, in order to solve the output regulation problem of a servomechanism with nonlinear backlash. Sharkawy [2010] developed a self-tuning PID control scheme with an application to

antilock braking systems (ABS) via combinations of fuzzy and genetic algorithms. In [Öztürk and Çelik, 2012] it is studied the speed control of a permanent magnet synchronous motor (PMSM) with a genetic based fuzzy controller, where the main goal is to obtain an optimal fuzzy controller without expert knowledge and to increase the controller performance including as assessed by the overshoot, rise time, and steady-state error parameters.

Genetic algorithms have also been used for nonlinear system identification. In [Pettersson *et al.*, 2007], a GA-based multi-objective optimization technique was utilized in the training process of a feedforward neural network, using noisy data from an industrial iron blast furnace. In [Du and Zhang, 2008], a new encoding scheme is presented for learning the Takagi-Sugeno fuzzy model from data by genetic algorithms. In [Tzeng, 2010], a fuzzy wavelet neural network model that uses a GA approach for adjusting parameters is introduced for function approximation. In [Ziaii *et al.*, 2012] it is investigated the effectiveness of a GA-based neuro-fuzzy system to identify and separate geochemical anomalies.

2.6 Fuzzy Model Predictive Control

Predictive control, or Model Based Predictive Control, or simply Model Predictive Control (MPC), originated in the late seventies and it has been used successfully in industrial control applications [Richalet, 1993]. The essence of predictive control is based on three key elements:

- Use of a model to predict the future output of the process based on the historical information;
- Calculation of an optimal control action based by minimizing an objective function;
- Feedback correction.

Predictive control is a control strategy based on a predictive model of the process under control, which is used to predict the future output based on the historical information of the process as well as the future input. The predictive model has the capability of showing the future behavior of the system. Therefore, the designer can experiment with different control laws to see the resulting system output, using

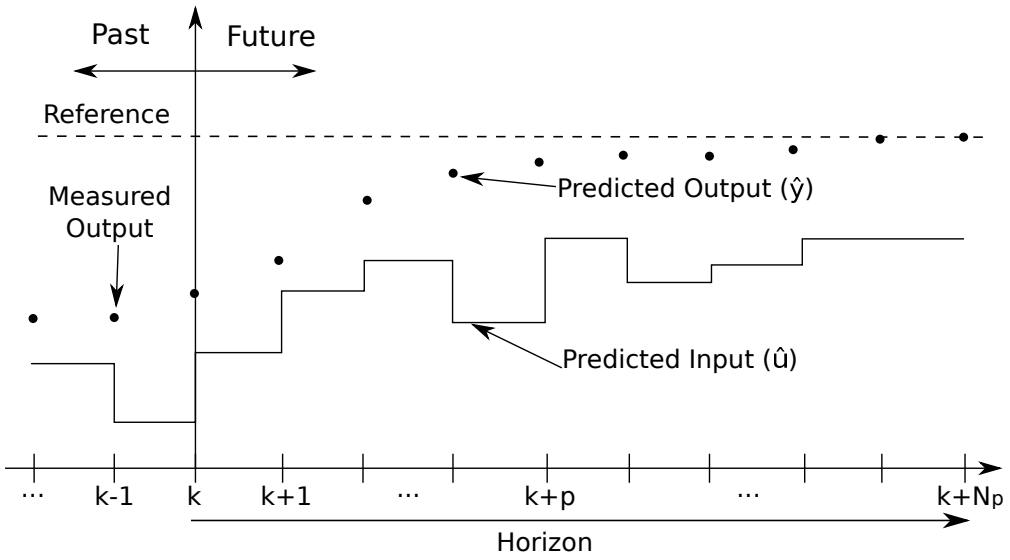


Figure 2.3: MPC strategy.

computer simulation. The term MPC does not designate a specific control method but an ample range of control methods which use a model of the process to obtain the control signal by minimizing an objective function.

The MPC strategy is commonly represented by the following principles (see Figure 2.3):

- At each time instant k, the process outputs, $y(k+p)$, $p = 1, \ldots, N_p$, are predicted using a process model over a predetermined finite prediction horizon N_p. This prediction model, $\hat{y}(k+p|k)$, depends on the knowledge of the values of system variables up to instant k (commonly, the model depends on past input and output variables);

- The reference trajectory, $r(k+p|k)$, $p = 1, \ldots, N_p$, that describes the future behavior of the process, is defined over the prediction horizon, N_p;

- The measurement of the output, $y(k)$, is available for feedback and state estimation;

- The control sequence, $u(k+p|k)$, $p = 0, \ldots, N_u - 1$, where N_u is the control horizon, is calculated by minimizing an objective function which is commonly

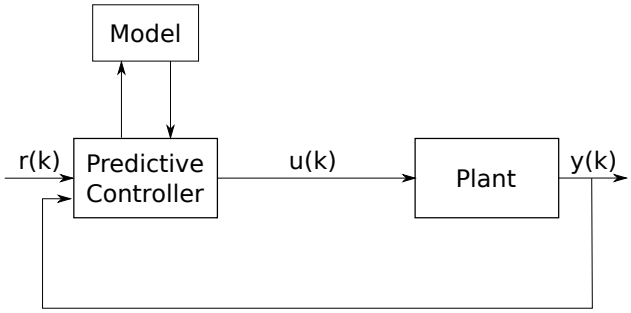


Figure 2.4: Block diagram of a predictive controller.

Algorithm 2.1 Generic algorithm of model predictive control strategy.

1. Sample the inputs and the output of the plant to be controlled;
2. Use the constructed model of the plant to predict its future behavior over a prediction horizon;
3. Design the objective function to be minimized, and possible constraints to be met. The objective function is commonly dependent on the error between the predicted future output and the future reference trajectory, and on the control effort of the predicted control sequence;
4. Calculate the optimal control sequence that minimizes the objective function and also taking into consideration the constraints.

formed by a quadratic function of the error between the predicted future output and the future reference trajectory, $e(k+p|k) = \hat{y}(k+p|k) - r(k+p|k)$, $p = 1, \ldots, N_p$, and of the control effort $\Delta u(k+p|k) = u(k+p|k) - u(k+p-1|k)$, $p = 0, \ldots, N_u - 1$, for the predicted control sequence;

- The control action applied to the real process at each instant k, $u(k|k)$, is the first element of the control sequence $u(k+p|k)$, $p = 0, \ldots, N_u - 1$.

A generic MPC control scheme is represented in Figure 2.4. As can be seen, the scheme consists of the plant, the predictive controller and the model of the plant to be controlled. In order to summarize this section which presents the key elements of MPC, a generic algorithm of the model predictive control strategy is described in Algorithm 2.1.

2.6.1 MPC with Linear Models

From the end of the 1970s, various works appeared showing interest in MPC in industry, such as the article of Richalet *et al.* [1978] which proposed model predictive heuristic control (later known as model algorithmic control (MAC)), and the work of Cutler and Ramaker [1980] which proposed the dynamic matrix control (DMC) algorithm which enjoyed great popularity. In both algorithms, a dynamic model of the process is explicitly used (impulse response in [Richalet *et al.*, 1978] and step response in [Cutler and Ramaker, 1980]) to predict the effect of the future control actions at the output.

MPC quickly became popular particularly in chemical processes. The generalized predictive control (GPC) algorithm [Camacho and Bordons, 2007], [Clarke, 1988], has emerged as the most popular algorithm in MPC. GPC generates a sequence of future control signals within each sampling interval in order to optimize the control effort of the controlled system and the output error. It is a model-based control method where a plant model is used to obtain a predictor model. The GPC has been applied in various plants, and has shown good performance results [Clarke, 1988], [Tham *et al.*, 1991].

However, most plants where MPC has been applied are linear. The reasons for this are: the identification of a linear model based on process data is relatively easy, linear models provide good results when the plant is operating in the neighbourhood of a specific operating point, and the quadratic optimization problem involved in MPC is easily solved for the linear prediction case. MPC is considered to be a mature control technique for linear systems. More complex systems, such as nonlinear, are a research field under study. In the surveys published by Qin and Badgwell [1997, 2003], some applications of MPC on nonlinear systems can be found, such as in refining, petrochemicals, and chemical industries, but they are still limited.

The next natural step in this area is the development of predictive control for nonlinear plants, based on nonlinear models. Such an approach is named nonlinear model predictive control (NMPC). The use of controllers that take into account the nonlinearities of the plant, and reducing the impact of disturbances, implies an improvement in the performance of the closed loop system, including the improvement of the tracking capabilities of the control system.

2.6.2 MPC with Nonlinear Models

The first approach to design a NMPC was the linearisation of the plant model [Rossiter *et al.*, 1991; Zhu *et al.*, 1991]. However, this approach may not predict exactly because the operating point may change and the predictor does not remain valid. A difficulty of all MPC methods is their assumption of an accurate model. This assumption may present problems, because many complex plants are difficult to be modelled mathematically based in physical laws, or by system identification methods (e.g. [Ljung, 1987]), or have large uncertainties and strong nonlinearities. An alternative for modelling nonlinear plants are fuzzy logic systems, or fuzzy models.

Fuzzy systems may be used to approximate unknown nonlinear functions of the plant model. This is theoretically supported by the fact that fuzzy logic systems are universal approximators [Wang and Mendel, 1992; Kosko, 1994]. The use of fuzzy models together with the concept of predictive control is a promising technique because both techniques can be explained in simple terms to operators. Takagi-Sugeno fuzzy models have gained much popularity in MPC because their rule consequents are real-valued functions. In [Skrjanc and Matko, 2000] predictive functional control is combined with a fuzzy model of the process, where a real-time experiment was realized on a heat-exchanger plant. In [Su *et al.*, 2006] an adaptive predictive control method based on T-S fuzzy models is proposed for discrete-time nonlinear systems, where the consequent parameters of the T-S fuzzy model are identified by a weighted recursive least squares method. In [Liu *et al.*, 2008] a locomotive brake control method based on T-S fuzzy MPC is proposed. A fuzzy clustering method is used to determine initial parameters, and a back-propagation algorithm is used for parameters adaptation by off-line learning. In [Wen and Liu, 2009] a fuzzy MPC method using T-S fuzzy systems was proposed for nonlinear plants subject to actuator saturation, and was tested on a continuous stirred tank reactor (CSTR).

There are also many existing predictive control strategies based on FNN or RFNN models. For example, Zhang and Morris [1999] presented a type of nonlinear model-based long-range predictive controller based on RFNN modeling. More recently, [Lu and Tsai, 2007] has presented a design methodology for predictive control of industrial processes using RFNN. A learning algorithm adopting a recursive least squares approach was employed to identify the unknown parameters in the

model. A physical variable-frequency oil-cooling machine was used to demonstrate the effectiveness of the proposed method.

2.7 Summary / Conclusions

In this chapter a review of fuzzy control and its derivations for industrial applications was performed. Taking into account that the objective of the thesis is the control of nonlinear industrial processes, the author considers that the use of fuzzy logic, genetic algorithms, and model predictive control constitutes a step further in the implementation of modern control techniques for industrial processes. MPC is a good control paradigm to apply in industrial processes because it easily works with a multivariable control scenario, it allows the integration of the explicit consideration of state and control command constraints, a nonlinear model for prediction can be used, a specified performance criteria is minimized on-line, and online adaptation can be incorporated into the control methodology. Fuzzy systems can be tuned/designed in an easy way by genetic algorithms, and may be used to approximate unknown nonlinear functions of the plant model (they are universal approximators) in MPC in order to include an accurate model of the process. Moreover, genetic algorithms are possible strong candidate methodologies to design/extract all fuzzy parameters of a fuzzy logic controller in order to control nonlinear industrial processes, from process data obtained while the process is being manually controlled.

Chapter 3

Concepts of Fuzzy Systems and Genetic Algorithms

Contents

3.1 Fuzzy Systems 24

3.1.1 Concept 24

3.1.2 Knowledge-Base 24

3.1.3 Fuzzifier 26

3.1.4 Fuzzy Inference Engine 27

3.1.5 Defuzzifier 28

3.1.6 Direct Adaptive Fuzzy Control 30

3.2 Modelling Using T-S Fuzzy Models 32

3.3 Genetic Algorithms 33

3.3.1 Concept 33

3.3.2 Chromosome 34

3.3.3 Initialization 34

3.3.4 Fitness Function 35

3.3.5 Selection 35

3.3.6 Crossover 35

3.3.7 Mutation 36

3.3.8 Replacement 37
3.3.9 Why to Use GAs to Design Fuzzy Systems 37

3.1 Fuzzy Systems

This section briefly overviews the main concepts of fuzzy systems. For a better understanding of this topic, [Wang, 1997] is recommended.

3.1.1 Concept

Fuzzy logic systems (FLS) are knowledge-based systems which make use of people's common sense and experience in the form of fuzzy *IF-THEN* rules. This type of systems are typically characterized by a group of four main elements: knowledge-base, fuzzifier, inference engine, and defuzzifier, as can be seen in Figure 3.1.

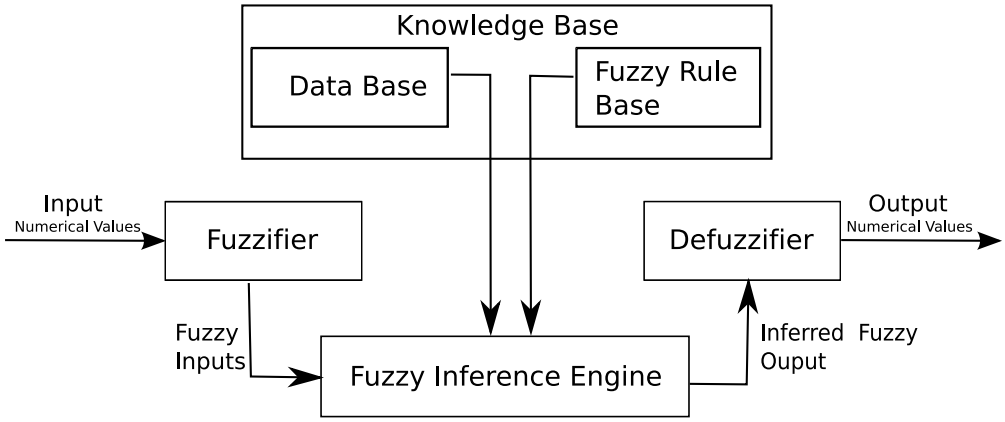


Figure 3.1: Basic configuration of a fuzzy logic system.

In the following subsections, a brief explanation of the elements that constitute the FLS represented by Figure 3.1 will be performed.

3.1.2 Knowledge-Base

The *knowledge-base* is one of the most important components of a fuzzy system, since all other components rely on it. The fuzzy rules are composed by two parts:

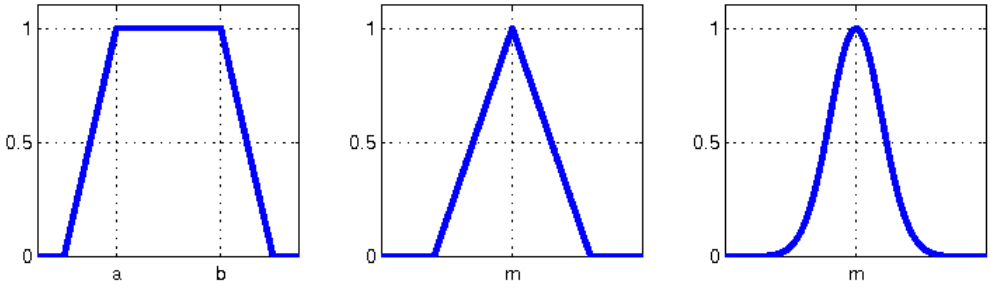


Figure 3.2: Examples of membership functions: a) Trapezoidal b) Triangular c) Gaussian.

the antecedent (*IF* part) and the consequent (*THEN* part). Therefore, a knowledge-base composed by a set of N fuzzy *IF-THEN* rules R_i can be represented in the form:

$$R_i : \text{IF } x_1(k) \text{ is } A_1^i, \text{ and } \ldots \text{ and } x_n(k) \text{ is } A_n^i \text{ THEN } u(k) \text{ is } B_i, \tag{3.1}$$

where A_j^i and B_i are the linguistic terms characterized by fuzzy membership functions $\mu_{A_j^i}(x) = U_j \rightarrow [0,1]$ and $\mu_{B_i}(u) = V \rightarrow [0,1]$, respectively; $i = 1, \ldots, N$, $j = 1, \ldots, n$; x_j $(j = 1, \ldots, n)$ are the fuzzy system input variables, and u is the output of the fuzzy system. $U_j \subset \mathbb{R}$ is the universe of discourse of x_j, for $j = 1, \ldots, n$, and $V \subset \mathbb{R}$ is the universe of discourse of u.

The most commonly used membership function types are the trapezoidal, triangular, and Gaussian membership functions, as represented in Figure 3.2.

The following example illustrates the construction of the knowledge-base of a FLC, in order to control the velocity of a car by a set of fuzzy *IF-THEN* rules:

IF the *Speed* of the car is *high*,
THEN apply *less Force* to the accelerator, (3.2)

IF the *Speed* of the car is *low*,
THEN apply *more Force* to the accelerator, (3.3)

IF the *Speed* of the car is *adequate*,
THEN *maintain* the *Force* applied to the accelerator, (3.4)

where *Speed* and *Force* ($\Delta Force$) represent the system's input and output linguistic

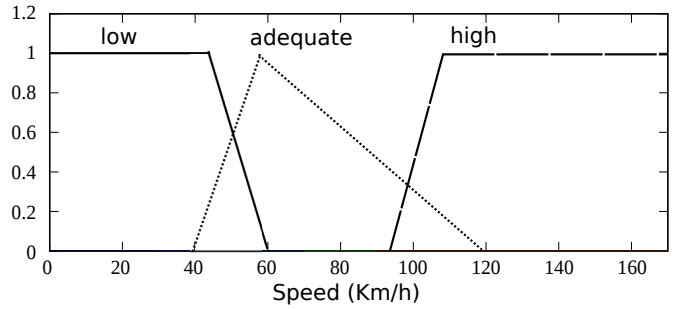


Figure 3.3: Input variable, *Speed*, membership functions.

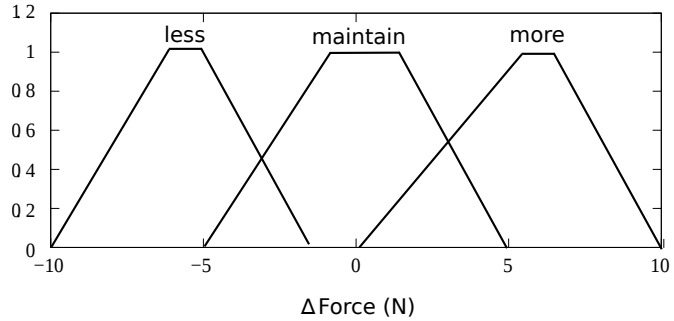


Figure 3.4: Output variable, $\Delta Force$, membership functions.

variables, respectively. The membership functions of the semantic terms that characterize the previous linguistic variables, *Speed* and *Force* ($\Delta Force$), can be seen in Figures 3.3 and 3.4, respectively. Figure 3.3 illustrates the membership functions of the semantic terms *low*, *adequate*, and *high* of the input linguistic variable, while Figure 3.4 presents the membership functions of the semantic terms *less*, *maintain*, and *more* of the output linguistic variable.

3.1.3 Fuzzifier

The next fuzzy system element is the *fuzzifier*. This element is responsible for mapping the real values, $\mathbf{x}^*$, of the vector of input linguistic variables, $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$, into corresponding fuzzy sets described by membership functions, where x_j, for $j = 1, \ldots, n$, are the input variables of the fuzzy system. The fuzzifier has the main goal of transforming the real-valued input vector $\mathbf{x}^* \in S \subset \mathbb{R}^n$ into a fuzzy set A' defined in a universe of discourse S. In the present thesis, only the *singleton fuzzifier* (3.5) is considered, due to its simplicity of implementation.

However, other fuzzifier methods can be consulted in [Wang, 1997]. The following representation illustrates the mapping performed by the *singleton fuzzifier*:

- **Singleton fuzzifier:**

$$\mu_{A'}(\mathbf{x}) = \begin{cases} 1, & \text{if } \mathbf{x} = \mathbf{x}^*, \\ 0, & \text{other cases,} \end{cases} \tag{3.5}$$

where $\mathbf{x}^*$ is the concrete input value.

3.1.4 Fuzzy Inference Engine

The next fuzzy system element is the *fuzzy inference engine* (FIE). The FIE uses the collection of fuzzy *IF-THEN* rules, where each rule i maps an input fuzzy set A' into the fuzzy rule consequent fuzzy set B_i. In a frequently used approach, the first step consists in processing each rule individually, and then the outputs of the fuzzy rules are combined into an overall inferred output fuzzy set Y. In fuzzy logic, the basic operations used to process the antecedent part of the rule are the following: *intersection*, *union*, and *complement*. Considering X_1 and X_2 as two fuzzy sets defined in a universe of discourse U, *intersection* can be denoted by $T = X_1 \cap X_2$; *union* by $S = X_1 \cup X_2$; and the *complement* of A by $C = \overline{X}_1$. Subsequently, these operations can be represented by norm operators. *intersection* is defined by a t-norm, *union* is given by a s-norm, and *complement* is determined by a c-norm [Wang, 1997]. In this thesis only the t-norm operators defined in Table 3.1(a) will be considered, since these are the most commonly used operators.

After calculating the antecedent value, the fuzzy propositions are then interpreted as fuzzy relations using an *implication* operator. In fuzzy logic, the sentences *IF* **A** *THEN* **B** can be modeled/written as $\mathbf{A} \rightarrow \mathbf{B}$, where **A** and **B** are the fuzzy propositions, whose values are fuzzy sets, and $\rightarrow$ is the fuzzy *implication* operator. The most commonly used implication methods are the Mamdani minimum and Mamdani product implications, which are described in Table 3.1(b).

The next step in the inference process is to aggregate the outputs of all fuzzy rules using an *aggregation* operator (∇). In this process, the output fuzzy sets of the rules are combined into an overall output fuzzy set of the system, which is used as the input value for the defuzzifier. Table 3.1(c) gives examples of aggregation

Table 3.1: Fuzzy inference operators: a) t-norms b) Mamdani implication methods c) Aggregation methods.

(a)

t-norm ($\cap$)	
Minimum	$min(\mu_{A_1}(x_1), \mu_{A_2}(x_2))$
Bounded product	$max(0, \mu_{A_1}(x_1) + \mu_{A_2}(x_2) - 1)$
Algebraic product	$\mu_{A_1}(x_1)\mu_{A_2}(x_2)$

(b)

Implication ($\rightarrow$)	
Minimum	$min(\mu_{A_1}(x_1), \mu_{A_2}(x_2))$
Product	$\mu_{A_1}(x_1)\mu_{A_2}(x_2)$

(c)

Aggregation (∇)	
Bounded sum	$min(\mu_{A_1}(x_1) + \mu_{A_2}(x_2), 1)$
Maximum	$max(\mu_{A_1}(x_1), \mu_{A_2}(x_2))$
Normalized sum	$\frac{\mu_{A_1}(x_1)+\mu_{A_2}(x_2)}{min(\mu_{A_1}(x_1)+\mu_{A_2}(x_2),1)}$

methods.

There is a variety of choices for the *fuzzy inference engine*, depending on the employed operators for the s-norms and t-norms, and for the implications and aggregation methods [Wang, 1997]. An example is the *product inference engine* which uses the Mamdani product implication, algebraic product for the t-norms and maximum for s-norms. Considering once again the example presented in Subsection 3.1.2, Figures 3.5 and 3.6 illustrate the implementation of Mamdani minimum *implication* and maximum rule *aggregation*, when the speed of the car reaches 55 [Km/h].

3.1.5 Defuzzifier

The *defuzzifier* is defined as a mapping from a fuzzy set B (in this case, associated to the output linguistic variable of the fuzzy inference engine) into a real valued output, u^*. There are also several possible choices for the *defuzzifier* to be used with the *fuzzy inference engine*. The choices are available for the control designer and can take into account, for example, the computational efficiency. Let b_i, and

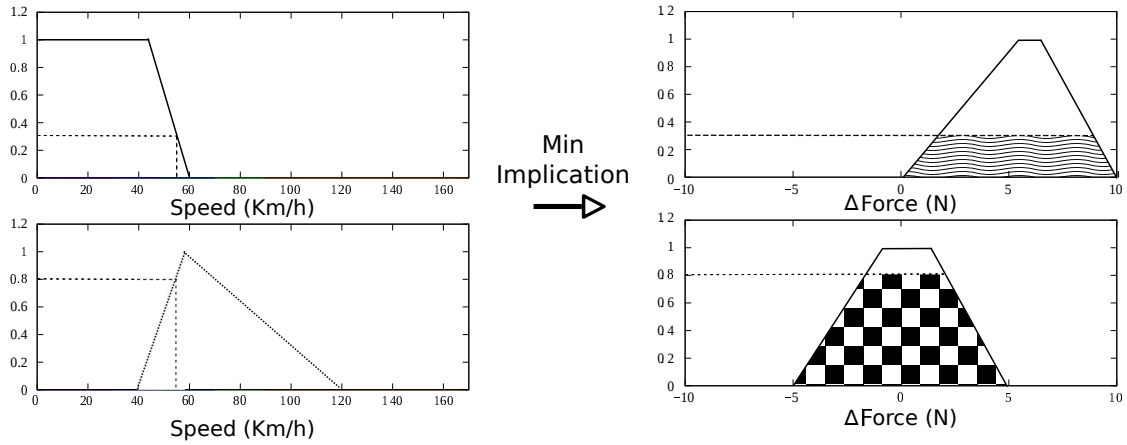


Figure 3.5: Mamdani minimum *implication* example.

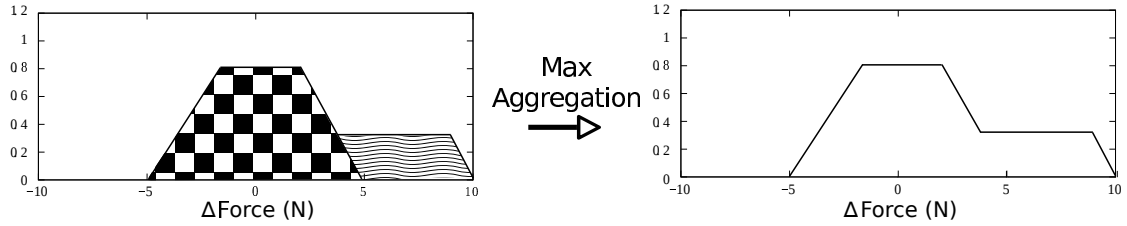


Figure 3.6: Maximum rule *aggregation* example.

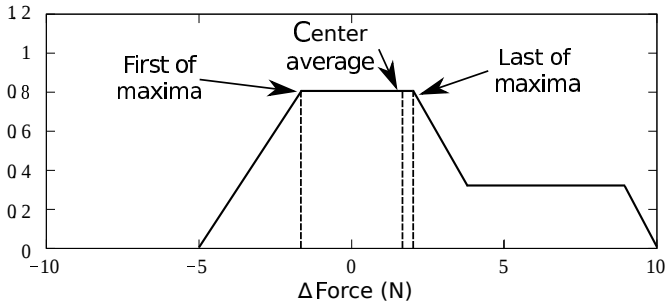


Figure 3.7: Example of the *defuzzification* methods of Table 3.2.

$hgt(V)$ be the center and height (maximum attained membership value) of fuzzy set V, respectively. Center of gravity, center of area, first of maxima, last of maxima, and the center average are commonly used defuzzifiers in fuzzy control (Table 3.2, and [Wang, 1997]). Figure 3.7 represents the application of some of the defuzzification methods of Table 3.2 (First of maxima, Center average and Last of maxima defuzzification methods) to the example considered in Figures 3.5 and 3.6 in Section 3.1.4.

Table 3.2: Defuzzification methods.

Defuzzification Methods	
Center of gravity	$u^* = \frac{\int_{min}^{max} u\mu_B(u)du}{\int_{min}^{max} \mu_B(u)du}$
Centre of area	$u^* = u', \int_{min}^{u'} \mu_B(u)du = \int_{u'}^{max} \mu_B(u)du$
First of maxima	$u^* = inf\{u \in hgt\,(B)\}$ $hgt\,(B) = \{u \in V \mid \mu_B\,(u) = \sup_{u \in V} \mu_B\,(u)\}$
Last of maxima	$u^* = sup\{u \in hgt\,(B)\}$ $hgt\,(B) = \{u \in V \mid \mu_B\,(u) = \sup_{u \in V} \mu_B\,(u)\}$
Center average	$u^* = \frac{\sum_{i=1}^{N} b_i hgt(B_i)}{\sum_{i=1}^{N} hgt(B_i)}$

3.1.6 Direct Adaptive Fuzzy Control

In order to obtain a better control performance, the control parameters of the learned/constructed FLC can be improved manually by using the information transmitted by a human operator, and/or by using direct adaptive control methodologies. The direct adaptive control methodology used (when a direct approach is used) in this thesis to adapt controller parameters was the same as the direct adaptive fuzzy control methodology employed in [Mendes *et al.*, 2011]. The adaptive control scheme is represented in Figure 2.1.

To design a direct adaptive fuzzy controller, knowledge about adequate plant control actions are used to design an initial control solution. The control knowledge will be expressed as a set of N fuzzy *IF-THEN* rules of the form (3.1).

To use the direct adaptive fuzzy controller as in [Mendes *et al.*, 2011], the fuzzy logic system (Figure 3.1) must be composed by: singleton fuzzifier, center-average defuzzifier, and the product inference engine. In these conditions, considering rules (3.1), the fuzzy system implements the following function:

$$u\,(\mathbf{x}) = \frac{\sum_{i=1}^{N} b_i \left(\prod_{j=1}^{n} \mu_{A_j^i}\,(x_j(k))\right)}{\sum_{i=1}^{N} \prod_{j=1}^{n} \mu_{A_j^i}\,(x_j(k))}, \tag{3.6}$$

where $\mathbf{x} = [x_1, \ldots, x_n]^T$ is the vector of inputs of the fuzzy system, and b_i is the center of B_i. Equation (3.6) can be rewritten as:

$$u(\mathbf{x}) = \mathbf{\Theta^T \Psi}(\mathbf{x}), \tag{3.7}$$

where $\mathbf{\Theta} = [b_1, \ldots, b_N]^T$ is a vector of adjustable parameters, and $\mathbf{\Psi}(\mathbf{x}) = [\overline{\omega}_1(\mathbf{x}), \ldots, \overline{\omega}_N(\mathbf{x})]^T$, where:

$$\bar{\omega}_i(\mathbf{x}) = \frac{\prod_{j=1}^{n} \mu_{A_j^i}(x_j(k))}{\sum_{p=1}^{N} \prod_{j=1}^{n} \mu_{A_j^p}(x_j(k))}, \quad i = 1, \ldots, N. \tag{3.8}$$

The adaptive law is derived by Lyapunov synthesis. For more details, references [Wang, 1996; Mendes *et al.*, 2011] are recommended. The following Lyapunov function is used [Wang, 1996]:

$$V = \frac{1}{2}\mathbf{e}^T\mathbf{P}\mathbf{e} + \frac{1}{2\gamma}(\mathbf{\Theta}^* - \mathbf{\Theta})^T(\mathbf{\Theta}^* - \mathbf{\Theta}), \tag{3.9}$$

where γ is a positive constant, $\mathbf{e}$ is the closed loop error defined in [Wang, 1996], and $\mathbf{\Theta}^*$ is the optimal value of the the vector of parameters, $\mathbf{\Theta}$, that optimizes the min-max approximation error between u and the ideal control u^*. $\mathbf{P}$ is a positive-definite matrix satisfying the following Lyapunov equation:

$$\mathbf{\Lambda}^T\mathbf{P} + \mathbf{P}\mathbf{\Lambda} = -\mathbf{Q}, \tag{3.10}$$

where $\mathbf{Q}$ is an arbitrary positive-definite matrix, and $\mathbf{\Lambda}$ is a matrix that can be designed to attain the desired error dynamics [Wang, 1996].

The employed adaptation law is [Wang, 1996]:

$$\dot{\mathbf{\Theta}} = \gamma \mathbf{e}^T\mathbf{p}\mathbf{\Psi}(\mathbf{x}), \tag{3.11}$$

where $\mathbf{p}$ is the last column of $\mathbf{P}$.

3.2 Modelling Using T-S Fuzzy Models

Takagi-Sugeno fuzzy systems with simplified linear rule consequents are universal approximators capable of approximating any continuous nonlinear function, or any nonlinear system with continuous constituent functions [Ying, 1997]. For more details about T-S fuzzy systems, references [Takagi and Sugeno, 1985; Wang, 1997; Ying, 1997] are recommended. In general, a discrete-time nonlinear system with continuous constituent functions can be described by a T-S fuzzy model defined by the following fuzzy rules:

$$\begin{aligned} R_i: \ & \text{IF } x_1(k) \text{ is } A_1^i, \text{ and } \ldots \text{ and } x_n(k) \text{ is } A_n^i \\ & \qquad \text{THEN } y_i(k) = \theta_{i1}x_1(k) + \ldots + \theta_{in}x_n(k), \\ & i = 1, \ldots, N, \end{aligned} \tag{3.12}$$

where R_i $(i = 1, \ldots, N)$ represents the i-th fuzzy rule, N is the number of rules, $x_1(k), \ldots, x_n(k)$ are adequately chosen input variables of the T-S fuzzy system. A_j^i $(i = 1, \ldots, N,\ j = 1, \ldots, n)$ are linguistic terms characterized by fuzzy membership functions $\mu_{A_j^i}(x_j)$ which describe the local operating regions of the plant. $\theta_{i1}, \ldots, \theta_{in}$ contain the model parameters of $y_i(k)$. Equation (3.12) is sufficiently general to the point that one of the input variables, let it be $x_1(k)$ for example, may be defined to have a so called dummy effect on (3.12) with the meaning that $x_1(k) \equiv 1$ in the consequent part of the rules, and $\mu_{A_1^i}(x_1) \equiv 1$, for $i = 1, \ldots, N$.

From (3.12), the output $y(k)$ of the T-S fuzzy model can be written as

$$y(k) = \sum_{i=1}^{N} \bar{\omega}_i[\mathbf{x}(k)]\mathbf{x}^T(k)\boldsymbol{\theta}_i, \tag{3.13}$$

where for $i = 1, \ldots, N$, $\bar{\omega}_i[\mathbf{x}(k)]$ is given by (3.8), and assuming Gaussian membership functions,

$$\mathbf{x}(k) = [x_1(k), \ldots, x_n(k)]^T, \tag{3.14}$$

$$\mu_{A_j^i}(x_j(k)) = \exp\left(-\frac{(x_j(k) - v_{ij})^2}{\sigma_{ij}}\right), \quad j = 1, \ldots, n, \tag{3.15}$$

$$\boldsymbol{\theta}_i = [\theta_{i1} \ldots, \theta_{in}]^T, \tag{3.16}$$

where v_{ij} and σ_{ij} represent, respectively, the center and the width of the Gaussian

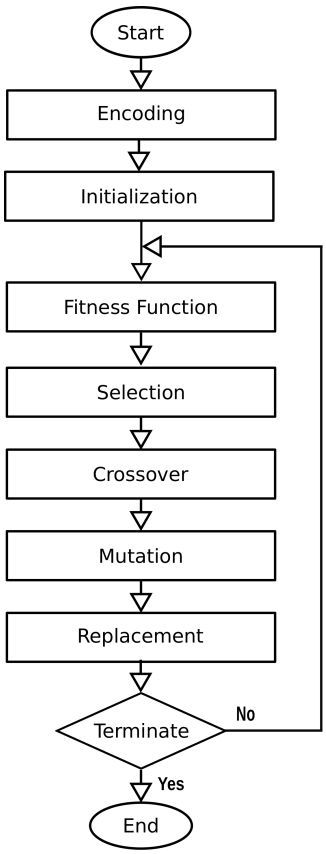


Figure 3.8: Basic genetic algorithm flowchart.

membership functions which need to be defined/learned.

3.3 Genetic Algorithms

This section briefly overviews the main concepts of the genetic algorithm (GA) optimization technique. For a better understanding of this topic, [Sivanandam and Deepa, 2007] is recommended.

3.3.1 Concept

Figure 3.8 shows a basic genetic algorithm flowchart. After the encoding of the *chromosomes*, an *initialization* of the population of chromosomes is performed. Each individual is evaluated using a *fitness function* that is specific to the problem being solved. Based on the fitness values, a number of individuals are chosen to be parents

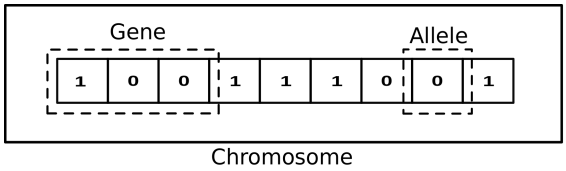


Figure 3.9: Example of the genetic structure used by a GA.

(*selection*). New individuals are then produced from the selected parents, using the reproduction operators, *crossover* and *mutation*. Finally, the individuals with the weakest fitness from the old generation are replaced by the new individuals (*replacement*). Each of these steps are described in more detail in the following subsections.

3.3.2 Chromosome

In the traditional GA, the representation used to characterize a solution/individual is a fixed-length bit (binary) string, know as *chromosome*. A set of positions in a *chromosome* is called a *gene*, and the possible individual bit values that represent a *gene* are known as *alleles*. The GA then handles the population of possible solutions/individuals, were each solution is represented through the *chromosome* (Figure 3.9). Encoding of chromosomes is the first part of the GA implementation, and it depends on the optimization problem at hands. The most common way of encoding is a bit string (binary encoded), where each bit in the string can represent some part of the solution. Other common ways of encoding is with real or integer numbers, i.e. real and integer encoding, respectively.

3.3.3 Initialization

After encoding of chromosomes, an initial population of chromosomes is generated. This step is named as *initialization*. The first population must offer a wide diversity of genetic information, and should be as large as possible to facilitate the convergence to the optimal solution. GAs are usually initialized with random population elements. However, this sort of approach increases the tuning/search difficulty of the GA, since a set of totally random populations can lead to a very exhausting optimality search, requiring more iterations to attain convergence. Thus, in order to reduce the computational cost and increase the GA's performance, initialization

methods will be proposed in Chapters 4 and 5.

3.3.4 Fitness Function

An important step is to define how the GA will select the fittest individuals. Each individual within the population is assigned a fitness value, which expresses how good the solution that it represents is at solving the problem. The fitness value determines how successful the individual will be at propagating its genes to subsequent generations. The best solution corresponds to an individual which maximizes the fitness function (if the GA is dealing with problems that involve the maximization of the fitness function). A common fitness function (to be maximized) for identification problems is $f = 1/MSE$, where $MSE = \frac{1}{L}\sum_{l=1}^{L}(y(l) - \hat{y}(l))^2$ is the mean square error, L is the number of data patterns, $\hat{y}(l)$ the predicted output pattern, and $y(l)$ is the target output pattern.

3.3.5 Selection

The *selection* operator chooses the individuals of the population that will create offsprings for the next generation. The purpose of selection is to emphasize fitter individuals and to give them priority to create offsprings for the next generations. Many *selection* techniques can be adopted. In this thesis the roulette wheel selection method is used. The principle of *roulette wheel* consists in a linear search of individuals through a roulette wheel, where the wheel slots are weighted in proportion to the individuals fitness values, as can be seen in Figure 3.10. In each generation, with the selection operator, two parents from the population are chosen for crossing.

3.3.6 Crossover

Crossover is the process of taking two parent solutions and producing offspring solutions from them. In this thesis, the single point crossover technique is used. As can be seen in Figure 3.11, the process consists of taking the two parents (A and B) selected with the selection operator and producing two offspring solutions (childs) from them. For the first child (C), the crossover process generates a random point of crossover, R_r, and the child will receive the alleles from 1 to R_r from the first parent

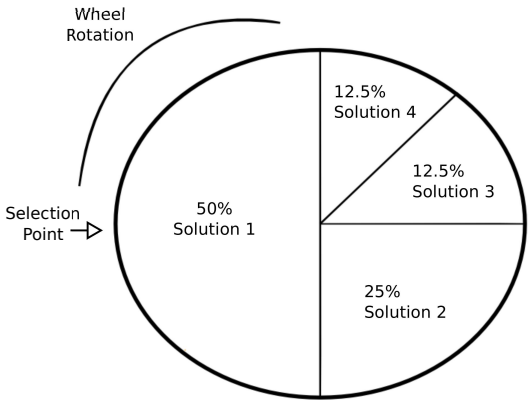


Figure 3.10: Roulette wheel selection.

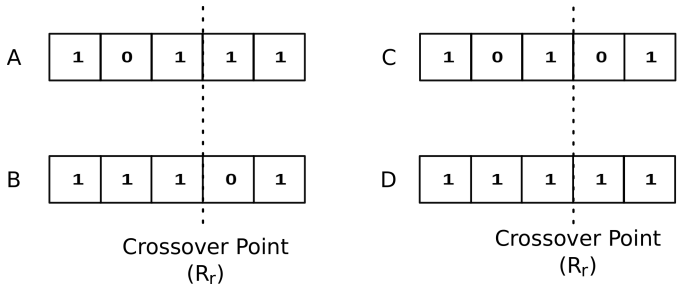


Figure 3.11: Single point crossover of parents A and B to form offsprings C and D.

(A) and the rest of the alleles are received from second parent (B). The second child (D) is constituted by the remaining alleles of the parents (A and B).

3.3.7 Mutation

The *mutation* operator is used to maintain the diversity of the population and to prevent the algorithm from being trapped in local minima. After crossover, each of the two chromosomes resulting from the crossover operator is subject to mutation with probability p_m. In binary-encoded chromosomes, the flip bit mutation technique is used, where the value of a random allele is inverted. In real and integer encoded chromosomes, in this work, uniform mutation is used, where the value of one randomly selected allele of the chromosome is replaced by a uniform random value selected between the upper and lower bounds defined for that allele. An illustration of this technique can be viewed in Figure 3.12.

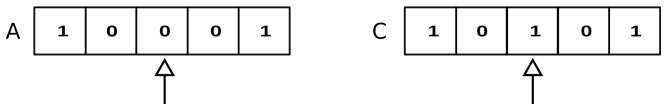


Figure 3.12: Uniform mutation of parent A, forming offspring B.

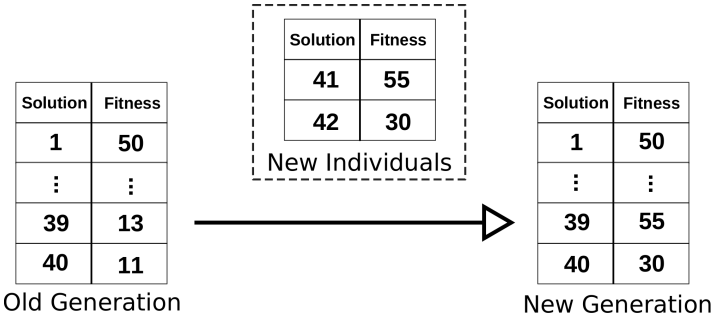


Figure 3.13: Weakest individuals replacement technique.

3.3.8 Replacement

Replacement is the last stage of the cycle. In this work the weakest individuals replacement technique is the employed replacement operator. As can be seen in Figure 3.13, it consists in taking two individuals with the weakest fitness from the old generation, and replacing them by the two new individuals that result from the application of the selection-crossover-mutation sequence of operators, in order to form the new population.

3.3.9 Why to Use GAs to Design Fuzzy Systems

Some biologically inspired algorithms, such as genetic algorithm (GA), ant colony optimization (ACO), and particle swarm optimization (PSO), have been proved efficient in optimization problems. GAs are search methods that are inspired on natural evolution, selection, and survival of the fittest in the biological world. PSO is inspired in the social behavior of living organisms such as bird flocking or fish schooling. ACO is a multiagent approach that simulates the foraging behavior of ants. All the above algorithms could be used to design the fuzzy systems. However, because GAs provide a robust search with the ability to find near optimal solutions in complex and large search spaces [Cordón *et al.*, 2001; Herrera, 2008], GAs are a useful soft computing technique to design fuzzy systems. Other advantages in the

use of GAs in the design of fuzzy systems are: GAs are simple to implement, they have the possibility of using different types of solution encoding (e.g. for different parts of the model), and they are adaptive, which means that they have the ability to learn, accumulating relevant knowledge to solve optimization problems [Kasabov, 1996].

A hierarchical genetic algorithm (HGA) will be used instead of a GA with just one optimization level due to the complexity of the problem of designing fuzzy systems, allowing the following parameters of the fuzzy systems to be learned: input variables and their respective time delays, antecedent fuzzy sets, consequent parameters, and fuzzy rules. It is well known that computation, search, and optimization problems become more difficult to solve when the dimensionality increases (curse of dimensionality), and, therefore, when more complex design decisions involving a large number of parameters must be made, a global formulation of the problem representing all the parameters in just one optimization level can be inadequate.

Chapter 4

Hierarchical Genetic Fuzzy System for Identification

Contents

4.1 Introduction / State of the Art 40

4.2 Overview 43

4.3 Hierarchical Genetic Fuzzy System 44

4.3.1 T-S Fuzzy Model 44

4.3.2 Least Squares Method 45

4.3.3 Hierarchical Structure 46

4.4 Initialization Methods 53

4.4.1 Fuzzy c-Means Clustering Algorithm 53

4.4.2 Fuzzy c-Regression Model Clustering Algorithm 56

4.4.3 Recursive Least Squares Method With Adaptive Directional Forgetting 58

4.5 Hierarchical Genetic Fuzzy System With Initialization Methods 60

4.6 Experimental Results 64

4.6.1 Wastewater Treatment System 65

4.6.2 Continuous-Stirred Tank Reactor 71

4.6.3 Real-World Setup of Two Coupled DC Motors 78

4.7 Conclusion 83

4.1 Introduction / State of the Art

Identification and model based control of industrial processes have been a focus of work in the context of many engineering problems and approaches that require accurate process models, such as soft sensor design or model predictive control design, respectively.

Data-driven soft sensors (DDSS) are inferential models that use on-line available sensor measures for on-line estimation of variables which cannot be automatically measured at all, or can only be measured at high cost, sporadically, or with large time delays (e.g. laboratory analysis). These models are based on measurements which are recorded and provided as historical data. The models themselves are empirical predictive models. They are valuable tools to many industrial applications such as refineries, pulp and paper mills, and wastewater treatment systems [Fortuna *et al.*, 2006]. Model predictive control (MPC), as explained in Section 2.6, is a popular control approach that is based on the use of a model of the process that predicts the future behavior of the system over a prediction horizon, and it is widely used in practice due to its high-quality control performance [Camacho and Bordons, 2007].

A weak point common to methodologies of both these two types, DDSS and MPC design, is their assumption of the knowledge of an accurate model of the process to be predicted/controlled. The majority of physical systems contain complex nonlinear relations, which are difficult to model with conventional techniques. The above assumption may present problems because many complex plants are difficult to be mathematically modelled based on physical laws, or have large uncertainties and strong nonlinearities. Several types approaches to modelling nonlinear plants can be considered to be used in DDSS or MPC. A suitable option, is the application of models based on fuzzy logic systems. This is theoretically supported by the fact that fuzzy logic systems are universal approximators [Wang and Mendel, 1992; Kosko, 1994]. Takagi-Sugeno (T-S) fuzzy models [Takagi and Sugeno, 1985], as explained in Section 2.2, are suitable to model a large class of nonlinear systems and have gained much popularity because of their rule consequent structure which is a mathematical function.

As mentioned in Subsection 3.3.9, several biologically inspired algorithms, such as for example ACO and PSO, could be used to design fuzzy systems. However GAs provide a robust search capacity with the ability to find near optimal solutions in complex and large search spaces, they are simple to implement, they have the possibility of using different types of solution encoding, and they are adaptive, which means that they have the ability to learn, accumulating relevant knowledge to solve the problem being considered. This motivates that GAs might be a useful soft computing technique for designing T-S fuzzy models.

In [Lo *et al.*, 2007], it is proposed a GAs approach to generate optimal fuzzy rules in a classification setting that continuously monitors system states for automatically detecting faults on a HVAC system. In [Chen and Lin, 2007], an approach is proposed for dynamic creation and evolving of a first order T-S fuzzy system model. In [Anh and Ahn, 2009], it is investigated a technique for modeling and identification of a new dynamic NARX fuzzy model by means of genetic algorithms. The paper proposed the use of a modified GA combined with the predictive capability of the NARX T-S fuzzy model for generating the dynamic NARX T-S fuzzy model. Zhao *et al.* [2010] proposed a methodology for automatically extracting T-S fuzzy models from data using particle swarm optimization. In [Yusof *et al.*, 2011], a technique for modeling nonlinear control processes using a fuzzy modeling approach based on the T-S fuzzy model with a combination of a GA and the recursive least squares method, is proposed.

In a prediction setting, the selection of the most adequate input variables and the respective time delays is crucial since the use of the correct variables with the correct delays can lead to better prediction accuracy because they can contain more information about the output than incorrect variables and/or variables with incorrect delays [Souza *et al.*, 2013]. Some studies have used techniques based on variance, such as as principal component analysis (PCA) for variable selection [Warne *et al.*, 2004]. These methods are designed for linear models, so they can not be the best choice for nonlinear modeling. The prediction methods proposed in [Tan *et al.*, 2004; Kim *et al.*, 2006; Lo *et al.*, 2007; Pettersson *et al.*, 2007; Delgado *et al.*, 2009], have the limitation of not being able to perform automatic selection of variables and delays: pre-selection is performed. Pre-selection may be performed completely from human knowledge (e.g. knowledge of the real model, such as in [Nie, 1995; Tan *et al.*, 2004; Kim *et al.*, 2006; Anh and Ahn, 2009; Tzeng, 2010; Zhao *et al.*, 2010;

Yusof *et al.*, 2011]) or using some auxiliary criteria that does not take advantage of taking into account the prediction model being learned, such as correlation coefficients, Kohonen maps and Lipschitz quotients [Delgado *et al.*, 2009], regularity criterion [Chen and Lin, 2007; Sugeno and Yasukawa, 1993], or analysis of "fuzzy curves" [Lin and Cunningham, 1995]. Some approaches have the limitation of not performing the selection of the time delays of input variables (e.g. [Lo *et al.*, 2007] and [Pettersson *et al.*, 2007]).

An approach using methods for both nonlinear variable selection and learning T-S fuzzy models was proposed in [Delgado *et al.*, 2001] and later in [Delgado *et al.*, 2009]. In [Delgado *et al.*, 2001], it is introduced a hierarchical evolutionary approach to optimize the parameters of T-S fuzzy systems, where the selection of the variables is performed completely from human knowledge, such as knowledge about the real model. The problems addressed are function approximation and pattern classification. As an evolution or improvement of [Delgado *et al.*, 2001], in the work [Delgado *et al.*, 2009], it was proposed to add to [Delgado *et al.*, 2001] a mechanism for pre-selection of the variables by an auxiliary criteria. The proposed method is addressed for soft sensors applications. It uses T-S fuzzy models learned from available input/output data by means of a coevolutionary GA and a neuro-based technique. The soft sensor design is carried out in two steps. First, the input variables of the fuzzy model are pre-selected from the variables of the dynamical process by means of correlation coefficients, Kohonen maps and Lipschitz quotients. Such selection procedure considers nonlinear relations among the input and output variables. Second, a hierarchical GA is used to identify the fuzzy model itself. The input variable selection approach proposed by [Delgado *et al.*, 2009] has some drawbacks. First, the selection of the number of neurons in the Kohonen maps is not automatically performed. Second, variables and delays selection is not jointly performed with the learning of the fuzzy model (pre-selection is performed), which precludes the global optimization of the prediction setting. Finally, the selection of input variables is not accompanied with the selection of the respective time delays. The later shortcoming can bring low-accuracy results because a variable with the correct delay can contain more information about the output than a variable with an incorrect delay.

4.2 Overview

This chapter proposes novel methodologies for identification of industrial systems. The proposed methods are automatic tools for T-S fuzzy model design, with the following main characteristics: (1) automatically performing the optimization of the variable and delay selection jointly with the learning and optimization of the system model without the need for any prior human knowledge, (2) the T-S fuzzy model structure is constructed just according to the data characteristics, and (3) it is optimized by means of GAs. This work has been inspired by [Delgado *et al.*, 2009]. However, it will jointly optimize a larger number of components of the prediction setting when compared to [Delgado *et al.*, 2009]. A hierarchical genetic algorithm (HGA) will be used to optimize a large set of parameters encoded at five different levels in order to design the T-S fuzzy model. When more complex design decisions involving a large number of parameters must be made, a global formulation of the problem representing all the parameters in just one optimization level can be inadequate. It is well known that computation, search, and optimization problems become more difficult to solve when the dimensionality of the state-space increases. In many cases, this problem is known as the curse of dimensionality. To tackle this issue, in this chapter the global problem is divided into various optimization levels, where the genetic evolution (optimization) of each level is performed separately, but is influenced by the current populations and optimization states of all the levels. HGAs make it possible to have different layers optimizing different parts of the T-S model, and facilitate the human interpretation of the optimization structure.

The main advancements of the proposed methodologies in comparison with [Delgado *et al.*, 2009] are the addition of a new hierarchical level responsible for the selection of variables and delays, and the application of initialization methods on the hierarchical evolutionary approach, in order to reduce the computational cost and increase the algorithm's performance. The hierarchical genetic fuzzy system is constituted by five levels. In the first level, the input variables and respective delays are chosen with the goal of attaining the highest possible prediction accuracy of the T-S fuzzy model. The selection of variables and delays is performed jointly with the learning of the fuzzy model, which increases the global optimization performance. The second level encodes the membership functions. The individual rules are defined at the third level. The population of the set of rules is defined at the fourth level,

and a population of fuzzy systems is treated at the fifth level. The least squares method is used to determine the parameters of the rule consequents. Levels two to five were based on [Delgado *et al.*, 2009].

Three methodologies are proposed in this chapter: a HGA without initialization method, named as HGA (Section 4.3, Algorithm 4.1), a HGA with an initialization method based on the fuzzy c-regression model (FCRM), named as HGA-FCRM (Section 4.5, Algorithm 4.5), and a HGA with an initialization method based on the fuzzy c-means (FCM) and using an adaptive methodology, named as AHGA-FCM (Section 4.5, Algorithm 4.6).

4.3 Hierarchical Genetic Fuzzy System

4.3.1 T-S Fuzzy Model

As mentioned in Section 3.2, any discrete-time nonlinear system with continuous constituent functions can be accurately approximated by a T-S fuzzy model of the form (3.12). In Section 4.3, the T-S fuzzy model that is employed, is defined by the following fuzzy rules:

$$\begin{aligned} R_i: \quad & \text{IF } x_1(k) \text{ is } A_1^i, \text{ and } \ldots \text{ and } x_n(k) \text{ is } A_n^i \\ & \text{THEN } y_i(\boldsymbol{\theta}_i, \mathbf{x}(k)) = \theta_{i0} + \theta_{i1}x_1(k) + \ldots + \theta_{in}x_n(k) \\ & \qquad\qquad +\theta_{i(n+1)}x_1^2(k) + \ldots + \theta_{i(2n)}x_n^2(k) \\ & \qquad\qquad +\theta_{i(2n+1)}x_1(k) \times \ldots \times x_n(k), \\ & i = 1, \ldots, N, \end{aligned} \tag{4.1}$$

where R_i $(i = 1, \ldots, N)$ represents the i-th fuzzy rule, N is the number of rules, and $x_1(k), \ldots, x_n(k)$ are the input variables of the T-S fuzzy system - they can be any variables chosen by the designer. A_j^i are linguistic terms characterized by fuzzy membership functions $\mu_{A_j^i}(x_j(k))$ which describe the local operating regions of the plant. Vector $\mathbf{x}(k) = [x_1(k), \ldots, x_n(k)]^T$ is both the system input, and an independent variable in the consequent functions $y_i(\cdot)$ $(i = 1, \ldots, N)$. $\boldsymbol{\theta}_i = \left[\theta_{i0}, \ldots, \theta_{i(2n+1)}\right]^T$ is a $Q = (2n + 2)$-dimensional vector that contains adjustable parameters of $y_i(\cdot)$ $(i = 1, \ldots, N)$.

Any discrete-time nonlinear system with continuous constituent functions can

also be accurately approximated by a T-S fuzzy model of the form (4.1). In fact, by making $\theta_{i0} = \theta_{i(n+1)} = \ldots = \theta_{i(2n)} = \theta_{i(2n+1)} = 0$, for $i = 1, \ldots, N$, equation (4.1) is reduced to (3.12). Parameters $\theta_{i0}, \theta_{i(n+1)}, \ldots, \theta_{i(2n)}, \theta_{i(2n+1)}$, for $i = 1, \ldots, N$, are used as additional degrees of freedom for tunning model(s) (4.1).

4.3.2 Least Squares Method

Let the l-th input sample be $\mathbf{x}(l) = [x_1(l), \ldots, x_n(l)]^T$. The l-th final output of the fuzzy model is inferred by a center weighted average defuzzification method as follows (see more details in [Wang, 1997], and in Section 3.2):

$$\begin{aligned} y[\mathbf{x}(l)] &= \sum_{i=1}^{N} \bar{\omega}_i[\mathbf{x}(l)] y_i(\boldsymbol{\theta}_i, \mathbf{x}(l)), && (4.2) \\ &= \sum_{i=1}^{N} \bar{\omega}_i[\mathbf{x}(l)] \mathbf{x}_e^T(l)\, \boldsymbol{\theta}_i, && (4.3) \\ &= \boldsymbol{\psi}[\mathbf{x}(l)]^T \boldsymbol{\Theta}, \end{aligned}$$

where, for $i = 1, \ldots, N$,

$$\begin{aligned} \bar{\omega}_i[\mathbf{x}(l)] &= \frac{\prod_{j=1}^{n} \mu_{A_j^i}[\mathbf{x}(l)]}{\sum_{p=1}^{N} \prod_{j=1}^{n} \mu_{A_j^p}[\mathbf{x}(l)]}, && (4.4) \\ \boldsymbol{\Theta} &= \left[\boldsymbol{\theta}_1^T, \ldots, \boldsymbol{\theta}_N^T\right]^T, && (4.5) \\ \mathbf{x}_e(l) &= \left[1, x_1(l), \ldots, x_n(l), x_1^2(l), \ldots, x_n^2(l), x_1(l) \times \ldots \times x_n(l)\right]^T, && (4.6) \\ \boldsymbol{\psi}[\mathbf{x}(l)] &= \left[(\bar{\omega}_1[\mathbf{x}(l)])\, \mathbf{x}_e^T(l), \ldots, (\bar{\omega}_N[\mathbf{x}(l)])\, \mathbf{x}_e^T(l)\right]^T. \end{aligned}$$

Let $y_d(\mathbf{x}(l))$ be the desired model of the system output, and consider L input patterns $\mathbf{x}(l)$, the corresponding desired output patterns $y_d(\mathbf{x}(l))$, $l = 1, \ldots, L$, and

$$\begin{aligned} \mathbf{y}_d &= [y_d(\mathbf{x}(1)), \ldots, y_d(\mathbf{x}(L))]^T, && (4.7) \\ \boldsymbol{\Psi} &= [\boldsymbol{\psi}(\mathbf{x}(1)), \ldots, \boldsymbol{\psi}(\mathbf{x}(L))]^T. && (4.8) \end{aligned}$$

Let $\boldsymbol{\Theta}^*$ be optimal values of $\boldsymbol{\Theta}$, a solution of $\boldsymbol{\Theta}^*$ can be computed using the pseudo-inverse least squares method, as follows [Ben-Israel and Greville, 2003]:

$$\mathbf{y}_d = \boldsymbol{\Psi}\boldsymbol{\Theta}, \quad (4.9)$$

$$\boldsymbol{\Theta}^* = \boldsymbol{\Psi}^+ \mathbf{y}_d. \tag{4.10}$$

If $\boldsymbol{\Psi}$ is full rank, then its pseudo inverse, $\boldsymbol{\Psi}^+$, can be computed in closed form as follows:

$$\boldsymbol{\Psi}^+ = \boldsymbol{\Psi}^T(\boldsymbol{\Psi}\boldsymbol{\Psi}^T)^{-1}, \quad \text{for } L \leqslant N(2n+2), \tag{4.11}$$

$$\boldsymbol{\Psi}^+ = \boldsymbol{\Psi}^{-1}, \quad \text{for } L = N(2n+2), \tag{4.12}$$

$$\boldsymbol{\Psi}^+ = (\boldsymbol{\Psi}^T\boldsymbol{\Psi})^{-1}\boldsymbol{\Psi}^T, \quad \text{for } L \geqslant N(2n+2). \tag{4.13}$$

Solution $\boldsymbol{\Theta}^*$ in (4.10) is the minimum-norm least squares solution of (4.9). A reasonable working assumption is that there are more data patterns than model parameters, i.e. $L \geqslant N(2n+2)$. Note that the full rank condition can always be made satisfied due to the fact that whenever $\boldsymbol{\Psi}$ is not full-rank, the linearly dependent columns of $\boldsymbol{\Psi}$ can be iteratively eliminated until $\boldsymbol{\Psi}$ has full rank [Delgado *et al.*, 2001].

4.3.3 Hierarchical Structure

The proposed coevolutionary model is illustrated in Figure 4.1. The approach is constituted by five hierarchical levels of populations, where each population represents different species:

- The first level consists of a population of sets of input variables and respective time delays;
- The second level represents the population of antecedent membership functions of the T-S fuzzy system;
- The population of individual rules is defined at the third level;
- The sets of fuzzy rules is represented at the fourth level;
- The fifth level represents the population with the indexes of the selected elements of the previous levels, where each element of the population represents a fuzzy system.

The detailed description of each level is given below:

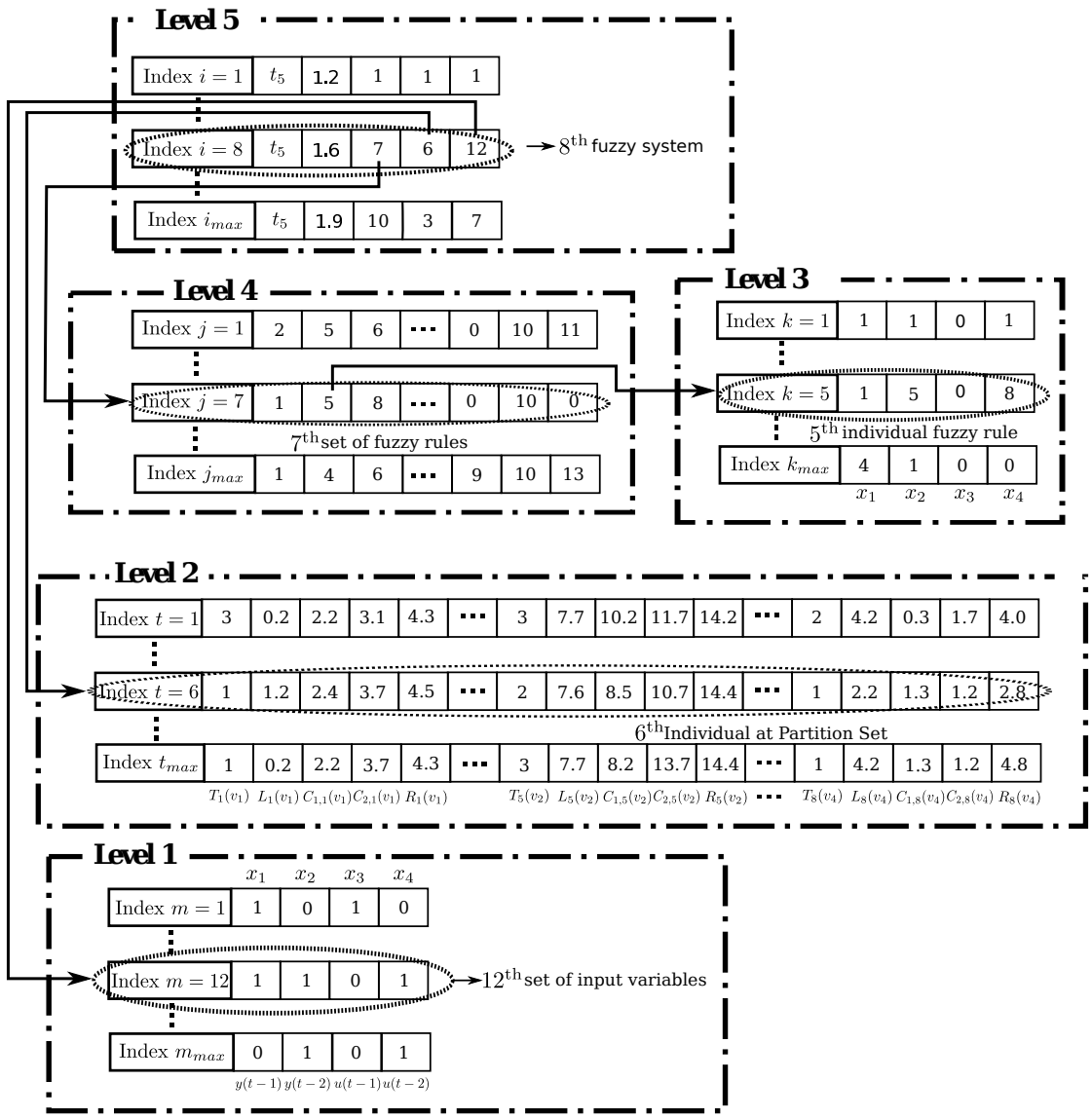


Figure 4.1: Encoding and hierarchical relations among the individuals of the different levels of the genetic hierarchy of the HGA approach.

***Level* 1:** it is formed by a set of input variables and respective delays that will be used in the T-S fuzzy model. The chromosome of Level 1 is represented by a binary encoding, where each allele (element of the chromosome that is located at a specific position) corresponds to each input variable and respective delay (see Figure 4.1). The length of the chromosome is given by the total number of pairs of system variables and respective delays that are considered as possible candidates to be used

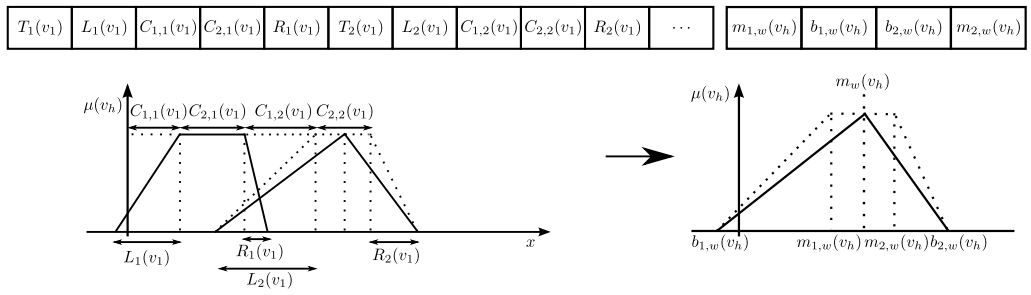


Figure 4.2: Membership functions encoding in Level 2 of the HGA approach.

as inputs of the T-S fuzzy model. In the example of Figure 4.1, Level 1 of the GA hierarchy is illustrated by describing the 12th ($m = 12$) set of input variables and respective delays. As can be seen, the pairs of variables and delays selected by the 12th chromosome of Level 1 correspond to $x_1 = y(t-1)$, $x_2 = y(t-2)$, and $x_3 = u(t-2)$.

***Level* 2:** contains the representation of all antecedent membership functions (including their corresponding parameters) defined in the universe of the variables involved. The chromosome is formed by the aggregations, one after another, of all partition sets associated with the input variables. A partition set of a variable is a collection of fuzzy sets associated to the variable. In each chromosome, associated to each variable there is exactly one partition set. For each variable the range of possible values that the variable may take is fixed by the designer. To reduce the computational time and the number of parameters to be tuned by the proposed HGA method, the designer may optionally choose to use a grouping methodology on Level 2. When using the grouping methodology, the set of variables considered on Level 2 is divided into several disjoint groups, where all variables belonging to the same group have the same range of possible values, and for each individual, all variables within each group are forced to share the same partition set. Level 2 is represented by integer and real encoding.

Figure 4.2 shows the detail of the chromosome of Level 2. Each chromosome is composed by a sequence of pentaplets[1] of alleles. In each pentaplet w, the first allele uses integer encoding to represent the type of a membership function w, $w = 1, \ldots, K_h$, where K_h is the total number of membership functions associ-

[1]A pentaplet is a set of five elements.

ated to variable v_h, $(h = 1, \ldots, n)$, where possible types of membership function are: trapezoidal $(T_w(v_h) = 1)$, triangular $(T_w(v_h) = 2)$, and Gaussian $(T_w(v_h) = 3)$. Variables v_h correspond to the input variables x_j, i.e. $v_h = x_h$, for $h = 1, \ldots, n$. Note that variables v_h $(h = 1, \ldots, n)$ are defined and used instead of x_j $(h = 1, \ldots, n)$, because in Chapter 5 v_h will represent the antecedent variables and also the consequent variable $(h = 1, \ldots, n + 1)$. Alleles 2-5 use real encoding to represent the parameters of the membership function. Considering the w-th membership function, for trapezoidal functions, alleles 2-5 are converted into absolute values, given by (see Figure 4.2):

$$m_{1,w}(v_h) = m_{2,w-1}(v_h) + C_{1,w}(v_h), \tag{4.14}$$

$$m_{2,w}(v_h) = m_{1,w}(v_h) + C_{2,w}(v_h), \tag{4.15}$$

$$b_{1,w}(v_h) = m_{1,w}(v_h) - L_w(v_h), \tag{4.16}$$

$$b_{2,w}(v_h) = m_{2,w}(v_h) + R_w(v_h), \tag{4.17}$$

where $m_{2,0}(v_h)$ is initialized by the first value of the universe of discourse of the respective variable v_h. For triangular membership functions, the center is found by the average between $m_{1,w}(v_h)$ and $m_{2,w}(v_h)$, i.e., $m_w(v_h) = (m_{1,w}(v_h) + m_{2,w}(v_h))/2$, (see Figure 4.2). For Gaussian membership functions, the central value is calculated in the same way as in the triangular case, and the dispersion is given by $\sigma_w(v_h) = ((L_w(v_h) + R_w(v_h))/2)/3$. In Figure 4.1, it is illustrated, as an example, the 6th $(t = 6)$ collection of individual partition sets on Level 2, that represents an example of the membership functions used in the 5th $(k = 5)$ individual fuzzy rule on Level 3. This partition set is illustrated by the 1st membership function of x_1, (v_1), the 5th membership function of x_2, (v_2), and the 8th membership function of x_4, (v_4), and these membership functions are of types trapezoidal $(T_1(v_1) = 1)$, triangular $(T_5(v_2) = 2)$, and trapezoidal $(T_8(v_4) = 1)$, respectively. Assuming $m_{2,7}(v_4) = 0.6$, and using (4.14)-(4.17), the parameters of the 8th membership function of x_4 (v_4) are:

$$m_{1,8}(v_4) = m_{2,7}(v_4) + C_{1,8}(v_4) = 0.6 + 1.3 = 1.9, \tag{4.18}$$

$$m_{2,8}(v_4) = m_{1,8}(v_4) + C_{2,8}(v_4) = 1.9 + 1.2 = 3.1, \tag{4.19}$$

$$b_{1,8}(v_4) = m_{1,8}(v_4) - L_8(v_4) = 1.9 - 2.2 = -0.3, \tag{4.20}$$

$$b_{2,8}(v_4) = m_{2,8}(v_4) + R_8(v_4) = 3.1 + 2.8 = 5.9, \tag{4.21}$$

as can be deduced from Figures 4.1 and 4.2.

***Level* 3:** it is formed by a population of individual rules. The length of the chromosome is determined by the maximum number of antecedent variables. The chromosome is represented by integer encoding where each allele is formed by the index that identifies the corresponding antecedent membership function (defined at Level 2). Null index values indicate the absence of membership function for the corresponding variable (i.e. the absence of the variable) in the rule. In the example of Figure 4.1, Level 3 of the GA hierarchy is illustrated by describing the 5th ($k = 5$) individual rule. As can be seen, in this rule x_1 is represented by its 1st membership function, x_2 is represented by its 5th membership function, and x_4 is represented by its 8th membership function.

***Level* 4:** it is formed by a set of fuzzy rules, where each allele contains the index of the corresponding individual rule that is being included in the set. Null values indicate that the corresponding allele does not contribute to the inclusion of any rule into the set of fuzzy rules. The chromosome is represented by integer encoding. The length of the chromosome is determined by the maximum number of fuzzy rules. In the example of Figure 4.1, taking into account the alleles that are filled with non-zero values, Level 4 of the GA hierarchy is illustrated by the 7th ($j = 7$) set of fuzzy rules that contains the 1st, 5th, 8th, and 10th individual rules, where these rules are described/represented in Level 3 of the hierarchy (but only the 1st and 5th rules are illustrated at the Level 3 of Figure 4.1).

***Level* 5:** it represents a fuzzy system. The chromosome is represented by integer and real encoding. The first allele represents the aggregation method used in the antecedent part of the rules. Here, only the $\mathbf{t}_5$ t-norm is used for aggregation, where

$$a \, \mathbf{t}_5 \, b = \frac{ab}{p_t + (1 - p_t)(a + b - ab)}, \tag{4.22}$$

and p_t is represented by allele 2, and its range is defined by $p_t \in [0, 10]$. In allele 1, other aggregation operators can be used (see more in [Wang, 1997]). Allele 3 chooses a j-th set of fuzzy rules specified at Level 4. Allele 4 selects a t-th partition set individual at Level 2, and Allele 5 chooses a m-th set of input variables and delays at Level 1.

Figure 4.1 presents an example of the encoding and the hierarchical relations. In this example, the 8th ($i = 8$) fuzzy system at Level 5 uses the $\mathbf{t}_5$-norm (4.22)

with the associated parameter $p_t = 1.6$, the 7th set of fuzzy rules at Level 4, the 6th partition set of Level 2, and the 12th set of selected input variables and delays at Level 1. The 7th set of fuzzy rules contains the 1st, 5th, 8th, and 10th individual rules, where the 5th individual rule, which is of the form (4.1), is composed of three input variables, x_1, x_2, and x_4 with membership functions 1, 5, and 8, respectively, i.e.:

$$\begin{aligned} R_5: \quad & \text{IF } x_1(k) \text{ is "1", and } x_2(k) \text{ is "5", and } x_4(k) \text{ is "8"} \\ & \text{THEN } y_5\left(\boldsymbol{\theta}_5, \mathbf{x}(k)\right), \end{aligned} \tag{4.23}$$

where $\boldsymbol{\theta}_5 = [\theta_{50}, \theta_{51}, \theta_{52}, \theta_{53}, \theta_{54}, \theta_{55}, \theta_{56}, \theta_{57}, \theta_{58}, \theta_{59}]^T$ is such that $\theta_{53} = \theta_{57} = 0$. Rule R_5 (4.23) could be equivalently written as follows:

$$\begin{aligned} R_5: \quad & \text{IF } x_1(k) \text{ is "1", and } x_2(k) \text{ is "5", and } x_4(k) \text{ is "8"} \\ & \text{THEN } \bar{y}_5\left(\mathbf{c}_5, [x_1, x_2, x_4]^T\right), \end{aligned} \tag{4.24}$$

where $\mathbf{c}_5 = [\theta_{50}, \theta_{51}, \theta_{52}, \theta_{54}, \theta_{55}, \theta_{56}, \theta_{58}, \theta_{59}]^T$, and $\bar{y}_5$ is defined such that $\bar{y}_5\left(\mathbf{c}_5, [x_1, x_2, x_4]^T\right) = y_5\left(\boldsymbol{\theta}_5, \mathbf{x}(k)\right)$. The linguistic terms "1", "5", and "8" of x_1, x_2, and x_4, respectively, are defined in the 6th chromosome at Level 2, and the input variables x_1, x_2, and x_4 are defined/selected at Level 1.

The main steps to learn/improve the T-S fuzzy model parameters are presented in Algorithm 4.1. The fitness functions of each individual for Levels 1 to 5 are defined in Algorithm 4.1. Each level of the genetic hierarchy is evolved separately as an independent genetic algorithm using its own population and its own different fitness function. However, since the (values of the) fitness functions of every levels depend on the populations of all the levels, then the evolution of each level also influences the evolution of all other levels. The flowchart of Figure 4.3 describes the operation of the method for identification of the T-S fuzzy model using GAs. Two possible stopping conditions, which are usually defined, consist on making the algorithm stop when a maximum number of generations is reached or when a desired fitness value is attained by the best individual in the population. In this chapter, for all proposed methods, the stopping condition is a pre-defined maximum number of generations.

Algorithm 4.1 Proposed HGA algorithm.

1. Set Generation $\leftarrow$ 1;
2. Initialize, randomly, the populations of all levels;
3. Compute the optimal parameters of the consequent of T-S fuzzy model for all individuals at Level 5 using the Least Squares method (Section 4.3.2);
4. Let i_{max}, j_{max}, k_{max}, t_{max}, and m_{max}, be the maximum number of chromosomes at Levels 5, 4, 3, 2, and 1, respectively. Compute the fitness of each individual i, j, k, t, and m, respectively of Level 5 to Level 1:
   (a) Level 5 (fuzzy system):
   $$J_5^i = 1/MSE, \tag{4.25}$$
   where MSE is the mean square error of the ith fuzzy system. The MSE is given by: $MSE = \frac{1}{L}\sum_{l=1}^{L}(y(l) - \hat{y}(l))^2$ where L is the number of data patterns, $\hat{y}(l)$ the predicted output pattern and $y(l)$ is the target output pattern;
   (b) Level 4 (rule base):
   $$J_4^j = max(J_5^{a_1}, \ldots, J_5^{a_p}), \tag{4.26}$$
   where $\{a_1, \ldots, a_p\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain rule-base j (set of fuzzy rules) on allele 3 of Level 5;
   (c) Level 3 (individual rule):
   $$J_3^k = max(J_4^{b_1}, \ldots, J_4^{b_q}), \tag{4.27}$$
   where $\{b_1, \ldots, b_q\} \subseteq \{1, \ldots, j_{max}\}$ is the subset of all chromosomes of Level 4 that contain individual rule k;
   (d) Level 2 (partition set):
   $$J_2^t = max(J_5^{c_1}, \ldots, J_5^{c_r}), \tag{4.28}$$
   where $\{c_1, \ldots, c_r\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain the collection of partition sets t on allele 4 of Level 5;
   (e) Level 1 (inputs and delays selection):
   $$J_1^m = max(J_5^{d_1}, \ldots, J_5^{d_s}), \tag{4.29}$$
   where $\{d_1, \ldots, d_s\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain the m-th selection of inputs and delays on allele 5 of Level 5.
5. Each level is evolved, considering it as a separate genetic algorithm. There may be only one common stopping condition that is used for all the levels. If the stopping condition does not hold, do:
   (a) Generation $\leftarrow$ Generation + 1;
   (b) For each level, apply the following evolutionary operators to form a new population: (1) selection, (2) crossover, and (3) mutation;
   (c) For each level, replace the current population with the new evolved population;
   (d) Return to Step 3.

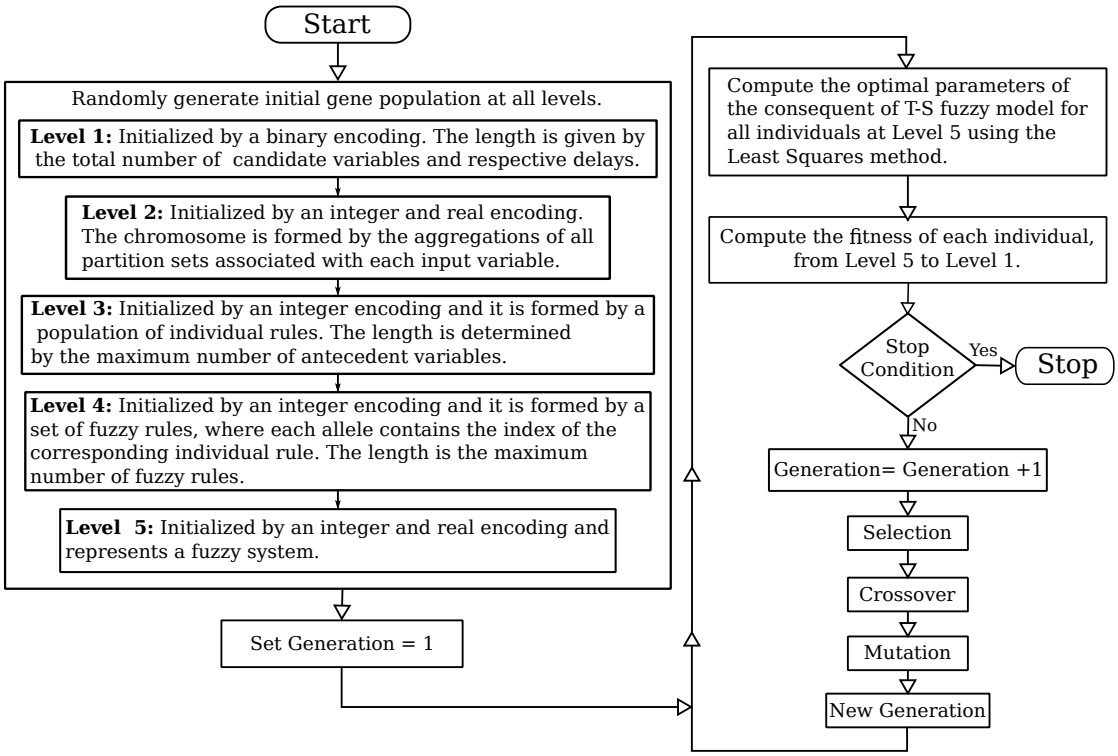


Figure 4.3: Flowchart of the HGA approach for learning the T-S fuzzy system.

4.4 Initialization Methods

GAs are usually initialized with random population elements. This sort of approach increases the tuning/search difficulty of the GA, since a set of totally random populations can lead to a very exhausting optimality search, requiring more iterations to attain convergence. Therefore, in order to reduce the computational cost and increase the algorithm's performance, two initialization methods are applied: the fuzzy c-regression model (FCRM) clustering algorithm proposed in [Li *et al.*, 2009], and the fuzzy c-means (FCM) clustering algorithm [Celikyilmaz and Trksen, 2009; Dovžan and Škrjanc, 2011].

4.4.1 Fuzzy c-Means Clustering Algorithm

This section presents, as an initialization method, the fuzzy c-means, FCM, clustering algorithm [Celikyilmaz and Trksen, 2009], [Dovžan and Škrjanc, 2011]. The

FCM algorithm can be used to perform the initialization of T-S fuzzy models of the forms (3.12) or (4.1), with Gaussian membership functions (3.15) in the antecedent parts of the rules. The objective of the fuzzy c-means (FCM) clustering algorithm is the partitioning of a data set $\mathbf{X}$ into a predefined number of clusters, N (typically denoted as c). In fuzzy clustering methods, the objects can belong to multiple clusters, with different degrees of membership.

Consider n samples which compose an input observation l (one sample of each input variable), which are grouped as an n-dimensional vector $\mathbf{x}_l = \mathbf{x}(l) = [x_1(l), \ldots, x_n(l)]^T$, where $\mathbf{x}(l) \in \mathbb{R}^n$. Let a set of L observations of the input variables be denoted by

$$\mathbf{X} = \begin{bmatrix} x_1(1) & x_2(1) & \ldots & x_n(1) \\ x_1(2) & x_2(2) & \ldots & x_n(2) \\ \vdots & \vdots & \vdots & \vdots \\ x_1(L) & x_2(L) & \ldots & x_n(L) \end{bmatrix}. \tag{4.30}$$

Note that one complete observation $(\mathbf{x}_l, y(l)) = (\mathbf{x}(l), y(l))$ is composed of an input observation $\mathbf{x}(l)$, and an output observation $y(l)$. The fuzzy partition of the set $\mathbf{X}$ into N clusters, is a family of fuzzy subsets $\{A^i \mid 1 \leqslant i \leqslant N\}$. The membership functions of these fuzzy subsets are defined as $\mu_i(l) = \mu_{A^i}(\mathbf{x}_l)$, and form the fuzzy partition matrix $\mathbf{U} = [u_{il}] = [\mu_i(l)] \in \mathbb{R}^{N \times L}$. The i-th row of matrix $\mathbf{U}$ contains the values of the membership function of the i-th fuzzy subset A^i for all the observations belonging to the data matrix $\mathbf{X}$. The partition matrix has to meet the following conditions [Dovžan and Škrjanc, 2011]: The membership degrees are real numbers in the interval $\mu_i(l) \in [0, 1], 1 \leqslant l \leqslant L$; the total membership of each sample in all the clusters must be equal to one $\sum_{i=1}^{N} \mu_i(l) = 1$; and none of the fuzzy clusters is empty, neither does any contain all the data $0 < \sum_{l=1}^{L} \mu_i(l) < L, 1 \leqslant i \leqslant N$.

FCM clustering tries to minimize the following objective function, which has a pre-defined number of clusters, N, and includes a fuzziness parameter, η:

$$J(\mathbf{X}, \mathbf{U}, \mathbf{V}) = \sum_{i=1}^{N} \sum_{l=1}^{L} (\mu_i(l))^{\eta} d_i(l)^2(\mathbf{x}(l), \mathbf{v}_i), \tag{4.31}$$

where $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_N]^T \in \mathbb{R}^{N \times n}$ is a matrix of cluster centroid vectors $\mathbf{v}_i = [v_{i1}, \ldots, v_{in}]^T$, $d_i(l)$ is the Euclidean distance (l^2-norm) between the observation $\mathbf{x}(l)$

and the cluster centroid $\mathbf{v}_i$, and the overlapping factor or the fuzziness parameter that influences the fuzziness of the resulting partition is denoted as η. The partition can range from a hard partition ($\eta = 1$) to a completely fuzzy partition ($\eta \to \infty$).

In order to find the fuzzy clusters in the data set $\mathbf{X}$, equation (4.31) must be minimized. If the derivative of the objective function is taken with respect to the cluster centers $\mathbf{V}$ and to the membership values $\mathbf{U}$, then optimum membership values are calculated as follows [Dovžan and Škrjanc, 2011]:

$$\mu_i(l) = \left(d_i(l)^2 \sum_{q=1}^{N} \left(d_q^2(l) \right)^{1/(\eta-1)} \right)^{-1}, \tag{4.32}$$

where

$$d_i(l)^2 = \left(\mathbf{x}(l) - \mathbf{v}_i \right)^T \left(\mathbf{x}(l) - \mathbf{v}_i \right), \tag{4.33}$$

$$\mathbf{v}_i = [v_{i1}, \ldots, v_{in}]^T = \frac{\sum_{l=1}^{L} \mu_i^{\eta}(l)\mathbf{x}(l)}{\sum_{l=1}^{L} \mu_i^{\eta}(l)}, \tag{4.34}$$

$$v_{ij} = \frac{\sum_{l=1}^{L} \mu_i^{\eta}(l) x_j(l)}{\sum_{l=1}^{L} \mu_i^{\eta}(l)}, \quad j = 1, \ldots, n. \tag{4.35}$$

To finalize the identification of the premise parameters, vector $\boldsymbol{\sigma}_i = [\sigma_{i1}, \ldots, \sigma_{in}]^T$, $i = 1, \ldots, N$, can be easily calculated using $\mathbf{U} = [\mu_i(l)]$, as follows:

$$\sigma_{ij} = \sqrt{\frac{2 \sum_{l=1}^{L} \mu_i(l)(x_j(l) - v_{ij})^2}{\sum_{l=1}^{L} \mu_i(l)}}, \quad j = 1, \ldots, n. \tag{4.36}$$

The components v_{ij}, and σ_{ij}, of $\mathbf{v}_i$, and $\boldsymbol{\sigma}_i$, are centers and widths of antecedent Gaussian membership functions (3.15), respectively.

To construct a T-S fuzzy system of the form (3.12) or (4.1) the antecedent parameters ($\mathbf{v}_i$ and $\boldsymbol{\sigma}_i$), and the consequent parameters ($\boldsymbol{\theta}_i$) are necessary. The antecedent parameters are given by the fuzzy c-means algorithm, (4.35), (4.36), and the consequent parameters can be given by a Least Squares Method (Subsection 4.3.2) or even by a Recursive Least Squares Method (Subsection 4.4.3). In this chapter, when the FCM algorithm is used to construct the antecedent parameters of a T-S

Algorithm 4.2 FCM algorithm.

1. Obtain a data set $\mathbf{X}$ (4.30) and define the number of clusters N, the degree of fuzziness η, and the stopping conditions $\epsilon > 0$, and Max. Initialize the partition matrix $\mathbf{U}$, randomly;
2. Find initial cluster centers using (4.35) with the membership values of the initial partition matrix $\mathbf{U}$; Let the initial vectors of cluster centers be denoted by $\mathbf{v}_i^{(0)}$;
3. For iteration $t = 1, \ldots, Max$:
   (a) Using (4.32), calculate the membership value at iteration t, $\mu_i^{(t)}(l)$, of each input data object $\mathbf{x}(l)$ in each cluster i, using the cluster center vector from iteration $(t-1)$, denoted as $\mathbf{v}_i^{(t-1)}$; Let $\mathbf{U} \leftarrow [\mu_i^{(t)}(l)]$;
   (b) Calculate the cluster center of each cluster i at iteration t, $\mathbf{v}_i^{(t)}$, by (4.35), using the membership values (4.32) at iteration t, $\mu_i^{(t)}(l)$; Let $\mathbf{V} \leftarrow [\mathbf{v}_1^{(t)}, \ldots, \mathbf{v}_N^{(t)}]$;
   (c) Exit the 'For' cycle if a termination condition is satisfied, e.g. $|\mathbf{v}_i^{(t)} - \mathbf{v}_i^{(t-1)}| \leqslant \epsilon$, and save the last iteration of the matrices $\mathbf{U}$ and $\mathbf{V}$. Otherwise let $t \leftarrow t+1$ and go to Step 3a;
4. Compute the parameters $\boldsymbol{\sigma}_i$ using (4.36);

fuzzy system, then the consequent parameters are obtained by the Recursive Least Squares Method described in Subsection 4.4.3. The FCM algorithm is presented in Algorithm 4.2.

4.4.2 Fuzzy c-Regression Model Clustering Algorithm

This section presents, as an initialization method, the fuzzy c-regression model (FCRM) clustering algorithm proposed in [Li *et al.*, 2009]. The FCRM algorithm can be used to perform the initialization of T-S fuzzy models of the form (3.12), with Gaussian membership functions (3.15) in the antecedent parts of the rules. As in the FCM algorithm (Section 4.4.1), the objective of the FCRM algorithm is to partition a data set $\mathbf{X}$ (4.30) into a predefined number of clusters N (typically named as c). Also, an observation l is assumed to be composed by n samples (one sample of each input variable) which are grouped into an n-dimensional vector $\mathbf{x}(l) = [x_1(l), \ldots, x_n(l)]^T \in \mathbb{R}^n$. A set of L observations is then denoted as in (4.30).

Assume that the L data pairs $(\mathbf{x}(l), y(l))$ $(l = 1, \ldots, L)$ are grouped into N

clusters. The data samples in i-th cluster are approximated with a linear regression model, which is actually a hyper-plane function, defined by

$$\begin{aligned} \hat{y}^i(l) &= f^i(\mathbf{x}(l), \boldsymbol{\theta}_i) = \theta_{i1}x_1(l) + \ldots + \theta_{in}x_n(l), \\ &= \mathbf{x}^T(l)\boldsymbol{\theta}_i,\ i = 1, \ldots, N, \end{aligned} \tag{4.37}$$

where $\boldsymbol{\theta}_i = [\theta_{i1} \ldots, \theta_{in}]^T$. The distance from output $y(l)$ to the i-th regression model output, $\hat{y}^i(l)$, with parameter $\boldsymbol{\theta}_i$, is defined as follows:

$$d_i(l, \boldsymbol{\theta}_i) = \left| y(l) - \hat{y}^i(l) \right|. \tag{4.38}$$

The objective function of FCRM is defined as

$$J(\mathbf{U}, \boldsymbol{\Theta}) = \sum_{l=1}^{L}\sum_{i=1}^{N}(\mu_i(l))^{\eta}(d_i(l, \boldsymbol{\theta}_i))^2, \tag{4.39}$$

where $\boldsymbol{\Theta} = [\boldsymbol{\theta}_1^T, \ldots, \boldsymbol{\theta}_N^T]^T$, $\eta \in (1, \infty)$ is the fuzzy weighting exponent, $\mathbf{U} = [\mu_i(l)] \in \mathbb{R}^{N \times L}$, $\mu_i(l) \in [0, 1]$ is the fuzzy membership degree the of l-th data pair to the i-th cluster that is given by [Li *et al.*, 2009]:

$$\mu_i(l) = \frac{1}{\sum_{q=1}^{N}[(d_i(l, \boldsymbol{\theta}_i))/(d_q(l, \boldsymbol{\theta}_q))]^{2/(\eta-1)}}. \tag{4.40}$$

To minimize (4.39), $\frac{\partial J(\mathbf{U}, \boldsymbol{\Theta})}{\partial \theta_{ij}} = 0$ is solved [Li *et al.*, 2009], yielding

$$\theta_{ij} = \frac{\sum_{l=1}^{L}(\mu_i(l))^{\eta}\left(y(l) - \sum_{t \neq j}\theta_{it}x_t(l)\right)x_j(l)}{\sum_{l=1}^{L}(\mu_i(l))^{\eta}x_j(l)^2},\ i = 1, \ldots, N,\ j = 1, \ldots, n. \tag{4.41}$$

Finally, the identification of the antecedent parameters (of the Gaussian membership functions) v_{ij} and σ_{ij} are given by (4.35) and (4.36), respectively.

To construct a T-S fuzzy system the antecedent parameters (v_{ij} and σ_{ij}) and the consequent parameters (θ_{ij}) are necessary. The antecedent parameters are given by FCRM presented in this Section 4.4.2 and the consequent parameters can be given by a Least Squares Method (Subsection 4.3.2) or even by a Recursive Least Squares

Algorithm 4.3 FCRM algorithm.

1. Obtain a data set $\mathbf{X}$ (4.30) and define the number of clusters N, the degree of fuzziness η, and the stopping conditions $\epsilon > 0$, and Max. Initialize vector $\boldsymbol{\Theta}$.
2. For iteration $t = 1, \ldots, Max$:
   (a) Calculate the membership values (4.40) at iteration t, $\mu_i^{(t)}(l)$, using $d_i(l, \boldsymbol{\theta}_i)$ (4.38). Recalculate $\theta_{ij}^{(t)}$ using (4.41); Let $\mathbf{U} \leftarrow \mathbf{U}^{(t)} \leftarrow [\mu_i^{(t)}(l)]$;
   (b) If $t > 1$, then exit the 'For' cycle if an additional termination condition is satisfied, e.g. $\|\mathbf{U}^{(t)} - \mathbf{U}^{(t-1)}\| \leqslant \epsilon$, and save the results of the last iteration of matrix $\mathbf{U}$. Otherwise let $t \leftarrow t + 1$ and go to Step 2a;
3. Compute the parameters v_{ij} using (4.35) and the parameters σ_{ij} using (4.36);

Method (Subsection 4.4.3). In this chapter, when the FCRM algorithm is used to construct the antecedent parameters of a T-S fuzzy system, then the consequent parameters are obtained by the Least Squares Method described in Subsection 4.3.2. The FCRM algorithm is presented in Algorithm 4.3.

4.4.3 Recursive Least Squares Method With Adaptive Directional Forgetting

In off-line training algorithms the T-S fuzzy model can be obtained from input-output data collected from a plant. However, such collected data set(s) can be limited, the obtained T-S fuzzy models may not provide adequate accuracy, the system can be nonlinear and/or time-varying, and can have varying operating points and varying parameters of the model. Adaptive methodologies are good possibilities to be applied to solve these problems.

Thus, for identification approaches, after defining/learning the antecedent parameters, the consequent parameters can be updated by an adaptive methodology. The classic RLS algorithm uses a constant forgetting factor and its performance in terms of convergence rate, tracking, misadjustment, and stability depends on the forgetting factor. Also, when excitation of the system is poor, the classic RLS can lead to the covariance wind-up problem. For these reasons, the adaptive methodology used in this thesis to adapt the model parameters is based on the recursive

least squares (RLS) method, and on an adaptive directional forgetting (RLS-ADF) approach of [Kulhavý, 1987; Bobál *et al.*, 2005], here adapted for the T-S fuzzy model. In particular, the RLS-ADF can be applied with the T-S fuzzy models of equations (3.12) or (4.1). Then, the resulting adaptive model approach is the one that is integrated within indirect adaptive control architectures when such type of architecture is used in this thesis.

At each iteration, l, the vector of model parameters ($\boldsymbol{\theta}_i$), is estimated by being updated using

$$\boldsymbol{\theta}_i(l) = \boldsymbol{\theta}_i(l-1) + \frac{\mathbf{C}_i(l-1)\boldsymbol{\psi}_i^T(l)}{1+\xi_i}\left[y_i(l) - \boldsymbol{\psi}_i(l)\boldsymbol{\theta}_i(l-1)\right], \tag{4.42}$$

where $\boldsymbol{\psi}_i(l) = \left(\bar{\omega}_i[\mathbf{x}(l)]\right)\mathbf{x}^T(l)$, $\xi_i = \boldsymbol{\psi}_i(l)\mathbf{C}_i(l-1)\boldsymbol{\psi}_i^T(l)$, $\mathbf{C}_i(l)$ is the covariance matrix of fuzzy rule i, and $y_i(l) = \left(\bar{\omega}_i[\mathbf{x}(l)]\right)y(l)$.

The covariance matrix is also updated at each iteration, l, using

$$\mathbf{C}_i(l) = \mathbf{C}_i(l-1) - \frac{\mathbf{C}_i(l-1)\boldsymbol{\psi}_i^T(l)\boldsymbol{\psi}_i(l)\mathbf{C}_i(l-1)}{\varepsilon_i^{-1}+\xi_i}, \tag{4.43}$$

where $\varepsilon_i = \varphi_i(l-1) - \frac{1-\varphi_i(l-1)}{\xi_i}$, and $\varphi_i(l-1)$ is the forgetting factor at iteration $(l-1)$ of the fuzzy rule i. The initial values, $\mathbf{C}_i(0)$, of $\mathbf{C}_i(l)$ should be set to a diagonal matrix where the main diagonal entries are suitably large numbers, as for example 10^5 for all main diagonal entries.

The adaption of the forgetting factor is performed using [Kulhavý, 1987; Bobál *et al.*, 2005]

$$\varphi_i(l) = \frac{1}{1+(1+\rho)\left\{ln(1+\xi_i) + \left[\frac{(\nu_i(l)+1)\gamma_i}{1+\xi_i+\gamma_i} - 1\right]\frac{\xi_i}{1+\xi_i}\right\}}, \tag{4.44}$$

where $\nu_i(l) = \varphi_i(l-1)(\nu_i(l-1)+1)$, $\gamma_i = \frac{(y_i(l)-\boldsymbol{\psi}_i(l)\boldsymbol{\theta}_i(l-1))^2}{\tau_i(l)}$, $\tau_i(l) = \varphi_i(l-1)\left[\tau_i(l-1) + \frac{(y_i(l)-\boldsymbol{\psi}_i(l)\boldsymbol{\theta}_i(l-1))^2}{1+\xi_i}\right]$, and ρ is positive constant. The initial values of $\varphi_i(0)$, $\tau_i(0)$ and $\nu_i(0)$ should be set between zero and one.

4.5 Hierarchical Genetic Fuzzy System With Initialization Methods

This section presents two additional hierarchical genetic fuzzy system approaches which in part can be seen as variants of the HGA presented in Section 4.3, Algorithm 4.1. Both approaches include initialization of the antecedent part of the fuzzy system to be learned by the HGA, and use the T-S fuzzy model defined in equation (3.12), and not the T-S fuzzy model defined in (4.1). The first approach, named as HGA-FCRM, integrates the HGA with the FCRM initialization method, and is presented in Algorithm 4.5. The second approach, named as AHGA-FCM, integrates the HGA with the FCM initialization method, and uses an adaptive methodology to update the consequent parameters of the T-S fuzzy system. The AHGA-FCM is presented in Algorithm 4.6.

The hierarchical architecture for both the HGA-FCRM and the AHGA-FCM approaches is illustrated in Figure 4.4. The description of the Levels 1, 3, and 4 is the same as the description done for the corresponding levels in Subsection 4.3.3. The modifications are introduced at Levels 2 and 5. On Level 2, only Gaussian membership functions are used and a different representation of the antecedent membership functions is done, and on Level 5 a different t-norm is used. The detailed description of each Level is given below.

***Level* 1:** it is the same as the Level 1 defined in Subsection 4.3.3.

***Level* 2:** contains the representation of all antecedent membership functions. The chromosome is formed by the aggregations, one after another, of all partition sets associated with the input variables, where the partition set of a variable is a collection of fuzzy sets associated to the variable (this, similarly to Subsection 4.3.3). In each chromosome, associated to each variable there is exactly one partition set. All alleles use real encoding to represent the parameters of the Gaussian membership function v_{ij} (4.35) and σ_{ij} (4.36). In the example of Figure 4.4, on Level 2, each chromosome is composed by a sequence of pairs of alleles. In each pair w, the first allele, $\sigma_w(v_h)$, represents the dispersion of a Gaussian membership function w of a variable v_h ($h = 1, \ldots, n$), and the second allele, $v_w(v_h)$, represents the center of the respective Gaussian membership function w, $w = 1, \ldots, K_h$, where K_h is the total number of membership functions of variable v_h. In Figure 4.4, it is illustrated, as an example, the 6th ($t = 6$) individual partition set on Level 2, that represents an

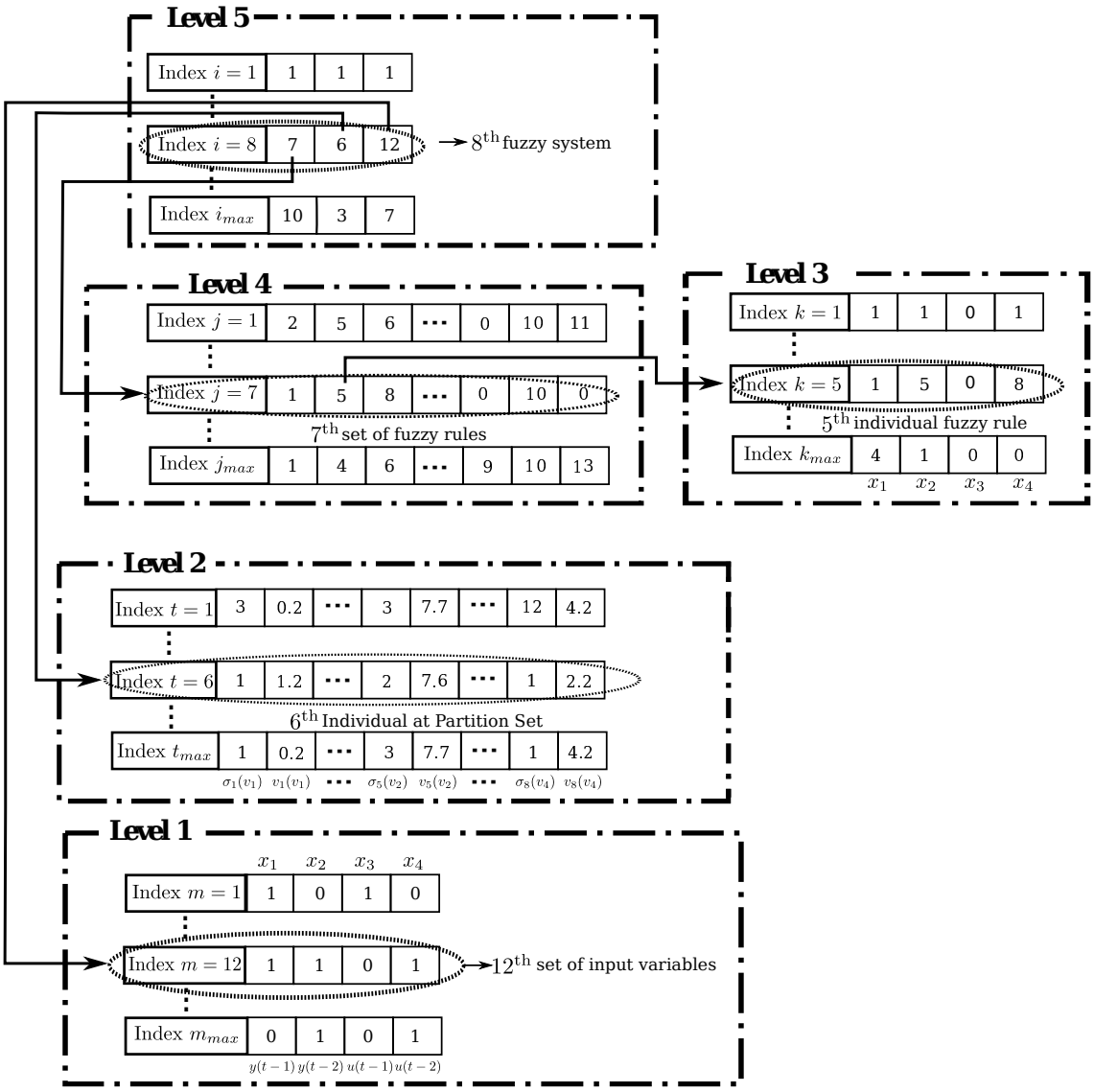


Figure 4.4: Encoding and hierarchical relations among the individuals of the different levels of the genetic hierarchy for the HGA-FCRM and AHGA-FCM approaches.

example of the membership functions used in the 5th ($k = 5$) individual fuzzy rule on Level 3. This partition set is illustrated by the 1st membership function of x_1, (v_1), the 5th membership function of x_2, (v_2), and the 8th membership function of x_4, (v_4), and these membership functions have widths $\sigma_1(v_1) = 1$, $\sigma_5(v_2) = 2$, and $\sigma_8(v_4) = 1$, and centers $v_1(v_1) = 1.2$, $v_5(v_2) = 7.6$, and $v_8(v_4) = 2.2$, respectively.

To reduce the computational time and the number of parameters to be tuned, the

Algorithm 4.4 Proposed initialization of the HGA methodology for the HGA-FCRM and HGA-FCM algorithms.

1. Compute the antecedent membership functions parameters σ_{ij} and v_{ij} using Algorithm 4.2 for the FCM case or Algorithm 4.3 for the FCRM case;
2. Initialize the populations of all levels:
   (a) Level 1: on the first individual, initialize with ones the alleles that represent the input variables defined in the data set used in Algorithm 4.2 for FCM or in Algorithm 4.3 for FCRM. Initialize with zeros the remaining alleles of the first individual;
   (b) Level 2: initialize all individuals with the antecedent membership functions computed in Step 1;
   (c) Level 3: initialize the first N individuals with the antecedent part of the fuzzy rules which is learned by the FCM algorithm or the FCRM algorithm. The way this is done is by initializing the first individual with ones, the second individual with twos, until the N-th individual with N's;
   (d) Level 4: initialize the first individual with indexes of the first N individuals of Level 3; Initialize with zeros the remaining alleles of the first individual;
   (e) Level 5: initialize the first individual of Level 5 with ones;
   (f) Randomly initialize the remaining individuals of Levels 1, 3, 4, and 5;

variables grouping methodology, proposed on Level 2 of the HGA (Section 4.3.3), can be optionally employed on Level 2 on the HGA-FCRM and AHGA-FCM methodologies.

***Level* 3:** it is the same as the Level 3 defined in Subsection 4.3.3.

***Level* 4:** it is the same as the Level 4 defined in Subsection 4.3.3.

***Level* 5:** each individual represents a fuzzy system. The chromosome is represented by integer encoding. The first allele chooses a j-th set of fuzzy rules specified at Level 4. Allele 2 selects a t-th partition set individual at Level 2, and allele 3 chooses the m-th set of input variables and delays at Level 1. In the example of Figure 4.4, the 8th fuzzy system at Level 5 uses the 7th set of fuzzy rules at Level 4, the 6th partition set of Level 2, and the 12th set of selected input variables and delays at Level 1.

The proposed method for initialization of the hierarchical methodology is presented in Algorithm 4.4. The main steps of the proposed HGA-FCRM approach are

Algorithm 4.5 Proposed HGA-FCRM algorithm.

1. Set Generation ← 1;
2. Initialize the populations of all levels with Algorithm 4.4 for the case it is used in conjunction with FCRM;
3. Compute the consequent parameters of the T-S fuzzy model for all individuals at Level 5 using the Least Squares method (Section 4.3.2);
4. Compute the fitness of each individual, from Level 5 to Level 1, as done by the Steps 4a-4e on Algorithm 4.1.
5. If the stopping condition does not hold, then do for each level:
   (a) Generation ← Generation + 1;
   (b) Apply the evolutionary operators to form a new population: selection, crossover, and mutation;
   (c) Replace the current population with the new evolved population;
   (d) Return to Step 3.

presented in Algorithm 4.5. The main steps of the proposed AHGA-FCM approach are presented in Algorithm 4.6. As in Subsection 4.3.3, each level of the genetic hierarchy is evolved separately as an independent genetic algorithm using its own population and its own different fitness function. However, since the (values of the) fitness functions of every levels depend on the populations of all the levels, then evolution of each level also influences the evolution of all other levels.

The AHGA-FCM has the advantage of integrating an adaptive methodology to update the consequent parameters instead of using the HGA-FCRM approach where the consequent parameters are obtained in an offline way. As can be seen in Step 4 of Algorithm 4.6, AHGA-FCM integrates, for all data samples $(\mathbf{x}(l), y(l))$ $(l = 1, \ldots, L)$, the application of the adaptation methodology defined in Subsection 4.4.3 which is based on the recursive least squares with adaptive directional forgetting (RLS-ADF) approach of [Kulhavý, 1987; Bobál *et al.*, 2005].

Algorithm 4.6 Proposed AHGA-FCM algorithm.

1. Set Generation $\leftarrow 1$;
2. Initialize the populations of all levels with Algorithm 4.4 for the case it is used in conjunction with FCM;
3. Design the identification parameters (ρ, φ_i, τ_i, ν_i, and $\mathbf{C}_i$, for all $1 \leqslant i \leqslant N$) of the recursive least squares method with adaptive directional forgetting (Subsection 4.4.3).
4. Compute the consequent parameters of the T-S fuzzy model for all individuals at Level 5 by initializing the components of $\boldsymbol{\theta}_i$ to small values (e.g. 10^{-10}), and then using the recursive least squares method with adaptive directional forgetting (Subsection 4.4.3) with the parameters designed in Step 3, using recursion (4.42) for $l = 1, \ldots, L$;
5. Compute the fitness of each individual, from Level 5 to Level 1, as done by the Steps 4a-4e on Algorithm 4.1.
6. If the stopping condition does not hold, then do for each level:
   (a) Generation $\leftarrow$ Generation + 1;
   (b) Apply the evolutionary operators to form a new population: selection, crossover, and mutation;
   (c) Replace the current population with the new evolved population;
   (d) Return to Step 4.

4.6 Experimental Results

This section presents simulation and real-world experiments and results to demonstrate the feasibility, performance and effectiveness of the proposed T-S fuzzy model design methodologies. The identification of a model for the estimation of the fluoride concentration in the effluent of a real-world wastewater treatment system, the product concentration on a simulated CSTR plant, and the velocity of a real-world experimental setup composed of two coupled DC motors are studied. The identification results of the CSTR and DC motors processes will be used on Chapter 6 to define the prediction model for fuzzy predictive control of these processes.

In all the three experiments, the results were obtained by considering that the crossover and mutation probabilities are 80% and 10%, respectively, the number of

generations is $Gen_{max} = 1500$, and the numbers of chromosomes for each level of the architecture are: $i_{max} = 30$, $j_{max} = 30$, $k_{max} = 30$, $t_{max} = 30$, and $m_{max} = 50$. These parameters were tuned by means of experimentation. In all data sets, the first half of the data set was used for training and the remaining data was used for test. The proposed methodologies were implemented in the Matlab Software with the main functions being implemented in the C programming language to reduce computational time.

The identification performance is, also, quantitatively compared with two non-adaptive approaches: multilayer perceptron (MLP) and extreme learning machine (ELM) [Huang *et al.*, 2006]; and two adaptive approaches: the Recursive Partial Least Squares (RPLS) method [Dayal and MacGregor, 1997], and the Incremental Local Learning Soft Sensing Algorithm (ILLSA) [Kadlec and Gabrys, 2011].

4.6.1 Wastewater Treatment System

In this subsection the methodologies proposed in this chapter are applied to the development of a data-driven soft sensor (DDSS).

DDSS are inferential models that use on-line available sensor measures for on-line estimation of variables which cannot be automatically measured at all, or can only be measured at high cost, sporadically, or with large time delays (e.g. laboratory analysis). These models are based on measurements which are recorded and provided as historical data. The models themselves are empirical predictive models. They are valuable tools to many industrial applications such as refineries, pulp and paper mills, wastewater treatment systems, just to give a few examples [Fortuna *et al.*, 2006].

The development of DDSS can be divided into four main stages: (I) Data collection, and selection of historical data; (II) Data pre-processing; (III) Model selection, training, and validation; (IV) Soft sensor maintenance. The model selection, training, and validation phase is one of most important in soft sensors development, requiring the correct learning of the model, so that it can correctly reproduce the target variable. DDSS are built based on empirical observations of the process.

In this subsection the nonlinear system identification application problem that is analyzed is the estimation of the fluoride concentration in the effluent of a real-world wastewater treatment plant (WWTP). The data set of plant variables that is

Table 4.1: Variables of the wastewater treatment plant data set.

Variables	Description
u_1	Amount of chlorine in the influent, [ppm];
u_2	Amount of chlorine in the effluent, [ppm];
u_3	Turbidity in the raw water, [NTU];
u_4	Turbidity in the influent, [NTU];
u_5	Turbidity in the effluent, [NTU];
u_6	Ph in the raw water;
u_7	Ph in the influent;
u_8	Ph in the effluent;
u_9	Color in the raw water, [TCU];
u_{10}	Color in the influent, [TCU];
u_{11}	Color in the effluent, [TCU];
y	Fluoride in the effluent, [mg.l^{-1}].

available for learning consists of 11 input variables, $u_1, \ldots, u_{11}$, and one target output variable to be estimated, y, and contains 1002 samples. The variables correspond to physical values, such as pH, turbidity, color of the water and others. The input variables are measured on-line by plant sensors, and the output variable in the data set is measured by laboratory analysis. The sampling interval is 2 [hours]. The plant variables are described in Table 4.1. Figure 4.5 shows the plots of the variables listed in Table 4.1.

To construct the data set, the first three delayed versions of each variable were chosen as candidates for inputs of the T-S fuzzy model. Specifically, the following combinations of process variables and delays are used as the candidates for inputs of the T-S fuzzy model to predict $y(t)$: $[u_1(t-1), u_1(t-2), u_1(t-3), \ldots, u_{11}(t-1), u_{11}(t-2), u_{11}(t-3)]$.

In both of the HGA-FCRM and AHGA-FCM methodologies, the number of clusters and the degree of fuzziness were chosen as $N = 13$, and $\eta = 2$, respectively.

Figure 4.6 shows the prediction results for the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies and the desired (real) values of the target variable to be estimated, for the WWTP experiment. Numerical results comparing the performance of the proposed methodologies are presented in Table 4.2. The membership functions obtained by the proposed HGA methods and respective initialization methods are shown in Figures 4.7-4.9. The Levels 2 of the algorithms

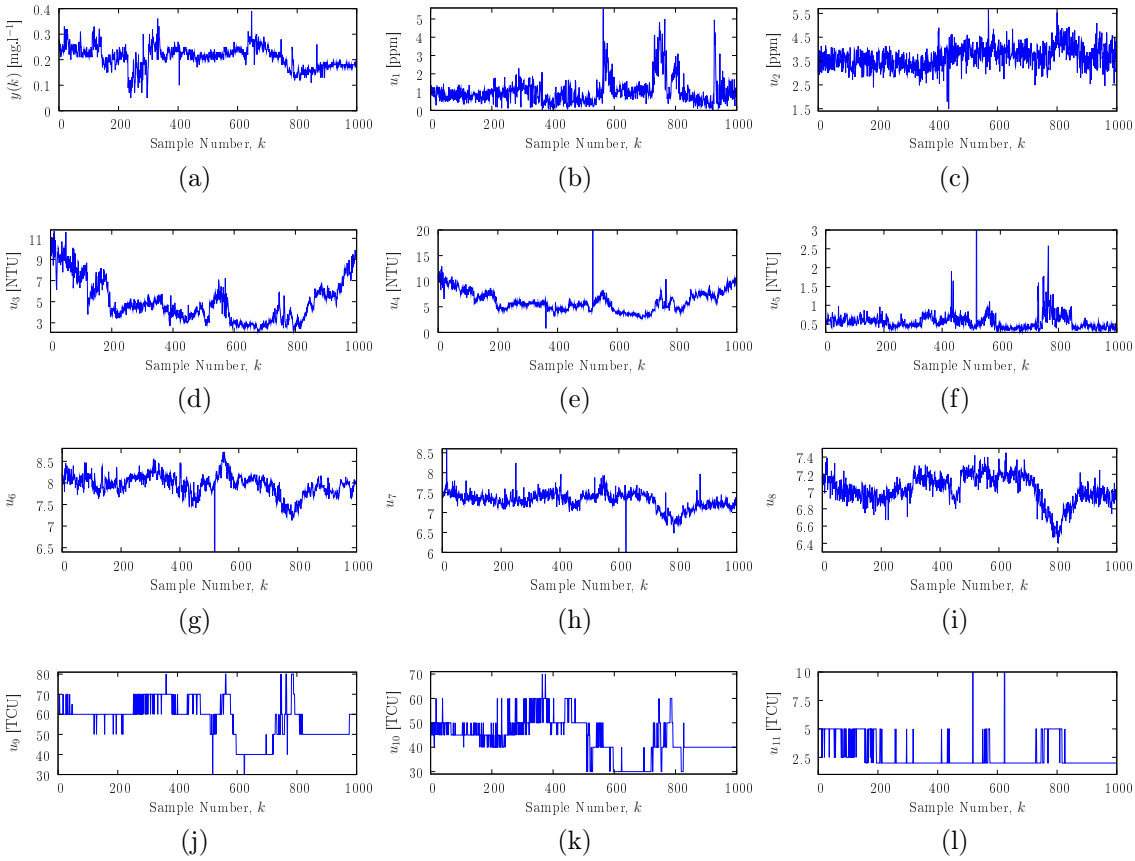


Figure 4.5: Plot of the variables of the WWTP data set:(a) y, (b) u_1, (c) u_2, (d) u_3, (e) u_4, (f) u_5, (g) u_6, (h) u_7, (i) u_8, (j) u_9, (k) u_{10}, and (l) u_{11}.

were configured to employ the variables grouping methodology (Section 4.3.3) as follows. For the HGA method the input variables were divided into two groups (see Figure 4.7): one group for all the delayed versions of u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, and u_{11}, and the other group for all the delayed versions of u_9, and u_{10}. For the HGA-FCRM (Figure 4.8), and AHGA-FCM (Figure 4.9) methods the input variables were divided into 11 groups, where each group contains all the delayed versions of one input variable: group 1 is given by $[u_1(t-1), u_1(t-2), u_1(t-3)]$, group 2 is given by $[u_2(t-1), u_2(t-2), u_2(t-3)]$, ..., and group 11 is given by $[u_{11}(t-1), u_{11}(t-2), u_{11}(t-3)]$.

As can be seen in Figure 4.6 and Table 4.2, the modeling performance attained in all of the HGA, HGA-FCRM, and AHGA-FCM methodologies is good. HGA-FCRM

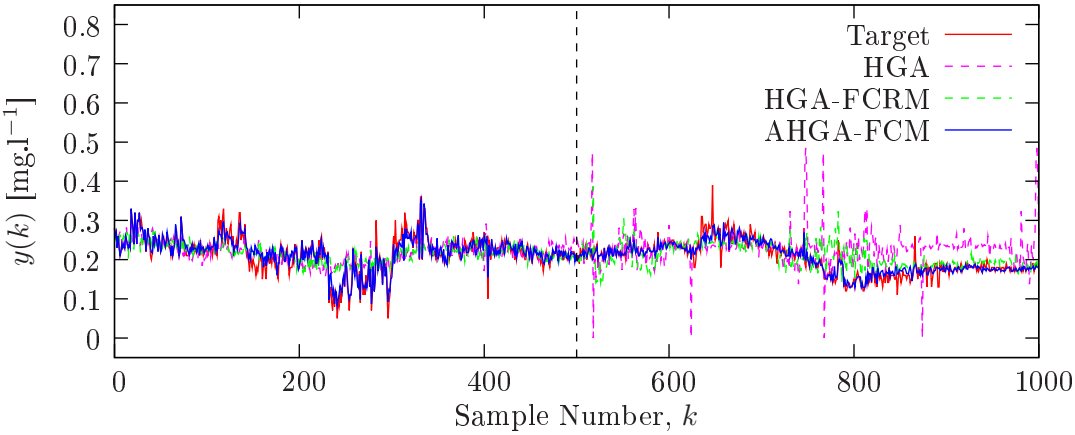


Figure 4.6: Modeling performance on the wastewater treatment system data set by the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies. The first half of the time interval of the graph corresponds to the training data set, and the second half corresponds to the test data set.

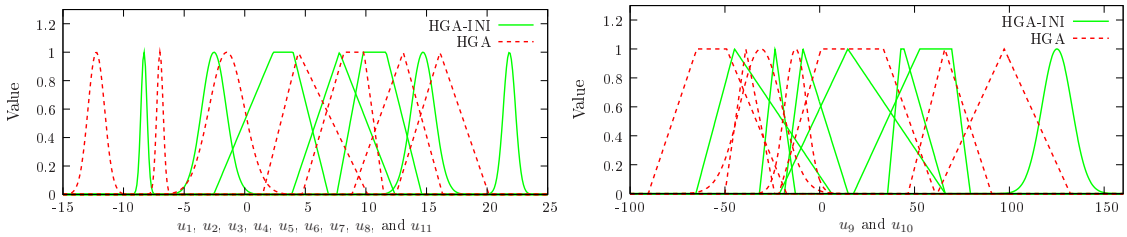


Figure 4.7: Membership functions of the proposed HGA method and of its best initial individual (HGA-INI) on Level 2, for all the delayed versions of the input variables (a) u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, and u_{11}; and (b) u_9, and u_{10}, for the WWTP process.

and AHGA-FCM which integrate initialization methodologies attain better modeling performance when compared to HGA. Additionally, as this WWTP data set changes its behavior on the second half, the prediction performance is somewhat degraded in the test data set (second half of the data set) when compared to the train data set, except for the AHGA-FCM, because AHGA-FCM is an adaptive methodology that adapts the consequent of the fuzzy rules when the process changes. In Table 4.2, the identification performance of the proposed methodologies are also quantitatively

Table 4.2: Comparison results of the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies with three state of the art non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and with two state of the art adaptive approaches: RPLS and ILLSA; for the WWTP test data set.

Method	Number of rules	Number of inputs	Inputs	$1/MSE$
ELM	-	-	All variables	419.5
MLP	-	-	All variables	424.3
RPLS	-	-	All variables	840.9
ILLSA	-	-	All variables	1197.6
HGA by [Delgado *et al.*, 2009]	20	27	$u_1(t-1)$, $u_1(t-2)$, $u_1(t-3)$, $u_2(t-2)$, $u_2(t-3)$, $u_3(t-1)$, $u_3(t-2)$, $u_3(t-3)$, $u_4(t-1)$, $u_4(t-2)$, $u_4(t-3)$, $u_5(t-1)$, $u_5(t-2)$, $u_5(t-3)$, $u_7(t-1)$, $u_7(t-2)$, $u_7(t-3)$, $u_8(t-1)$, $u_8(t-2)$, $u_8(t-3)$, $u_9(t-2)$, $u_{10}(t-1)$, $u_{10}(t-2)$, $u_{10}(t-3)$, $u_{11}(t-1)$, $u_{11}(t-2)$, $u_{11}(t-3)$	279.1
HGA	20	15	$u_1(t-2)$, $u_2(t-3)$, $u_3(t-2)$, $u_3(t-3)$, $u_4(t-1)$, $u_4(t-2)$, $u_4(t-3)$, $u_5(t-2)$, $u_5(t-3)$, $u_6(t-3)$, $u_7(t-1)$, $u_7(t-2)$, $u_8(t-3)$, $u_{10}(t-2)$, $u_{11}(t-3)$	441.6
HGA-FCRM	20	11	$u_1(t-1)$, $u_2(t-1)$, $u_3(t-1)$, $u_4(t-1)$, $u_5(t-1)$, $u_6(t-2)$, $u_7(t-1)$, $u_8(t-1)$, $u_9(t-3)$, $u_{10}(t-1)$, $u_{11}(t-1)$	659.3
AHGA-FCM	10	17	$u_1(t-1)$, $u_1(t-3)$, $u_2(t-2)$, $u_3(t-1)$, $u_3(t-2)$, $u_5(t-2)$, $u_6(t-2)$, $u_6(t-3)$, $u_7(t-1)$, $u_7(t-2)$, $u_7(t-3)$, $u_8(t-1)$, $u_8(t-2)$, $u_8(t-3)$, $u_9(t-3)$, $u_{10}(t-1)$, $u_{11}(t-3)$	5791.7

compared with three state of the art non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and with two state of the art adaptive approaches: RPLS and ILLSA. Numerical results have shown that AHGA-FCM methodology has a superior performance when compared to the two state of the art adaptive methods (RPLS and ILLSA) and to the three state of art non-adaptive methods (MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]), and that HGA and HGA-FCRM methodologies have a superior performance when compared to the three state of art non-adaptive methods (MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]). From the numerical results presented in Table 4.2, the best identification performance was obtained by the proposed AHGA-FCM method. Moreover, the models learned with the HGA approaches have the advantage of being more interpretable than the other models with respect to the parameters.

Figure 4.10 presents the evolution of the fitness functions on Level 5 for all

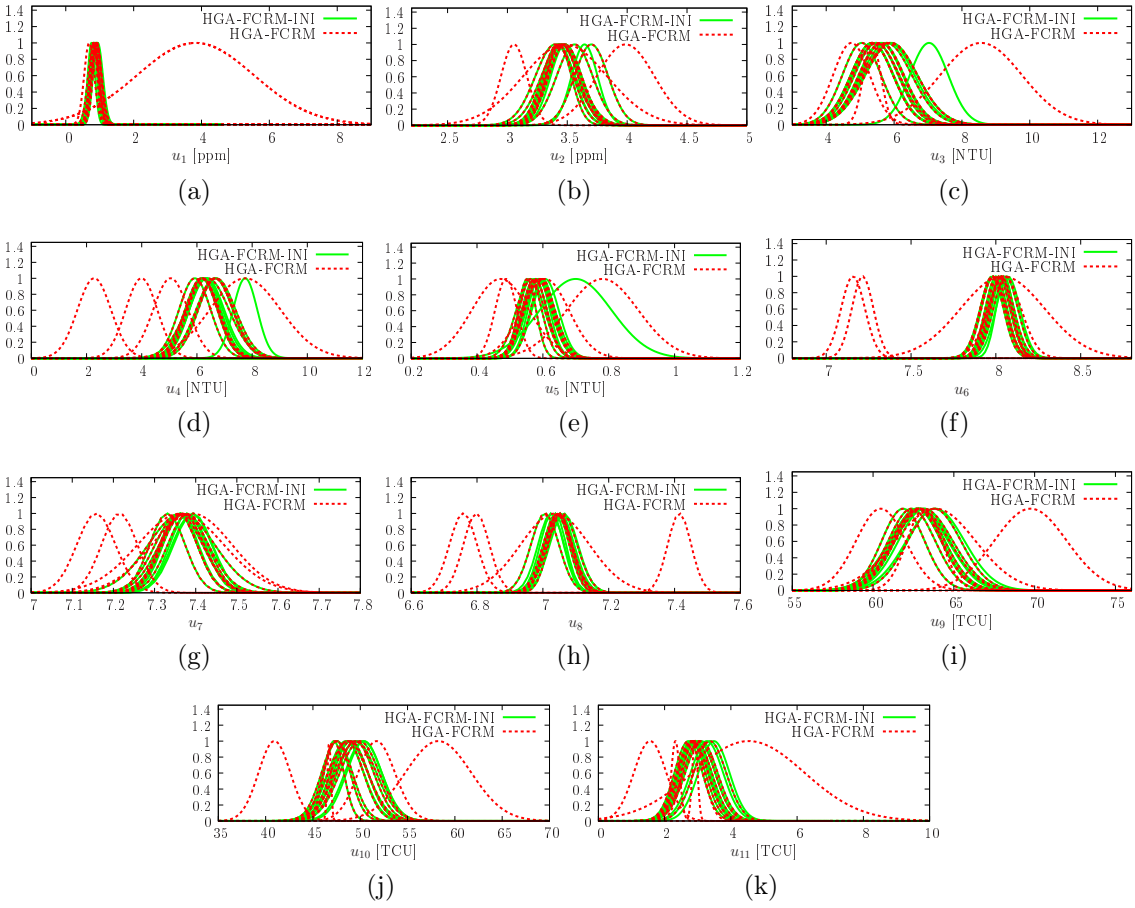


Figure 4.8: Membership functions of the proposed HGA-FCRM method and of their initialization method (HGA-FCRM-INI), for all the delayed versions of the input variable (a) u_1; (b) u_2; (c) u_3; (d) u_4; (e) u_5; (f) u_6, (g) u_7; (h) u_8; (i) u_9; (j) u_{10}; and (k) u_{11}, for the WWTP process.

generations of the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies. As can be seen, in the HGA-FCRM and AHGA-FCM methodologies there is a good initialization performed by the FCRM and FCM algorithms, respectively, that outperforms the initialization obtained by the HGA method, showing the importance of the using an initialization method. However, the initialization of FCRM method was only slightly better that the HGA initialization. Afterwards, the performance evolution attained by all the methods in good. AHGA-FCM methodology attains faster response of the fitness values, and better results, when compared to the results

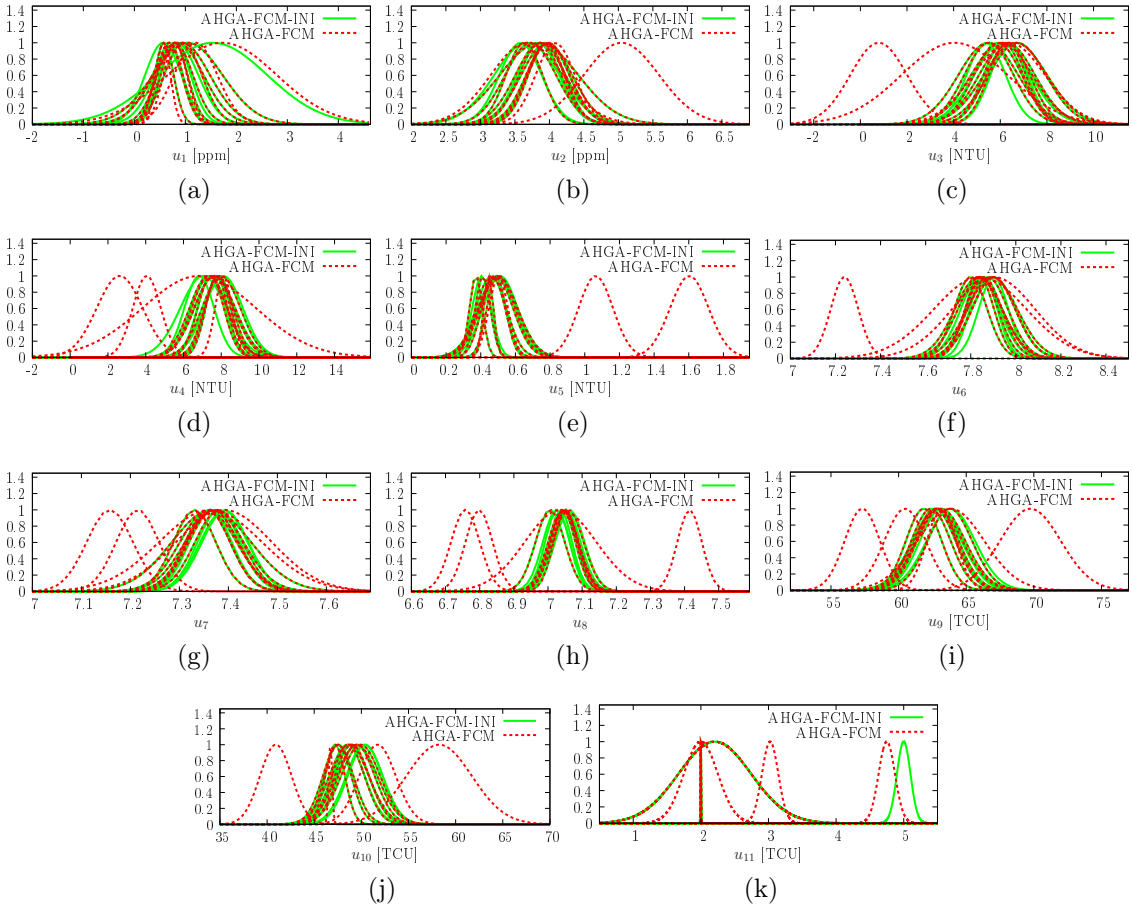


Figure 4.9: Membership functions of the proposed AHGA-FCM method and of their initialization method (AHGA-FCM-INI), for all the delayed versions of the input variable (a) u_1; (b) u_2; (c) u_3; (d) u_4; (e) u_5; (f) u_6, (g) u_7; (h) u_8; (i) u_9; (j) u_{10}; and (k) u_{11}, for the WWTP process.

obtained by the HGA and HGA-FCRM methods, due to its adaptation capacity.

4.6.2 Continuous-Stirred Tank Reactor

A Continuous Stirred Tank Reactor (CSTR) is a highly nonlinear process which is very common in chemical and petrochemical plants. In the process, a single irreversible, exothermic reaction is assumed to occur in the reactor. The CSTR for an exothermic irreversible reaction $A \rightarrow B$ is described by the following dynamic model based on a component balance for reactant A and on an energy balance

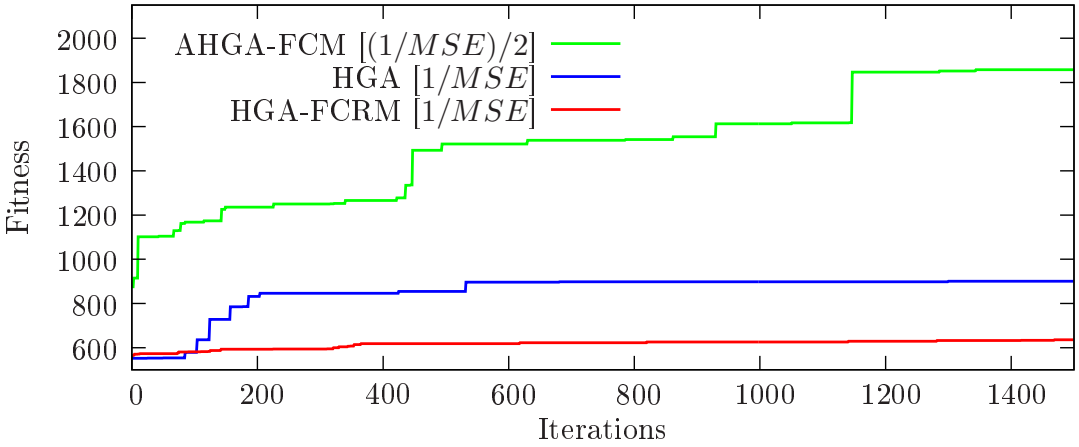


Figure 4.10: Evolution of the best fitness function value on Level 5 for all generations in the WWTP experiment by the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies.

[Morningred *et al.*, 1992]:

$$\begin{aligned}
\frac{\partial C_A(t+d_c)}{\partial t} &= \frac{q(t)}{V}\left(C_{A0}(t) - C_A(t+d_c)\right) - k_0 C_A(t+d_c)\exp\left(-\frac{E}{RT(t)}\right) \\
&\quad +\vartheta(t), \\
\frac{\partial T(t)}{\partial t} &= \frac{q(t)}{V}\left(T_0(t) - T(t)\right) - \frac{(-\Delta H)k_0 C_A(t+d_c)}{\rho_{c1}C_p}\exp\left(-\frac{E}{RT(t)}\right) \\
&\quad +\frac{\rho_{c2}C_{pc}}{\rho_{c1}C_p V}q_c(t)\left[1-\exp\left(\frac{-hA}{q_c(t)\rho_{c2}C_{cp}}\right)\right]\left(T_{c0}(t) - T(t)\right), \\
y(t) &= C_A(t), \quad u(t) = q_c(t),
\end{aligned} \tag{4.45}$$

where $\vartheta(t)$ is the stochastic disturbance.

The plant variables and the respective nominal values for this case study are described in Table 4.3. The sampling period was assumed to be $T = 0.1\,[\text{min}]$, and the time delay is assumed to be $d_c = 5T = 0.5\,[\text{min}]$.

A data set representative of the CSTR operation was constructed. The data set contains 1000 samples, and was obtained by applying the actuation command signal, $u(k)$, represented in Figure 4.11: in order to represent a possible real data set in industry, a sequence of step control signals was applied. To avoid abrupt variations in the command signal, a moving average filter with 20 elements was applied to the command signal.

Table 4.3: Variables of the continuous stirred tank reactor (CSTR) [Morningred *et al.*, 1992].

Variables-Description	Value
C_A - Product concentration	0.1 [mol/l]
T - Reactor temperature	438.54 [K]
q_c - Coolant flow rate	103.41 [l/min]
q - Process flow rate	100 [l/min]
C_{A0} - Feed concentration	1 [mol/l]
T_o - Feed temperature	350 [K]
T_{c0} - Inlet coolant temperature	350 [K]
V - CSTR volume	100 [l]
hA - Heat transfer term	7×10^5 [cal/min/K]
K_0 - Reaction rate constant	7.2×10^{10} [min^{-1}]
E/R - Activation energy term	1×10^4 [K]
$-\Delta H$ - Heat of reaction	-2×10^5 [cal/mol]
ρ_{c1}, ρ_{c2} - Liquid densities	1×10^3 [g/l]
C_p, C_{pc} - Specific heats	1 [cal/g/K]

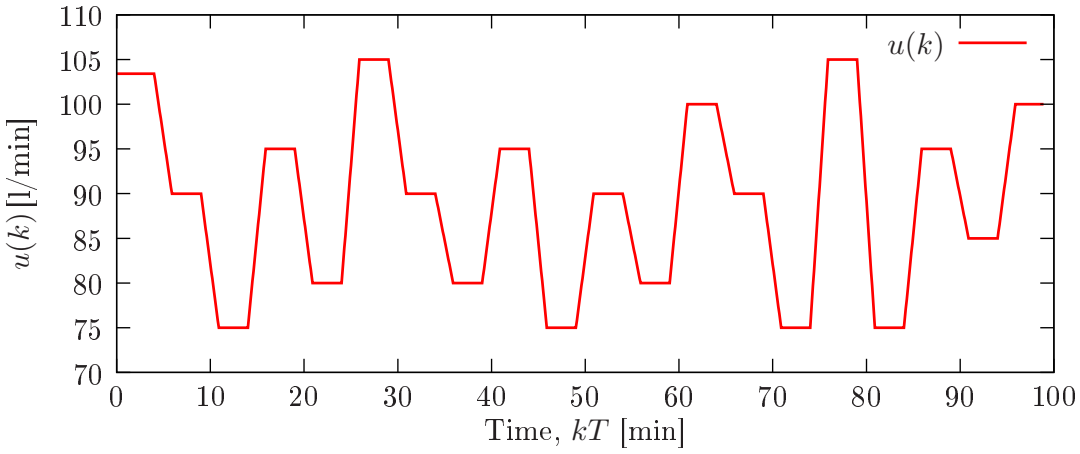


Figure 4.11: Control signal used to compile the data set for the CSTR process.

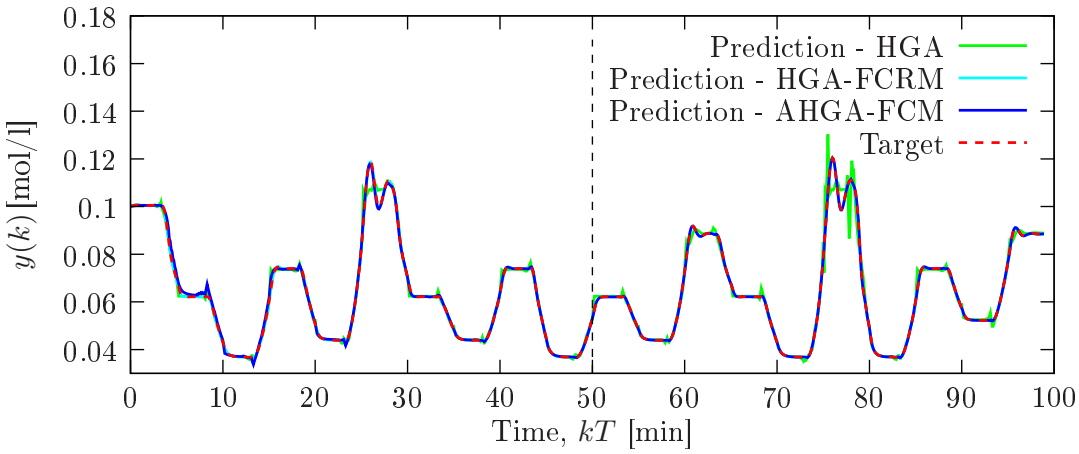


Figure 4.12: Modeling performance on the CSTR data set by the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies. The first half of the time interval of the graph corresponds to the training data set, and the second half corresponds to the test data set. The output of the plant is $y(k) = C_A(k)$.

The variables chosen for the data set were $C_A(k-2)$, $C_A(k-4)$, $C_A(k-6)$, $C_A(k-8)$, $C_A(k-10)$, $C_A(k-12)$, $q_c(k-1)$, $q_c(k-3)$, $q_c(k-5)$, $q_c(k-7)$, $q_c(k-9)$, $q_c(k-11)$, $q_c(k-13)$, where k is the sample time. In the HGA-FCM and AHGA-FCM methodologies, the number of clusters and the degree of fuzziness were chosen as $N = 20$, and $\eta = 2$, respectively.

Figure 4.12 shows the comparison among the predicted and target values of the output, $y(k) = C_A(k)$, of the CSTR plant, where the predicted values obtained with the HGA, HGA-FCRM, and AHGA-FCM methodologies are presented. The membership functions obtained by the proposed methods, and by their respective initialization methods, are shown in Figure 4.13. The Levels 2 of the algorithms were configured to employ the variables grouping methodology (Section 4.3.3) as follows. For the HGA, HGA-FCRM, and AHGA-FCM methods (Figure 4.13) the input variables were divided into two groups, where each group contains all the delayed versions of one input variable: group 1 is given by $[C_A(k-2), C_A(k-4), C_A(k-6), C_A(k-8), C_A(k-10), C_A(k-12)]$, and group 2 is given by $[q_c(k-1), q_c(k-3), q_c(k-5), q_c(k-7), q_c(k-9), q_c(k-11), q_c(k-13)]$.

As can be seen in Figure 4.12 and in Table 4.4 the modeling of the target variable, $C_A(t)$, given by the HGA-FCRM and AHGA-FCM methodologies is more accurate

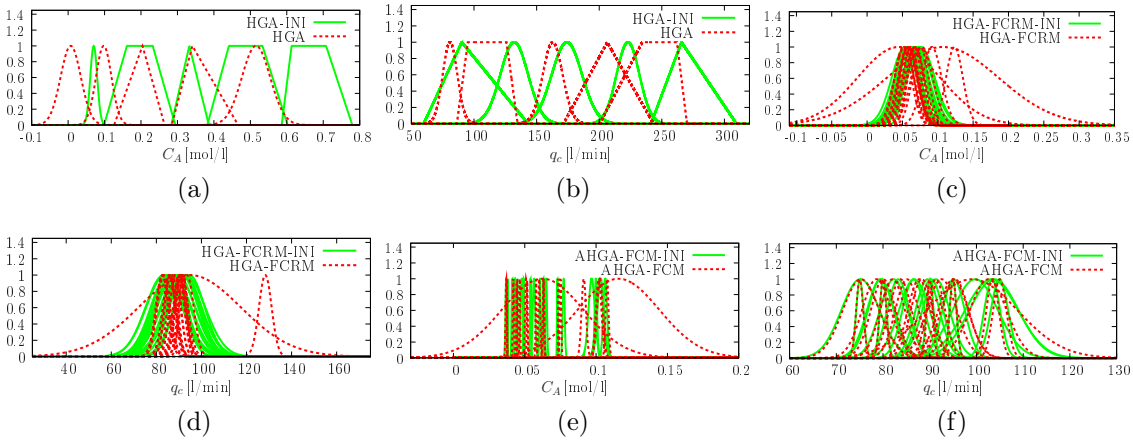


Figure 4.13: Membership functions of the proposed HGA, HGA-FCRM and AHGA-FCM methods and of their best initial individual (HGA-INI, HGA-FCRM-INI, and AHGA-FCM-INI, respectively) on Level 2, for all the delayed versions of the HGA input variables (a) C_A, and (b) q_c; the HGA-FCRM input variables (c) C_A, and (d) q_c; and the AHGA-FCM input variables (e) C_A, and (f) q_c, for the CSTR process.

Table 4.4: Comparison results of the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies with three state of the art non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and with two state of the art adaptive approaches: RPLS and ILLSA; on the CSTR test data set.

Method	Number of rules	Number of inputs	Inputs	$1/MSE$
ELM	-	-	All variables	9.1088×10^4
MLP	-	-	All variables	5.2259×10^7
RPLS	-	-	All variables	7.0871×10^4
ILLSA	-	-	All variables	3.9814×10^5
HGA by [Delgado *et al.*, 2009]	20	11	$C_A(k-2)$, $C_A(k-4)$, $C_A(k-6)$, $C_A(k-8)$, $C_A(k-10)$, $C_A(k-12)$, $q_c(k-1)$, $q_c(k-3)$, $q_c(k-5)$, $q_c(k-7)$, $q_c(k-9)$	6.4874×10^4
HGA	20	4	$C_A(k-4)$, $q_c(k-1)$, $q_c(k-5)$, $q_c(k-7)$	1.0088×10^5
HGA-FCRM	20	7	$C_A(k-2)$, $C_A(k-4)$, $C_A(k-8)$, $q_c(k-1)$, $q_c(k-3)$, $q_c(k-11)$, $q_c(k-13)$	6.0833×10^7
AHGA-FCM	20	13	All variables	7.3707×10^7

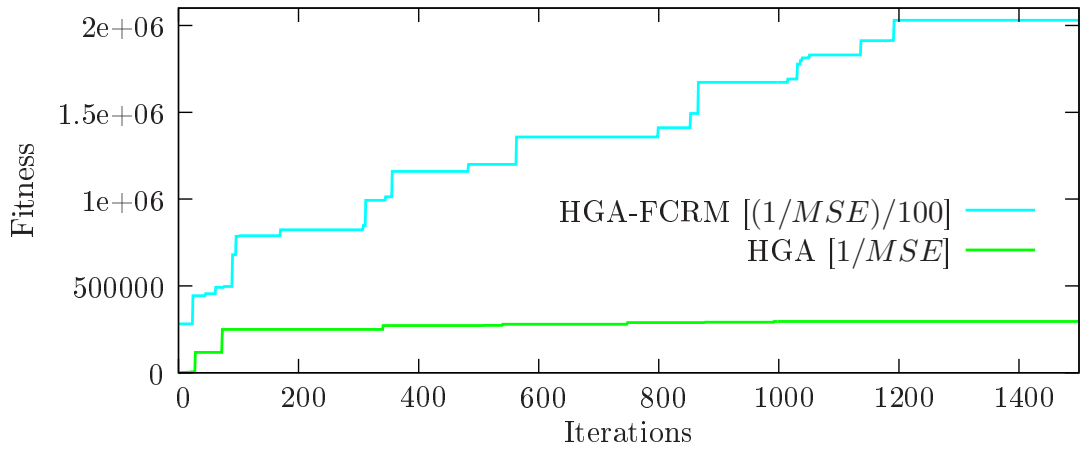


Figure 4.14: Evolution of the best fitness function value on Level 5 of the proposed HGA and HGA-FCRM methodologies for all generations in the CSTR experiment (training data set).

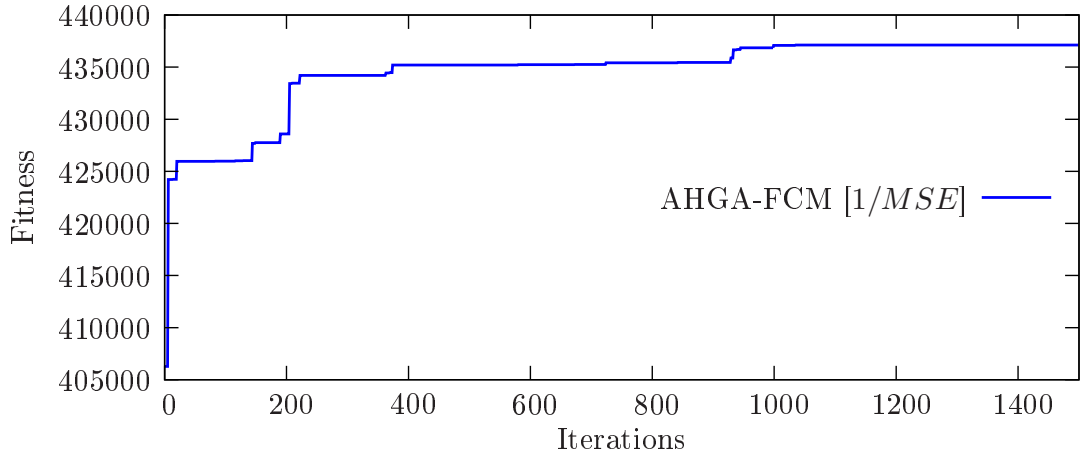


Figure 4.15: Evolution of the best fitness function value on Level 5 of the proposed AHGA-FCM methodology for all generations in the CSTR experiment (training data set).

and better than the one obtained by the HGA method. This fact, is supported by the evolution of the fitness functions on Level 5 for all generations in the training data set, presented in Figures 4.14 and 4.15. As can be seen, on the HGA-FCRM and AHGA-FCM methodologies there are good training initializations performed by the FCRM and FCM algorithms, respectively, that outperform the initializa-

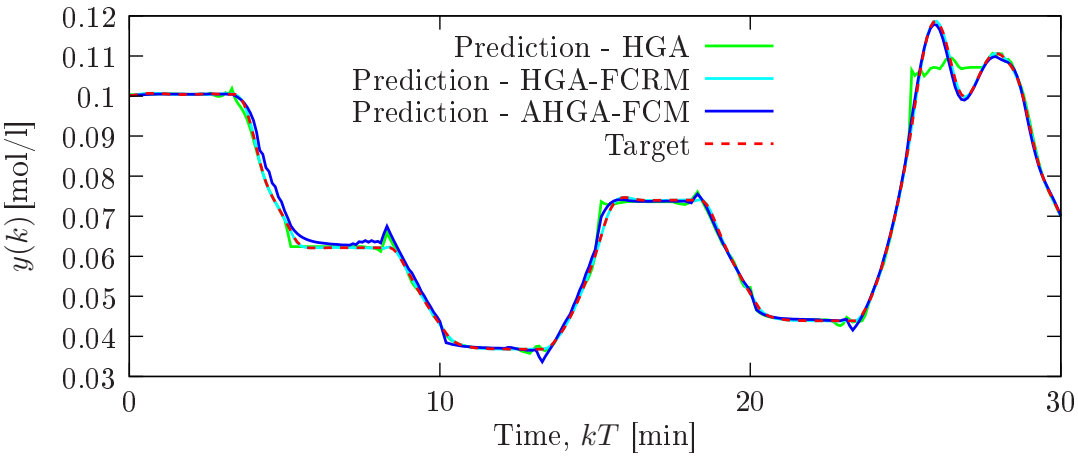


Figure 4.16: First data of the modeling performance on the CSTR train data set given by Figure 4.12.

tion obtained by HGA method. Afterwards, the training evolution attained by all methods in good, where the performance of the HGA-FCRM method is much better when compared to the AHGA-FCM methodology as can be seen by comparing Figures 4.14 and 4.15. In this data set, the modeling performance of the AHGA-FCM methodology given by the fitness function during training is lower than HGA-FCRM methodology, because the AHGA-FCM methodology needs some initial time instants to adapt its consequent parameters and the modeling error related with these initial instants has a large weight on the "final" total training error; And also because the HGA-FCRM methodology has a high modeling performance in this data set. An illustrative example of this fact is shown in Figure 4.16 that presents the prediction results corresponding to the initial 30 time intervals of Figure 4.12. However, after the initial instants the modeling performance of the AHGA-FCM methodology is higher than HGA-FCRM methodology as can be seen in Table 4.4 that presents information characterizing the results obtained on the test data set. The proposed AHGA-FCM methodology has better results, when compared to the results obtained by the HGA and HGA-FCRM (Table 4.4), because of its adaptation capacity. However, in this data set, the AHGA-FCM methodology has shown that some initial time instants are necessary to reach a higher performance.

In Table 4.4, the identification performance of the proposed methodologies were also quantitatively compared with three state of the art non-adaptive approaches:

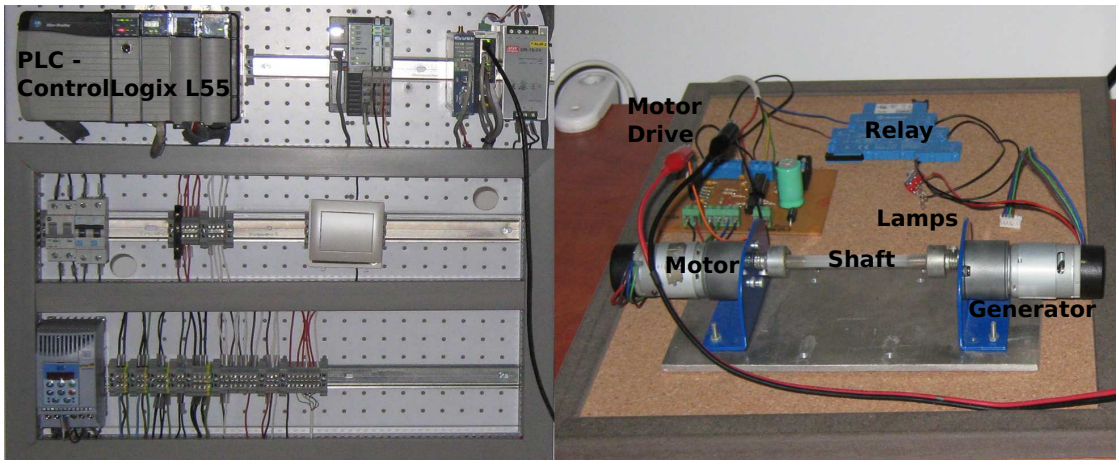


Figure 4.17: The experimental scheme of the two coupled DC motors.

MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and two state of the art adaptive approaches: RPLS and ILLSA. Numerical results have shown that the AHGA-FCM methodology has a superior performance when compared to the two state of the art adaptive methods (RPLS and ILLSA) and than the three state of the art non-adaptive methods (MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]), and that the HGA-FCRM methodology has a superior performance when compared to the three state of the art non-adaptive methods (MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]) and than the state of the art RPLS adaptive method. The HGA methodology has a superior performance when compared to the two state of the art non-adaptive methods (ELM, and the HGA proposed in [Delgado *et al.*, 2009]) and than the state of the art RPLS adaptive method, however MLP has a superior performance when compared to the proposed HGA. From the numerical results presented in Table 4.4, the best identification performance was given by the proposed AHGA-FCM method. Moreover, the models learned with the HGA approaches have the advantage of being more interpretable than the other models with respect to the parameters.

4.6.3 Real-World Setup of Two Coupled DC Motors

The real experimental system consists of two similar DC motors coupled by a shaft (Figure 4.17), where the first motor acts as an actuator, while the second motor is

used as a generator and to produce nonlinearities and/or a time-varying load. The system exhibits noise, parasitic electro-magnetic effects, friction and other phenomena commonly encountered in practical applications, that make control tasks more difficult.

The command signal $u(k)$ that is applied to the plant is the voltage to the DC motor, and is in the range of $[0, 12]$ [V]. The proposed control methodology runs on a PC that communicates by OPC[2] to a PLC[3] (ControlLogix L55 expanded with an analog I/O module for signal conditioning). The PLC provides the voltage command signal to the DC motor through the signal conditioning circuit. The output $y(k)$ of the plant is the velocity of the motors. The velocity units are [pp/(0.25 sec)] (pulses per 250 milliseconds). The encoder resolution is 12 *pulses per revolution* (PPR) which is poor and leads to low resolution velocity measures, which in turn makes control tasks more difficult. The generator has an electrical load composed of 2 lamps connected in parallel. When the lamps are connected in the generator circuit, the electrical load to the generator is increased (load resistance is decreased), and consequently the mechanical load that the generator applies to the motor also increases. Thus, it is possible to change the mechanical load to the motor, and consequently change its model. The sampling period was $T = 0.25$ [s].

To identify the experimental setup, a data set was constructed. The data set contains 952 samples, and it was obtained by applying to the motor (with the two lamps switched-off) the control signal represented in Figure 4.18. The following restriction is enforced on the control signals applied to the system: in absolute value, the difference between the command signals on two consecutive time steps is limited to 5% of the maximum control signal. The variables chosen for the data set were the first four delayed versions of the velocity $[y(k-1), y(k-2), y(k-3), y(k-4)]$, and the command signal and its first three delayed versions $[u(k), u(k-1), u(k-2), u(k-3)]$, where k is the sample time. In the HGA-FCRM and AHGA-FCM methodologies, the number of clusters and the degree of fuzziness were chosen as $N = 8$, and $\eta = 2$, respectively.

Figure 4.19 shows the comparison of the velocity values of the motor (with the two lamps switched-off) obtained by the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies, and the target velocity values. The membership functions

[2]OLE (Object Linking and Embedding) for Process Control.
[3]Programmable Logic Controller.

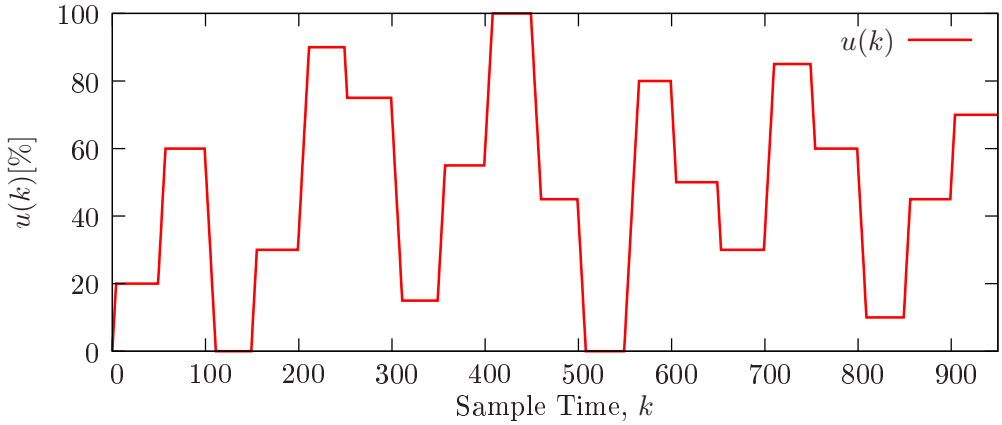


Figure 4.18: Control signal used to compile the data set on the DC motors process.

Table 4.5: Comparison of results of the proposed methodologies HGA, HGA-FCRM and AHGA-FCM with three state of the art non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and with two state of the art adaptive approaches: RPLS and ILLSA; for the DC Motors test data set.

Method	Number of rules	Number of inputs	Inputs	$1/MSE$
ELM	-	-	All variables	0.0149
MLP	-	-	All variables	0.0297
RPLS	-	-	All variables	0.0241
ILLSA	-	-	All variables	0.0197
HGA by [Delgado *et al.*, 2009]	20	7	$y(k-1)$, $y(k-2)$, $y(k-3)$, $y(k-4)$, $u(k)$, $u(k-1)$, $u(k-2)$	0.0072
HGA	20	3	$y(k-2)$, $y(k-4)$, $u(k-2)$	0.0158
HGA-FCRM	20	8	All variables	0.0304
AHGA-FCM	20	4	$y(k-1)$, $y(k-2)$, $u(k-1)$, $u(k-3)$	0.0609

obtained by the proposed methods and respective initialization methods are shown in Figure 4.20. The Levels 2 of the algorithms were configured to employ the variables grouping methodology (Section 4.3.3) as follows. For the HGA, HGA-FCRM, and AHGA-FCM methods (Figure 4.20) the input variables were divided into two groups, where each group contains all the delayed versions of one input variable: group 1 is given by $[y(k-1), y(k-2), y(k-3), y(k-4)]$, and group 2 is given by $[u(k), u(k-1), u(k-2), u(k-3)]$.

As can be seen in Figure 4.19 and in Table 4.5 the modeling of the velocity

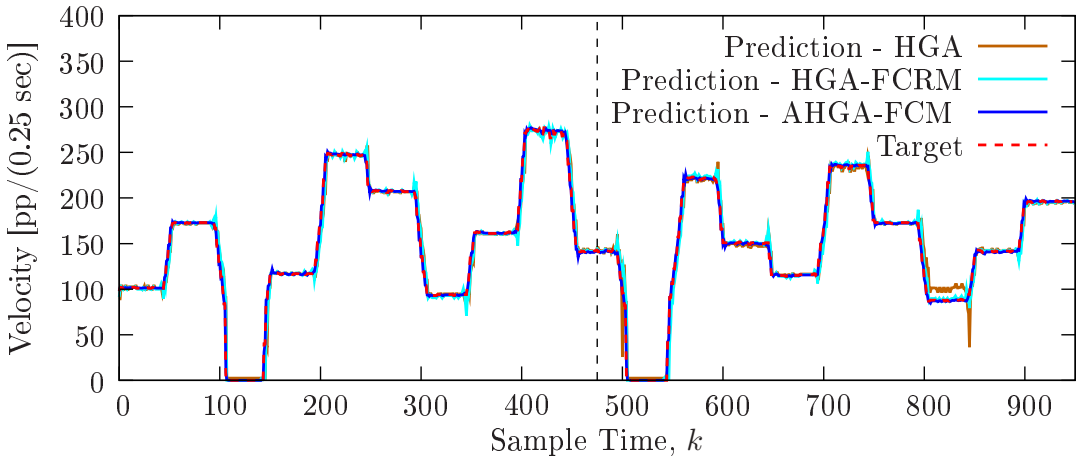


Figure 4.19: Modeling performance of the proposed HGA, HGA-FCRM, and AHGA-FCM methodologies for the Motors data set. The first half of the time interval of the graph corresponds to the training data set, and the second half corresponds to the test data set.

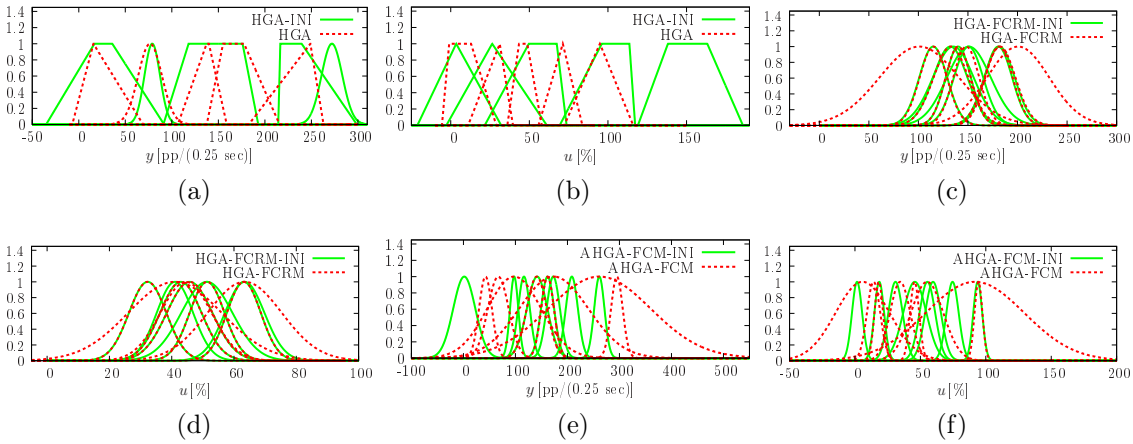


Figure 4.20: Membership functions of the proposed HGA, HGA-FCRM, and AHGA-FCM methods and of their best initial individual (HGA-INI, HGA-FCRM-INI, and AHGA-FCM-INI, respectively), for all the delayed versions of the HGA input variables (a) y, and (b) u; the HGA-FCRM input variables (c) y, and (d) u; and the AHGA-FCM input variables (e) y, and (f) u, for the DC motors process.

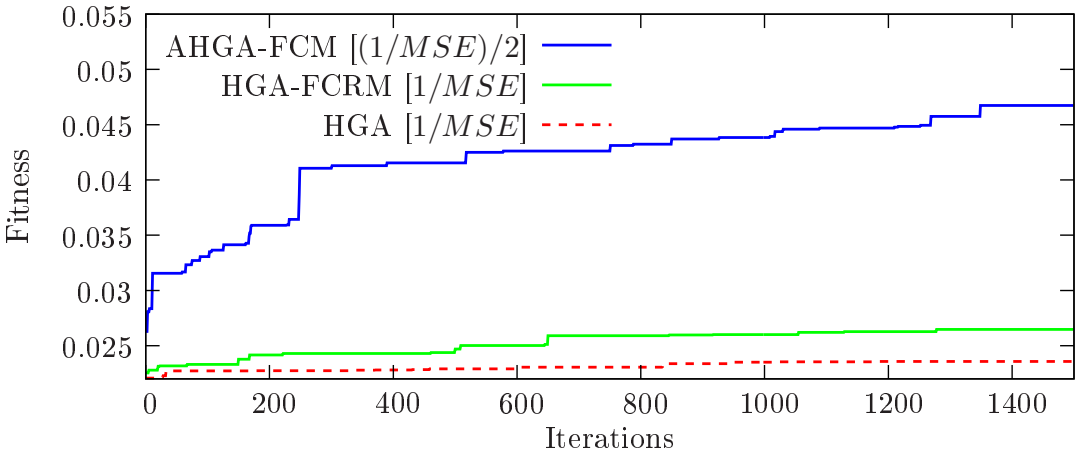


Figure 4.21: Evolution of the best fitness functions value on Level 5 for the HGA, HGA-FCRM, and AHGA-FCM for all generations on the real DC motors process.

by the proposed HGA-FCRM and AHGA-FCM methodologies is accurate and has better prediction results when compared to the ones obtained by the proposed HGA method. This fact is also supported by the evolution of the fitness functions on Level 5 for all generations during training, presented in Figure 4.21. As can be seen, there are good initialization capacities performed by the FCRM and FCM algorithms on the HGA-FCM and AHGA-FCM methodologies, respectively, that outperform the initialization obtained by the HGA method. Afterwards, the evolution of the fitness function attained by all methods in good. The proposed AHGA-FCM methodology attains faster learning response during training (Figure 4.21), and better prediction results during test when compared to the results obtained by the HGA and HGA-FCRM (Table 4.5). These better results of the AHGA-FCM are due to its adaptation capacity on the consequent parameters of the T-S fuzzy model.

In Table 4.5, the identification performance of the proposed methodologies were also quantitatively compared with three state of the art non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and two state of the art adaptive approaches: RPLS and ILLSA. Numerical results have shown that AHGA-FCM and HGA-FCRM methodologies have a superior performance when compared to the two state of the art adaptive methods (RPLS and ILLSA) and to the three state of the art non-adaptive methods (MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]), and that the HGA methodology has a superior

performance when compared to two state of the art non-adaptive methods (ELM, and the HGA proposed in [Delgado *et al.*, 2009]), however MLP has a superior performance when compared to the proposed HGA method. From the numerical results presented in Table 4.5, the best identification performance was obtained by the proposed AHGA-FCM method. Moreover, the models learned with the HGA approaches have the advantage of being more interpretable than the other models with respect to the parameters.

4.7 Conclusion

This chapter proposed new methodologies for identification of industrial processes. A hierarchical genetic algorithm was used to identify T-S fuzzy models from input/output data to approximate unknown nonlinear processes. The proposed methods are automatic tools for T-S fuzzy model design because they do not require any prior knowledge concerning the structure (e.g. the number of rules) and the database (e.g. antecedent fuzzy sets) of the T-S fuzzy model, and concerning the selection of the adequate input variables and their respective time delays for the prediction setting. Input variables and delays, antecedent aggregation operators, fuzzy rules, and type, location, and shape of membership functions were learned by a coevolutionary hierarchical GA.

Three methodologies were proposed in this chapter: a HGA method with random initialization, named as HGA (Section 4.3, Algorithm 4.1); a HGA with the fuzzy c-regression model (FCRM) initialization method, named as HGA-FCRM (Section 4.5, Algorithm 4.5); and a HGA that uses the fuzzy c-means (FCM) initialization method and an adaptive methodology, named as AHGA-FCM (Section 4.5, Algorithm 4.6).

To validate and demonstrate the performance and effectiveness of the proposed algorithms, they were tested on the identification problems of the estimation of the fluoride concentration in the effluent of a real-world wastewater treatment system, the product concentration on a simulated CSTR plant, and the velocity of a real experimental setup composed of two coupled DC motors. The presented results have shown that the developed evolving T-S fuzzy models can identify the nonlinear systems satisfactorily with appropriate input variables and delay selection, and with a reasonable number of rules. The proposed methodologies are able to design all the parts of the T-S fuzzy prediction model. The identification performance of the

proposed methodologies were also quantitatively compared with three state of art non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and two state of art adaptive approaches: RPLS and ILLSA. Numerical results have shown that AHGA-FCM methodology has a superior performance than the other adaptive methods, and that HGA-FCRM methodology has a superior performance than the other non-adaptive methods.

Thus, the aim of proposing methodologies to automatically identify a T-S fuzzy model (the input variables and delays, the fuzzy rules, and the membership functions) from input/output data to approximate unknown nonlinear processes, was reached as can be seen by the results of the proposed methods.

Chapter 5

Hierarchical Genetic Fuzzy System for Control

Contents

5.1 Introduction / State of the Art 86

5.2 Hierarchical Genetic Fuzzy System 88

5.2.1 Fuzzy Control Rules 88

5.2.2 Hierarchical Structure 89

5.3 Initialization Methods 94

5.4 Hierarchical Genetic Fuzzy Systems With Initialization Methods 94

5.4.1 HGA-Control Algorithm 98

5.4.2 AHGA-Control Algorithm 99

5.5 Experimental Results 102

5.5.1 Dissolved Oxygen Control 103

5.5.2 Real-World Control of Two Coupled DC Motors 111

5.6 Conclusion 117

5.1 Introduction / State of the Art

Fuzzy control systems (FCSs) have been used for a wide variety of industrial systems and consumer products, attracting the attention of many researchers. Fuzzy logic controllers (FLCs) are rule-based systems which are useful in the context of complex ill-defined processes, especially those which can be controlled by a skilled human operator without any mathematical knowledge of the process's underlying dynamics [Herrera *et al.*, 1995]. FLCs are based on a set of fuzzy control rules that make use of people's common sense and experience. However, there still exist many difficulties in designing fuzzy systems to solve certain complex nonlinear problems.

In general, it is not easy to determine the most suitable fuzzy rules and membership functions of a controller in order to control the output of a plant, when the only available knowledge concerning the process is the empirical information transmitted by a human operator. Thus, a major challenge in current fuzzy control research is translating human empirical knowledge into FLCs. As mentioned in Subsection 3.3.9, a possible candidate to meet this challenge is the application of the genetic algorithm (GA) approach to data extracted from a given process while it is being manually controlled.

GA's have been successfully applied to a wide variety of applications over the years. In particular, these algorithms have been applied in many automatic control problems, such as the development and tuning of FLCs. For that matter, they have been previously employed to select adequate sets of membership functions and fuzzy rules.

Alam and Tokhi [2008] proposed a GA-based hybrid fuzzy logic control strategy for input tracking and vibration reduction at the end point of a single-link flexible manipulator. For that matter, a GA is used to extract and optimize the rule base of the fuzzy logic controller. Ali and Ramaswamy [2009] present an optimal fuzzy logic control algorithm for vibration mitigation in buildings using magneto-rheological (MR) dampers. A micro-genetic algorithm (m-GA) and a Particle Swarm Optimization (PSO) approach are used to optimize the FLC parameters. In [Homayouni *et al.*, 2009], a genetic fuzzy logic control methodology is used to develop two production control architectures: genetic distributed fuzzy (GDF) and genetic supervisory

fuzzy (GSF) controllers. The GA is used to tune the input variable membership functions for the GSF and GDF controllers. Coban and Can [2010] designed a trajectory tracking genetic fuzzy logic controller for research reactors. Membership function boundaries and fuzzy control rule action weights were optimally determined by GAs. In [Kumar *et al.*, 2012], a new genetic swarm algorithm (GSA) is proposed for obtaining near optimal rule sets, and for membership function tuning, by combining strengths of GA and PSO. The GA is used to find the near optimal rules and PSO is used to tune the membership function.

The above cited methods only optimize membership function parameters and consider the other components of the fuzzy system, such as implication, aggregation, and defuzzifier methods, to be fixed. Other common limitation is the selection of the correct set of input variables. The variable selection process is usually manual and not accompanied with the accurate selection of the right time delays, probably leading to low-accuracy results [Souza *et al.*, 2013].

Fuzzy controllers can be classified into two categories [Wang, 1997]: direct and indirect controllers. In direct fuzzy control, the controller is constructed from human control knowledge, and in the indirect fuzzy control the controller is constructed from human knowledge about the plant to be controlled.

Chapter 4 has proposed useful methodologies towards the implementation of indirect fuzzy control. Specifically methodologies for learning T-S fuzzy models have been proposed. A hierarchical genetic algorithm (HGA) approach with five levels is used to optimize the parameters of T-S fuzzy systems. In the first level, the input variables and respective delays are chosen with the goal of attaining the highest possible prediction accuracy for the T-S fuzzy model. The selection of variables and delays is performed jointly with the learning of the fuzzy model, which increases the global optimization performance. The second level encodes the antecedent membership functions. The individual rules are defined at the third level. The population of the set of rules is defined in the fourth level, and a population of fuzzy systems is treated at the fifth level. The consequent parameters are given by methodologies based on the least squares method.

The work proposed in this chapter is based on the work of Chapter 4, although, applied to controller design, i.e. for a direct fuzzy control approach. The main advances and differences contemplated in this work are the improvement of the whole hierarchical structure to automatically extract the fuzzy control rules, and

to extract a standard FLC. The T-S fuzzy model approach (for an identification problem) that was pursued in the methods proposed in Chapter 4, is replaced in the current chapter by a standard fuzzy control system, i.e. where the consequent part of the rules is represented by membership functions, rather than making use of consequent functions as in T-S fuzzy models. For the implication operator, only the Mamdani (minimum and product) implications were considered here. However, for the implication, and for other parts of the FLC, the proposed methodology gives freedom to choose the possible options to be considered for learning the FLC.

The main objective of this chapter is the extraction of a FLC from data extracted from a given process while it is being manually controlled. After the extraction of the FLC by the proposed method, in order to obtain better control results, if necessary, the learned FLC can be improved manually by using the information transmitted by a human operator, and/or the learned FLC could be easily applied to initialize the required fuzzy knowledge-base of direct adaptive controllers such as the ones used in [Mendes *et al.*, 2011].

Two methodologies are proposed in this chapter: a HGA for control with the initialization method based on [Andersen *et al.*, 1997], named as HGA-Control (Section 5.4.1, Algorithm 5.4), and a HGA with the initialization method based on [Andersen *et al.*, 1997] and on a fuzzy *c*-means (FCM) clustering algorithm [Celikyilmaz and Trksen, 2009; Dovžan and Škrjanc, 2011], named as AHGA-Control, and where to improve the results of the learned FLC, a direct adaptive fuzzy controller used in [Mendes *et al.*, 2011] was applied (Subsection 5.4.2, Algorithm 5.5).

5.2 Hierarchical Genetic Fuzzy System

5.2.1 Fuzzy Control Rules

The type of knowledge-base used in this chapter is defined by a collection of fuzzy control rules of the following form:

$$R_i : \text{IF } x_1(k) \text{ is } A_1^i, \text{ and } \ldots \text{ and } x_n(k) \text{ is } A_n^i \text{ THEN } u \text{ is } B_i, \tag{5.1}$$

where $i = 1, \ldots, N$; x_j $(j = 1, \ldots, n)$ are the input variables of the fuzzy system, u is the output variable, and A_j^i and B_i are linguistic terms characterized by the fuzzy

membership functions $\mu_{A_j^i}(x)$ and $\mu_{B_i}(u)$, respectively.

5.2.2 Hierarchical Structure

This section describes the hierarchical structure, which has as its main goal to learn a FLC that does not require prior explicit expert knowledge about the plant to be controlled. To do so, the work proposes an automatic method based on a HGA, with five levels, for the extraction of all fuzzy parameters of a FLC from a data set obtained from an existing controller (human or automatic). The proposed coevolutionary model is constituted by five hierarchical levels (Figure 5.1).

1. The first level represents the population of the set of input variables and their respective time delays;

2. The second level represents the population of the antecedent and consequent membership functions which constitute the fuzzy control rules;

3. The population of individual rules is defined at the third level;

4. The population of sets of fuzzy rules is obtained on the fourth level;

5. The fifth level represents the population with the indexes of the selected elements of the previous levels, as well as the antecedent aggregation method, the inference engine, and the defuzzification method that are used on the fuzzy controller.

The descriptions of the Levels 1, and 4 are the same as the descriptions done in Subsection 4.3.3 for the same levels. The modifications are introduced on Levels 2, 3 and 5. On Level 2 the partition sets associated with the output variable were added, on Level 3 it was added one allele that characterizes the output variable, and on Level 5 alleles were added to represent the methods used for antecedent aggregation, inference engine, and defuzzification. The detailed description of each level is given below.

***Level* 1:** it is the same as the Level 1 defined in Subsection 4.3.3. In the example of Figure 5.1 the selected pairs of variables and delays correspond to $x_1 = e(t-1)$, and $x_3 = d(t-1)$, where $e(t) = r(t) - y(t)$ is the tracking error, $r(t)$ is the desired reference for the output of the process $y(t)$, and $d(t)$ is the derivative of $e(t)$.

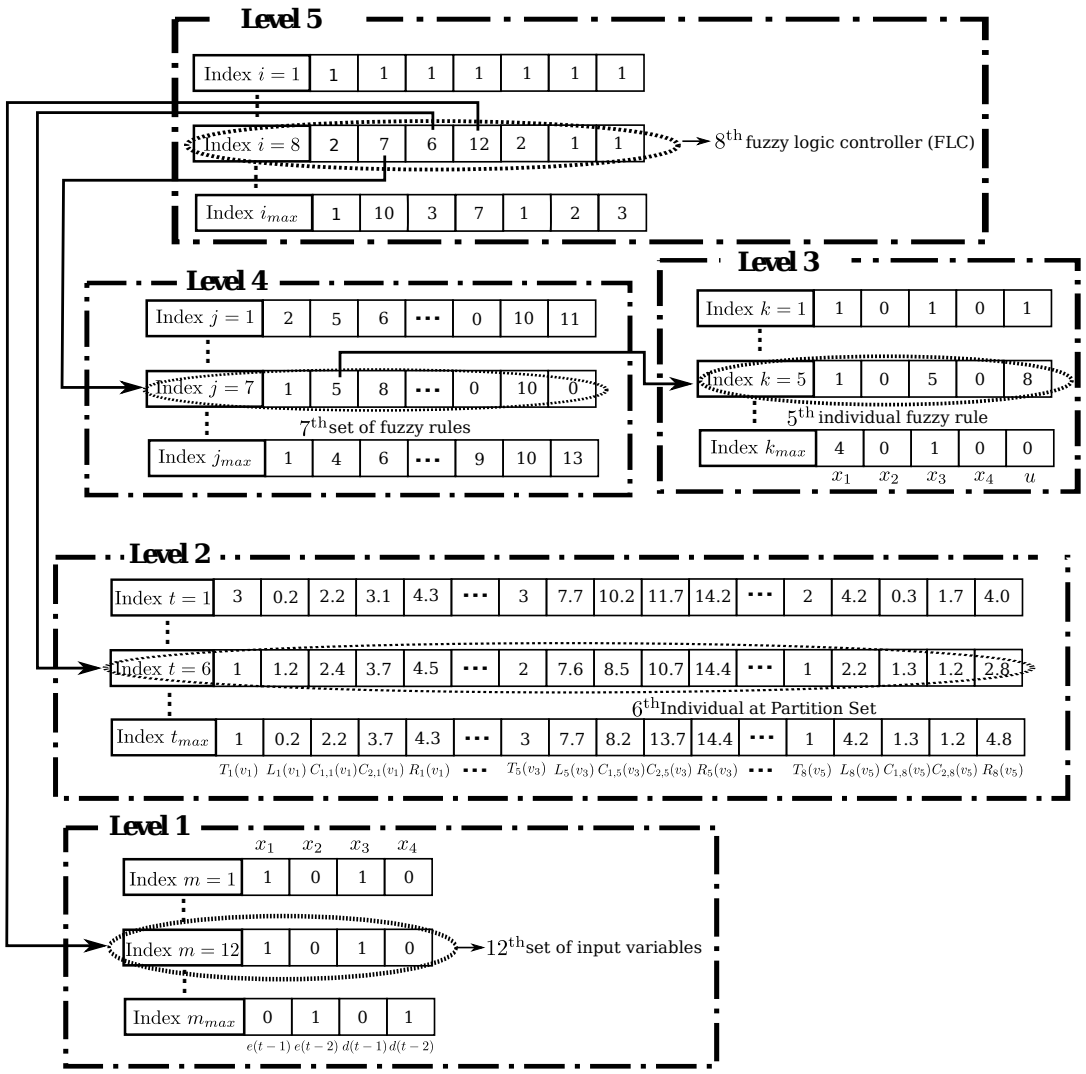


Figure 5.1: Encoding and hierarchical relations among the individuals of different levels of the genetic hierarchy of the HGA-Control approach.

***Level* 2:** contains every membership function defined in the universes of discourse of the variables involved. The chromosome is formed by the aggregations, one after another, of all partition sets associated with the input and output variables. The partition set of a variable is a collection of fuzzy sets associated to the variable. In each chromosome, associated to each variable there is exactly one partition set. For each variable, the range of possible values that the variable may take is fixed by the designer. The structure of this level is the same of the Level 2 defined on

Subsection 4.3.3 but contains also the partition sets associated with the output variable. Specifically, variables v_h, for $h = 1, \ldots, n+1$ are contemplated, where v_h, for $h = 1, \ldots, n$, correspond to input variables, and v_{n+1} corresponds to the output variable.

In Figure 5.1, it is illustrated, as an example, the 6th ($t = 6$) individual partition set of Level 2, that represents an example of the membership function used in the 5th ($k = 5$) individual fuzzy rule on Level 3. This partition set is illustrated by the 1st membership function of x_1, (v_1), the 5th membership function of x_3, (v_3), and the 8th membership function of u, (v_5), and these membership functions are of types trapezoidal ($T_1(v_1) = 1$), triangular ($T_5(v_3) = 2$), and trapezoidal ($T_8(v_5) = 1$), respectively. Assuming $m_{2,7}(v_5) = 0.6$, and using (4.14)-(4.17), the parameters of the 8th membership function of u, (v_5) are obtained as follows:

$$m_{1,8}(v_5) = m_{2,7}(v_5) + C_{1,8}(v_5) = 0.6 + 1.3 = 1.9, \tag{5.2}$$

$$m_{2,8}(v_5) = m_{1,8}(v_5) + C_{2,8}(v_5) = 1.9 + 1.2 = 3.1, \tag{5.3}$$

$$b_{1,8}(v_5) = m_{1,8}(v_5) - L_8(v_5) = 1.9 - 2.2 = -0.3, \tag{5.4}$$

$$b_{2,8}(v_5) = m_{2,8}(v_5) + R_8(v_5) = 3.1 + 2.8 = 5.9, \tag{5.5}$$

as can be deduced from Figures 5.1, and 4.2.

To reduce the computational time and the number of parameters to be tuned, the variables grouping methodology, proposed in Level 2 of the HGA (Section 4.3.3), can be optionally employed.

***Level* 3:** it is constituted by a population of individual rules. The length of the chromosome is determined by the maximum number of input variables that can be selected by Level 1, plus an additional allele that characterizes the output variable. The chromosome is represented by integer encoding, where each allele contains the index of the corresponding antecedent or consequent membership function. Null values in antecedent indexes indicate the absence of the corresponding variable(s) in the rule, and null values in the consequent index indicate the absence of fuzzy rule for that chromosome. In the example of Figure 5.1, Level 3 of the GA hierarchy is illustrated by describing the 5th ($k = 5$) individual rule. As can be seen, in this rule x_1 is represented by its 1st membership function, x_3 is represented by its 5th membership function, and u is represented by its 8th membership function.

***Level* 4:** it is the same as the Level 4 defined in Subsection 4.3.3. In the example of Figure 5.1, taking into account the alleles that are filled, Level 4 of the GA hierarchy is illustrated by the 7th ($j = 7$) set of fuzzy rules that contains the 1st, 5th, 8th and 10th individual rules, where these rules are described/represented in Level 3 of the hierarchy (but only the 1st and 5th rules are illustrated in the Level 3 of Figure 5.1).

***Level* 5:** it represents a fuzzy system, i.e. all the information required to develop the fuzzy controller is contemplated at this level. The chromosome is represented by integer encoding and is constituted by seven alleles. The first allele represents the t-norm operator, used to implement the fuzzy "and" operations used for aggregation in rule antecedents. For this matter, the GA selects from between three types of t-norms: (1) product, (2) minimum, and (3) bounded difference (other aggregation operators can be used, see more in [Wang, 1997]). The second allele indicates the index, k, of a set of rules specified in Level 4. The third allele contains a number of a t-th partition set given by Level 2. The fourth allele represents the index, m, of the set of input variables selected in Level 1. The fifth allele specifies the type of implication operator used. For this study the implemented implication methods were the (1) Mamdani product, and the (2) Mamdani minimum [Wang, 1997]. The sixth allele indicates the type of operator used to perform aggregation of the rules, where the following operators have been used as possibilities: (1) Maximum, (2) Bounded sum, and (3) Normalized sum. Finally, the seventh allele is responsible for determining the type of defuzzifier. The considered defuzzifiers are: (1) center of gravity (COG), (2) first of maximum (FOM), (3) last of maximum (LOM), and (4) mean of maximum (MeOM). All these operators (t-norm, implication, aggregation, and defuzzifier) can be consulted in [Mendes *et al.*, 2011].

An example of the encoding and the hierarchical relations is given in Figure 5.1. In this example the 8th individual at Level 5 indicates that the corresponding fuzzy system uses the minimum t-norm operator as the antecedent aggregation method (allele 1), the 7th set of fuzzy rules of Level 4 (allele 2), the 6th partition set of Level 2 (allele 3), the 12th set of input variables and delays selected in Level 1 (allele 4), minimum implication (allele 5), maximum aggregation (allele 6), and center of gravity deffuzification (allele 7). The 7th set of fuzzy rules (Level 4) contains the 1st, 5th, 8th, and 10th individual rules, where the 5th individual rule (Level 3), R_5, is composed by two input variables, x_1 and x_3, with linguistic terms 1 and 5,

respectively, and one output variable corresponding to linguistic term 8, i.e.:

$$R_5: \text{ IF } x_1(t) \text{ is "1" and } x_3(t) \text{ is "5" THEN } u(t) \text{ is "8".} \tag{5.6}$$

The linguistic terms "1", "5" and "8" of x_1, x_3, and u, respectively, are defined in the 6th chromosome of Level 2, and the input variables x_1, and x_3 are determined on Level 1.

***Fitness functions*:** Let i_{max}, j_{max}, k_{max}, t_{max} and m_{max}, be the maximum numbers of chromosomes at levels 5, 4, 3, 2, and 1, respectively. The fitness functions of each individual i, j, k, t, and m, respectively of Level 5 to Level 1, of the hierarchical population are defined as follows:

- Fuzzy system (Level 5):

$$J_5^i = \frac{1}{\left(1 + \frac{2dim}{L}\right) \text{MSE}(\mathbf{u}, \hat{\mathbf{u}}^i)}, \tag{5.7}$$

  where $\text{MSE}(\mathbf{u}, \hat{\mathbf{u}}^i) = \frac{1}{L}\sum_{l=1}^{L}(u(l) - \hat{u}^i(l))^2$ is the mean square error between the target control output $\mathbf{u} = [u(1), \ldots, u(L)]^T$, and estimated control output $\hat{\mathbf{u}}^i = [\hat{u}^i(1), \ldots, \hat{u}^i(L)]^T$ obtained with individual i over L samples, and $\left(1 + \frac{2dim}{L}\right)$ is the Akaike's information criterion (AIC) [Espinosa *et al.*, 2004] which penalizes more complex individuals to avoid overparameterization. The complexity is measured by dim which represents the total number of the parameters of the tuned FLC.

- Rule base (Level 4):

$$J_4^j = max(J_5^{a_1}, \ldots, J_5^{a_p}), \tag{5.8}$$

  where $\{a_1, \ldots, a_p\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain rule-base j (set of fuzzy rules) on allele 2 of Level 5;

- Individual rule (Level 3):

$$J_3^k = mean(J_4^{b_1}, \ldots, J_4^{b_q}), \tag{5.9}$$

  where $\{b_1, \ldots, b_q\} \subseteq \{1, \ldots, j_{max}\}$ is the subset of all chromosomes of Level 4 that contain individual rule k;

- Partition set (Level 2):

$$J_2^t = max(J_5^{c_1}, \ldots, J_5^{c_r}), \tag{5.10}$$

where $\{c_1, \ldots, c_r\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain partition set t on allele 3 of Level 5;

- Inputs and delays selection (Level 1):

$$J_1^m = max(J_5^{d_1}, \ldots, J_5^{d_s}), \tag{5.11}$$

where $\{d_1, \ldots, d_s\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain the m-th selection of inputs and delays on allele 4 of Level 5.

5.3 Initialization Methods

The use of random initialization of the population in a GA may result in a very exhausting optimality search, requiring more iterations to attain convergence. So in order to obtain an initial satisfactory starting point, that could also be a better starting solution, reducing the computational cost and increasing the algorithm's performance, two initialization methods are applied/proposed. First, it is proposed the initialization Algorithm 5.1 which is based on the algorithm proposed in [Andersen *et al.*, 1997], and is named as INICONTROL. Second, it is proposed the initialization Algorithm 5.2 which is based on [Andersen *et al.*, 1997] and on a fuzzy c-means (FCM) clustering algorithm [Celikyilmaz and Trksen, 2009; Dovžan and Škrjanc, 2011], and is named as INICONTROL-FCM.

The proposed initialization approach for hierarchical methodologies is presented in Algorithm 5.3.

5.4 Hierarchical Genetic Fuzzy Systems With Initialization Methods

Two methodologies are proposed in this section. First, it is proposed a HGA for control which uses the INICONTROL initialization method (Section 5.3, Algorithm

Algorithm 5.1 INICONTROL - Initialization algorithm of the antecedent and consequent membership functions to be used on the HGA-Control methodology.

procedure

For the sole role of specification of this Algorithm 5.1, let m be the total number of input variables used in the training and operation of the HGA. In the rest of the thesis, except also in Algorithm 5.2, this number is denoted by n;

for all sampling instances $l = 1, \ldots, L$ **do**

Apply the input vector $\mathbf{x}(l) = [x_1(l), \ldots, x_m(l)]^T$ to the system to be modeled (in this case the controller) and obtain the data sample ($\mathbf{x}(l)$,$u(l)$), where $u(l)$ is the output scalar;

end for

Construct a data set containing all the data samples ($\mathbf{x}(l)$,$u(l)$);

Using the collected data samples ($\mathbf{x}(l)$,$u(l)$), select a set of variables $X_s = \{x_{\upsilon_1}(t), \ldots, x_{\upsilon_n}(t)\}$ which will be used to initialize the HGA; $\upsilon_j \in \{1, \ldots, m\}$, for $j = 1, \ldots, n \leqslant m$; For the sole role of specification of this Algorithm 5.1, renumber the input variables such that X_s becomes $X_s = \{x_1(t), x_2(t), \ldots, x_n(t)\}$;

for all all input variables $x_j \in X_s$ **do**

Specify the minimum and maximum values, x_j^- and x_j^+, of input variable x_j (i.e. the universe of discourse limits); Define the number of individual membership functions K_j for each input variable x_j;

for all $k = 1, \ldots, K_j$ **do**

Determine the initial center, $c_{A_j^k}$, of each antecedent membership function $\mu_{A_j^k}$ (assuming a Gaussian type) by uniformly distributing the centers over the universe of x_j using an interval width of $\left(x_j^+ - x_j^-\right)/K_j$. In this way, the centers are chosen such that they are distributed evenly over the universe of x_j;

For each antecedent membership function $\mu_{A_j^k}$, determine the associated dispersion, σ_{kj}, where σ_{kj} is given by $\sigma_{kj} = (\alpha_j/K_j) \cdot \left(x_j^+ - x_j^-\right)$ and α_j is the scaling factor which influences the width of the Gaussian membership functions;

end for

end for

Form the rule-based system containing a different rule for every possible combination of antecedent membership functions, where the total number of rules is $N = \prod_{j=1}^{n} K_j$;

for all $i = 1, \ldots, N$ **do**

Obtain the centers, b_i, of the consequent membership functions by using the heuristic method, $b_i = \left(\sum_{l=1}^{L} u(l) \prod_{j=1}^{n} \mu_{A_j^i}(x_j(l))\right) / \left(\sum_{l=1}^{L} \prod_{j=1}^{n} \mu_{A_j^i}(x_j(l))\right)$;

end for

Build the output membership functions assuming (for example) a triangular shape where the centers are given by b_i, and the triangular aperture can be represented by $(u^+ - u^-)/(N.g)$, where u^+ and u^- are the upper and lower limits of the output universe of discourse, respectively, and is g a scaling factor (e.g. $g = 2, 4, 8, \ldots$).

end procedure

Algorithm 5.2 INICONTROL-FCM - Initialization algorithm of the antecedent and consequent membership functions to be used on the AHGA-Control methodology.

1. For the sole role of specification of this Algorithm 5.2, let m be the total number of input variables used in the training and operation of the HGA. In the rest of the thesis, except also in Algorithm 5.1, this number is denoted by n;

2. For all sampling instances $l = 1, \ldots, L$, apply the input vector $\mathbf{x}(l) = [x_1(l), \ldots, x_m(l)]^T$ to the system to be modeled (in this case the controller) and obtain the data sample ($\mathbf{x}(l)$,$u(l)$), where $u(l)$ is the output scalar;

3. Construct a data set containing all the data samples ($\mathbf{x}(l)$,$u(l)$);

4. Using the collected data samples ($\mathbf{x}(l)$,$u(l)$), select a set of variables $X_s = \{x_{\upsilon_1}(t), \ldots, x_{\upsilon_n}(t)\}$ which will be used to initialize the HGA; $\upsilon_j \in \{1, \ldots, m\}$, for $j = 1, \ldots, n \leqslant m$; For the sole role of specification of this Algorithm 5.2, renumber the input variables such that X_s becomes $X_s = \{x_1(t), x_2(t), \ldots, x_n(t)\}$;

5. Obtain the antecedent parameters of each rule i, namely the vector of centers $\mathbf{v}_i = [v_{i1}, \ldots, v_{in}]^T$ (4.35) and the vector of dispersions $\boldsymbol{\sigma}_i = [\sigma_{i1}, \ldots, \sigma_{in}]^T$ (4.36) of the antecedent membership functions, for $i = 1, \ldots, N$, using Algorithm 4.2, where N is the number of clusters; Using $\mathbf{v}_i$ and $\boldsymbol{\sigma}_i$, each Gaussian antecedent membership function $\mu_{A_j^i}$ becomes defined;

6. For $i = 1, \ldots, N$ do:

   (a) Obtain the centers, b_i, of the consequent membership functions of the i-th fuzzy rule by using the heuristic method,

   $$b_i = \frac{\left(\sum_{l=1}^{L} u(l) \prod_{j=1}^{n} \mu_{A_j^i}(x_j(l))\right)}{\left(\sum_{l=1}^{L} \prod_{j=1}^{n} \mu_{A_j^i}(x_j(l))\right)}, \tag{5.12}$$

   where $u(l)$ is the observation l of the output variable u, and $\mu_{A_j^i}(x_j(l))$ is the antecedent fuzzy membership function used for variable x_j on the i-th fuzzy rule;

7. Build the output membership functions assuming (for example) a triangular shape where the centers are given by b_i, and the triangular aperture can be represented by $(u^+ - u^-)/(N.g)$, where u^+ and u^- are the limits of the output universe of discourse, and is g a scaling factor (e.g. $g = 2, 4, 8, \ldots$).

Algorithm 5.3 Proposed initialization of the HGA-Control and AHGA-Control methodologies.

1. Compute the antecedent and consequent membership functions parameters using Algorithm 5.1 for INICONTROL or Algorithm 5.2 for INICONTROL-FCM;
2. Initialize the populations of all levels:
   (a) Level 1: initialize with ones the alleles of the first individual that represent the input variables used to initialize the HGA in Algorithm 5.1 for INICONTROL or Algorithm 5.2 for INICONTROL-FCM (these are the variables that belong to the set X_s defined in these algorithms). Initialize with zeros all the remaining alleles of the first individual;
   (b) Level 2: initialize all individuals with the antecedent and consequent membership functions computed in Step 1;
   (c) Level 3: initialize the first N individuals with the antecedent part of the fuzzy rules which is learned by the INICONTROL algorithm or the INICONTROL-FCM algorithm:
      i. For the INICONTROL algorithm: form the rule-based system containing every possible combination of antecedent membership functions;
      ii. For the INICONTROL-FCM algorithm: the way this is done is by initializing the first individual with ones, the second individual with twos, until the N-th individual with N's;
   (d) Level 4: initialize the first individual with the indexes of the first N individuals of Level 3. Initialize with zeros all the remaining alleles;
   (e) Level 5: initialize the first individual of Level 5 with ones;
   (f) The remaining individuals of Levels 1, 3, 4, and 5 are randomly initialized;

5.1), and is named as HGA-Control (Subsection 5.4.1, Algorithm 5.4). Second, it is proposed a HGA, named as AHGA-Control, which uses the INICONTROL-FCM initialization method (Section 5.3, Algorithm 5.2), and where to improve the results of the learned FLC, a direct adaptive fuzzy control methodology used in [Mendes *et al.*, 2011] is applied (Subsection 5.4.2, Algorithm 5.5) is applied for on-line adaptation of the consequent parameters of the fuzzy control rules. Note that, in both initialization methods, INICONTROL and INICONTROL-FCM, the antecedent membership functions are assumed to be of Gaussian type, and the consequent membership functions are assumed to be of triangular type. However, these assumptions are just for the INICONTROL and INICONTROL-FCM methods. The antecedent and conse-

Algorithm 5.4 Proposed HGA-Control algorithm.

1. Set Generation ← 1;
2. Initialize the populations of all levels with Algorithm 5.3, in the variant that uses INICONTROL;
3. Compute the fitness of each individual, for all levels from Level 5 to Level 1, as specified in equations (5.7)-(5.11);
4. If the stopping condition does not hold, then do for each level:
   (a) Generation ← Generation + 1;
   (b) Apply the evolutionary operators to form a new population: selection, crossover, and mutation;
   (c) Replace the current population with the new evolved population;
   (d) Return to Step 3.

quent membership functions given by the INICONTROL and INICONTROL-FCM initialization methods can be changed during the operation of the HGA-Control and AHGA-Control approaches, respectively, and in these approaches they can be of triangular, trapezoidal, or Gaussian type.

In the following subsections, similarly to Chapter 4, note that, each level of the genetic hierarchy is evolved separately as an independent genetic algorithm using its own population and its own fitness function. However, since the values of the fitness functions of each level depend on all the other populations, then the evolution of each level is also influenced by the evolution of every other level.

In this chapter, for all proposed methods, the stopping condition is a pre-defined maximum number of generations.

5.4.1 HGA-Control Algorithm

The main steps of the GA algorithm used to learn/improve the fuzzy controller parameters are presented in Algorithm 5.4. This Algorithm is based on the INICONTROL algorithm (Section 5.3, Algorithm 5.1) and on the HGA methodology presented in Section 5.2.

5.4.2 AHGA-Control Algorithm

The descriptions of Levels 1, 3, and 4 of the AHGA-Control algorithm are the same as the corresponding descriptions made in Subsection 5.2.2, except that Levels 1 and 4 that have a different fitness function, in the AHGA-Control algorithm, to penalize more complex individuals in order to avoid overparameterization. Specifically, Level 1 penalizes individuals with a large number of input variables and Level 4 penalizes individuals with a large number of fuzzy rules. Further modifications are introduced on Levels 2 and 5. On Level 2, a different representation of the antecedent and consequent membership functions is employed to improve the design/learning of the membership functions. In Subsection 5.2.2, on Level 2 the representation of the membership functions is given by the distance between parameters of membership functions, thus the position of a given membership function of a given variable is dependent of the previous membership function of the same variable. Thus, when an allele of a given membership functions is changed, all the membership functions that follow are affected by this modification which decreases the performance of the learning. On Level 5 the possibility of choosing the t-norm, implication, aggregation, and deffuzification operators, is not available, i.e. these operators are fixed and cannot be selected by the GA optimization algorithm. The goal for fixing these operators is to make it possible the use of the adaptation law defined in Section 3.1.6, in order to improve the results of the learned FLC. The following operators are considered: product t-norm, Mamdani product implication, and center-average deffuzification. The detailed description of each level is given below (see also Figure 5.2).

***Level* 1:** it is the same as the Level 1 defined on Subsection 5.2.2, except that the following different fitness function is used:

$$J_1^m = \frac{max(J_5^{d_1}, \ldots, J_5^{d_s})}{1 + \frac{nVar(m)}{nTotalVar}}, \tag{5.13}$$

where $\{d_1, \ldots, d_s\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain the m-th selection of inputs and delays on allele 3, i_{max} is the maximum number of chromosomes of the Level 5, $nVar(m)$ is the number of variables of the m-th chromosome of Level 1, and $nTotalVar$ is the total number of candidate input variables.

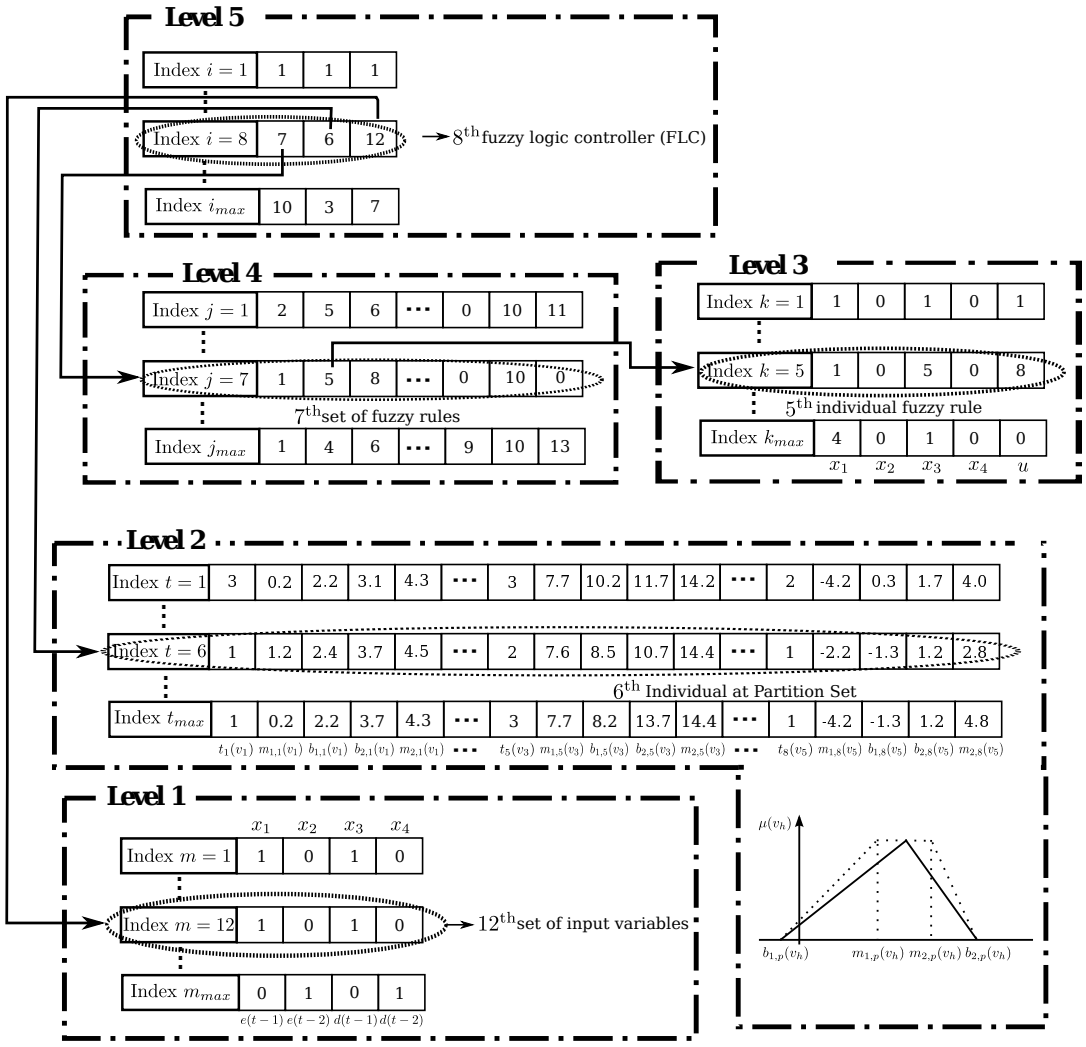


Figure 5.2: Encoding and hierarchical relations among the individuals of different levels of the genetic hierarchy of the AHGA-Control approach.

***Level* 2:** it is formed by the aggregations, one after another, of all partition sets (membership functions) associated with the input and output variables. A partition set of a variable is a collection of fuzzy sets associated to the variable. In each chromosome, associated to each variable there is exactly one partition set. To reduce the computational time and the number of parameters to be tuned, the variables grouping methodology, proposed in Level 2 of the HGA (Section 4.3.3), can be optionally employed in Level 2 on the AHGA-Control methodology.

Each chromosome is composed by a sequence of pentaplets of alleles. An exam-

ple of the structure of Level 2 can be seen in Figure 5.2, where in each pentaplet p the first allele uses integer encoding to represent the type of membership function: trapezoidal ($t_p(v_h) = 1$), triangular ($t_p(v_h) = 2$), or Gaussian ($t_p(v_h) = 3$); and the next four alleles ($m_{1,p}(v_h)$, $b_{1,p}(v_h)$, $b_{2,p}(v_h)$, and $m_{2,p}(v_h)$; see also Figure 4.2 for a definition of these parameters) use real encoding to represent the parameters of the membership function p ($p = 1, \ldots, K_h$) of an input or output variable v_h, ($h = 1, \ldots, n+1$), where v_h, for $h = 1, \ldots, n$, correspond to input variables, and v_{n+1} corresponds to the output variable, and K_h is the number of individual membership functions defined for variable v_h. The trapezoidal functions parameters are represented in Figure 5.2. For triangular and Gaussian membership functions, the center, $m_p(v_h)$, is found by the average between $m_{1,p}(v_h)$ and $m_{2,p}(v_h)$, i.e. $m_p(v_h) = (m_{1,p}(v_h) + m_{2,p}(v_h))/2$, and the dispersion of Gaussian functions is given by $\sigma_p(v_h) = (b_{2,p}(v_h) - b_{1,p}(v_h))/6$. The fitness function of this level is the same as the one defined for Level 2 in Subsection 5.2.2.

***Level* 3:** it is the same as the Level 3 defined in Subsection 5.2.2.

***Level* 4:** it is the same as the Level 4 defined in Subsection 5.2.2, except that the following different fitness function is employed (for chromosome j):

$$J_4^j = \frac{max(J_5^{a_1}, \ldots, J_5^{a_p})}{1 + \frac{nRules(j)}{NTotalRules}}, \tag{5.14}$$

where $\{a_1, \ldots, a_p\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain rule-base j on allele 1, $nRules(j)$ is the number of fuzzy rules of j-th chromosome of Level 4, and $NTotalRules$ is the maximum number of fuzzy rules that can be selected in Level 4.

***Level* 5:** represents a fuzzy system, i.e. all the information required to develop the fuzzy controller is contemplated at this level. The chromosome is represented by integer encoding and is constituted by three alleles. The first allele indicates the index, j, of the set of rules specified on Level 4. The second allele selects a t-th partition set defined Level 2. The third allele represents the index, m, of the set of input variables selected on Level 1. The fitness function of this level is given (for chromosome i) by

$$J_5^i = \frac{1}{\left(1 + \frac{nRules(j)+nVar(m)}{N+nTotalVar}\right) \mathrm{MSE}(\mathbf{u}, \hat{\mathbf{u}}^i)}, \tag{5.15}$$

where $nVar(m)$ is the number of variables of the m-th chromosome of Level 1, and $nTotalVar$ is the total number of candidate input variables, $\text{MSE}(\mathbf{u}, \hat{\mathbf{u}}^i) = \frac{1}{L}\sum_{l=1}^{L}(u(l) - \hat{u}^i(l))^2$ is the mean square error between the target control output $\mathbf{u} = [u(1), \ldots, u(L)]^T$, and the estimated control output $\hat{\mathbf{u}}^i = [\hat{u}^i(1), \ldots, \hat{u}^i(L)]^T$ obtained with individual i over L samples, and $\left(1 + \frac{nRules(j)+nVar(m)}{N+nTotalVar}\right)$ penalizes the more complex individuals to avoid overparameterization.

An example of the encoding and the hierarchical relations is given in Figure 5.2. In this example the 8th individual of Level 5 indicates that the corresponding fuzzy system uses the 7th set of fuzzy rules of Level 4 (allele 1), the 6th partition set of Level 2 (allele 2), and the 12th set of selected input variables and delays in Level 1 (allele 3). The 7th set of fuzzy rules (Level 4) contains the 1st, 5th, 8th, and 10th individual rules, where the 5th individual rule (Level 3) is composed of two input variables, x_1 and x_3, with linguistic terms 1 and 5, respectively, and one output variable, u, using to linguistic term 8, i.e.:

$$R_5: \text{ IF } x_1(t) \text{ is "1" and } x_3(t) \text{ is "5" THEN } u(t) \text{ is "8".} \tag{5.16}$$

The linguistic terms "1", "5", and "8" of x_1, x_3, and u, respectively, are defined in the 6th chromosome of Level 2, and the input variables x_1, and x_3 are determined on Level 1.

The main steps of the AHGA-Control algorithm used to learn/improve the fuzzy controller parameters are presented in Algorithm 5.5. To improve the results of the FLC learned in the offline part of AHGA-Control (Steps 1 to 4), a direct adaptive fuzzy controller can be applied as specified in Step 5 of Algorithm 5.5.

5.5 Experimental Results

This section presents simulation and real-world results to demonstrate the feasibility, performance and effectiveness of the proposed FLC design methodologies. The control of the dissolved oxygen in an activated sludge reactor within a simulated wastewater treatment plant (WWTP), and the velocity of a real experimental setup composed of two coupled DC motors are studied.

In both experiments, the results were obtained by considering that the crossover and mutation probabilities are 80% and 10%, respectively, the number of gener-

Algorithm 5.5 Proposed AHGA-Control algorithm.

1. Set Generation ← 1;
2. Initialize the populations of all levels with Algorithm 5.3, in the variant that uses INICONTROL-FCM;
3. Compute the fitness of each individual, for all levels from Level 5 to Level 1, using equations (5.15), (5.14), (5.9), (5.10), and (5.13), respectively;
4. If the stopping condition does not hold, then do for each level:
   (a) Generation ← Generation + 1;
   (b) Apply the evolutionary operators to form a new population: selection, crossover, and mutation;
   (c) Replace the current population with the new evolved population;
   (d) Return to Step 3.
5. Online operation: for/using each newly arriving online-sample, do:
   (a) Compute control signal $u(k)$ with (3.7);
   (b) If it is desirable to improve the results of the FLC learned in Steps 1 to 4, then:
      i. Adapt the consequent parameters $\boldsymbol{\Theta}$ using the adaptation law (3.11) defined in Section 3.1.6.

ations is $Gen_{max} = 1000$, and the numbers of chromosomes for each level of the architecture are: $i_{max} = 100$, $j_{max} = 80$, $k_{max} = 200$, $t_{max} = 15$, and $m_{max} = 200$. These parameters were tuned by means of experimentation. The proposed methodologies were implemented in the Matlab Software with the main functions being implemented in the C programming language to reduce computational time.

5.5.1 Dissolved Oxygen Control

This section addresses the application of the HGA algorithms proposed in this chapter for automatic extraction of the fuzzy parameters of a fuzzy controller in order to control the dissolved oxygen (DO) in an activated sludge reactor within a wastewater treatment plant.

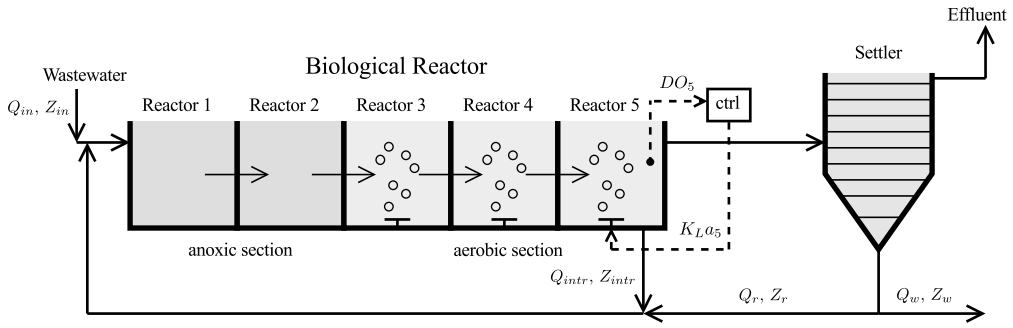


Figure 5.3: General overview of the BSM1 plant [Belchior *et al.*, 2012].

First, a data set of the plant being controlled by any controller (human or automatic) is obtained with the aim of providing a set of input/output data necessary for the HGAs to learn the FLC's parameters. The data set used in this thesis is obtained by applying the FLC proposed by [Belchior *et al.*, 2012] to the process, and recording along time the values of the input and output variables that constitute the data set. The HGAs are then applied to the obtained data set, with the aim of determining a controller with a response similar to the one that is being replicated. Note that in this simulation, the aim is to learn a controller that replicates the output of an existing FLC, but it could be to learn any other controller, such as for example, to learn to control a process by replicating the control actions sent to a process that is being controlled by a human operator.

Wastewater treatment plants are large and complex nonlinear systems subject to large disturbances in influent flow rate and pollutant load, together with uncertainties concerning the composition of the incoming wastewater [Belchior *et al.*, 2012]. The proposed methodologies, HGA-Control and AHGA-Control, are applied on the control of the dissolved oxygen (DO) in an activated sludge reactor within a WWTP in the Benchmark Simulation Model n.1 (BSM1). BSM1 is a platform-independent simulation environment developed under COST Action 682 and 624 that is dedicated to the optimization of performance and cost-effectiveness of wastewater management systems [Jeppsson and Pons, 2004].

A general overview of the BMS1 plant is presented in Figure 5.3. The biological reactor is distributed over five reactors connected in cascade. Reactors 1 and 2 are non-aerated compartments with a volume of 1000 $[m^3]$ each. Reactors 3, 4, and 5 are aerated and their volumes are approximately equal to 1333 $[m^3]$ each. Reactors

3 and 4 have a fixed oxygen transfer coefficient, and the DO of reactor 5 should be controlled by manipulation of the oxygen transfer rate *KLa5* from an aerator process to the activated sludge inside the biological reactor. The DO concentration is measured on reactor 5 and is controlled by manipulation of *KLa5* on the same reactor. For more details about the BMS1 plant, references [Jeppsson and Pons, 2004; Belchior *et al.*, 2012] are recommended. The sampling period is 15 [min], and each simulation corresponds to a maximum of 14 [days] of elapsed time in the WWTP.

The input/output data set is obtained by controlling the DO concentration with the FLC described in [Belchior *et al.*, 2012], a controller named as FLC-BSM1. The data set was obtained, while controlling the BSM1 plant, by extracting the incremental command signal, $\Delta KLa5(t)$, the tracking error, $E(t)$, of the DO concentration, $E(t) = DO_{ref}(t) - DO(t)$, and the first difference of $E(t)$, $\Delta E(t)$, where $DO_{ref}(t)$ is the desired reference for $DO(t)$. The first four delayed versions of $E(t)$ and $\Delta E(t)$, i.e. $E(t-1), \ldots, E(t-4)$ and $\Delta E(t-1), \ldots, \Delta E(t-4)$, are also included in the learning data set, allowing a better selection of the FLC's input variables. The Levels 2 of the algorithms were configured to employ the variables grouping methodology (Section 4.3.3) as follows. The input variables were divided into two groups: one group for variables $[E(t), E(t-1), E(t-2), E(t-3), E(t-4)]$, and the other group for $[\Delta E(t), \Delta E(t-1), \Delta E(t-2), \Delta E(t-3), \Delta E(t-4)]$. For the AHGA-Control methodology, the number of clusters and the degree of fuzziness were chosen as $N = 20$, and $\eta = 2$, respectively, and for the HGA-Control methodology 5 membership functions were considered for each of the input variables and for the output variable. Also, for both methodologies $X_s = \{E(t), \Delta E(t)\}$ and $g = 5$ were used, and additionally, for the HGA-Control methodology, $\alpha_j = 1$, for all $1 \leqslant j \leqslant n$. The reference signal used to obtain the data set was

$$DO_{ref}(t) = \begin{cases} 1, & 0 < t \leqslant 10\,[\text{days}], \\ 2, & 10\,[\text{days}] < t \leqslant 12\,[\text{days}], \\ 3, & 12\,[\text{days}] < t \leqslant 14\,[\text{days}]. \end{cases} \tag{5.17}$$

Figure 5.4 shows the response obtained with the FLC-BSM1 that was used to construct the data set. Figure 5.5 shows the target response (FLC-BSM1) and the response of the proposed HGA-Control and AHGA-Control methodologies. In Fig-

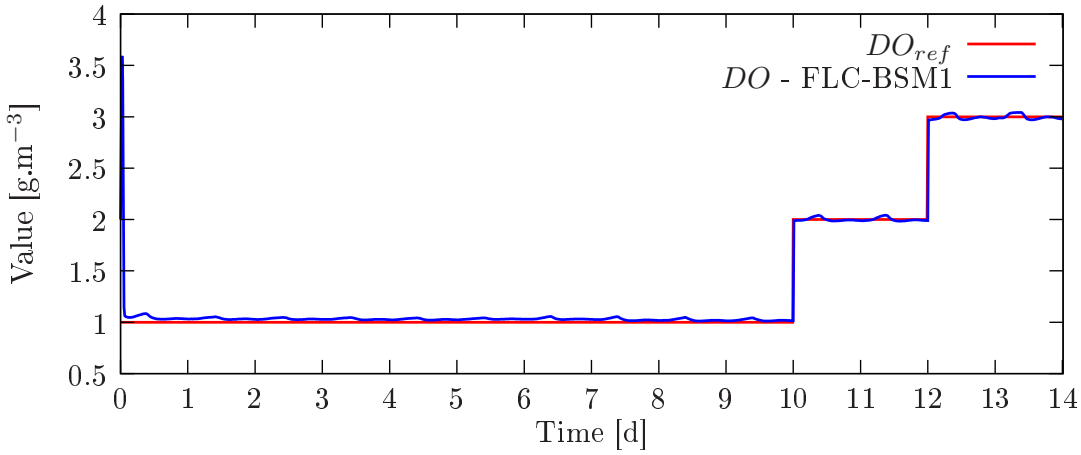


Figure 5.4: Performance of the FLC-BSM1 for the $DO_{ref}(t)$ trajectory (5.17) used to compile the learning data set for the BSM1 plant.

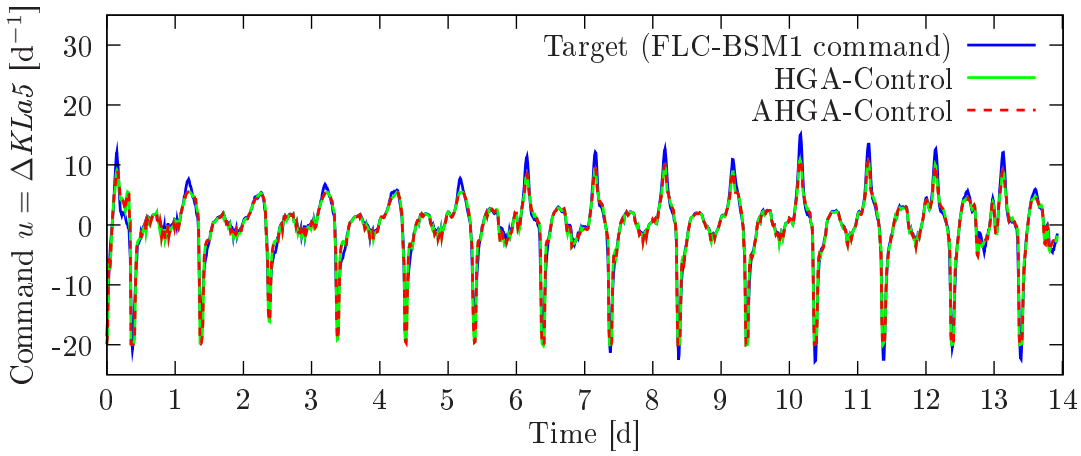


Figure 5.5: FLC target $\Delta KLa5$ commands (FLC-BSM1), and the corresponding command signals learned by the proposed HGA-Control and AHGA-Control methodologies for the BSM1 plant.

ure 5.5 it can be seen that the performed identification was still sufficient to obtain a controller with a response resembling the one that was intended to be achieved. Figure 5.6 shows the time evolution of the fitness function of the proposed HGA-Control and AHGA-Control methodologies. As can be seen in Figure 5.6 and Table 5.1, the AHGA-Control methodology attains faster learning response and better results when compared to the results obtained by the HGA-Control methodology,

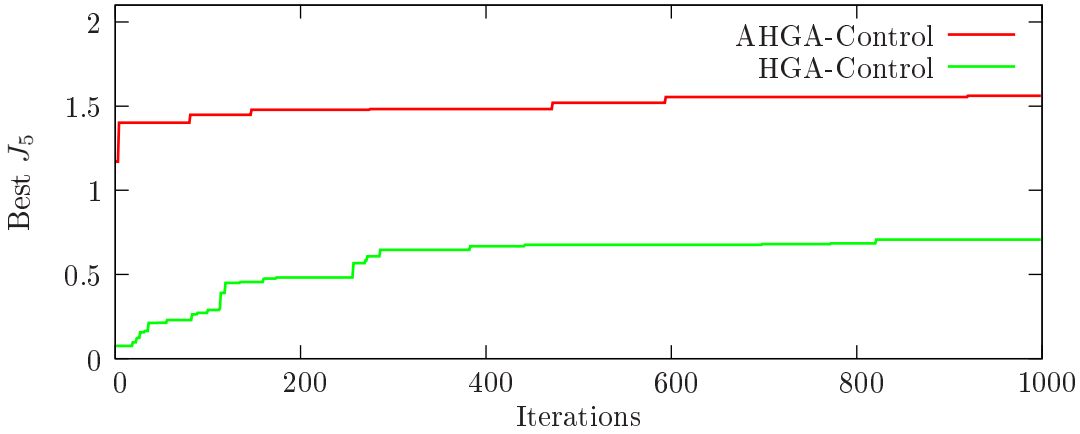


Figure 5.6: Evolution of the best individual fitness function value along the generations for the proposed HGA-Control and AHGA-Control methodologies for the BSM1 plant.

Table 5.1: Comparison of results of the proposed HGA-Control and AHGA-Control methodologies on the BSM1 train data set.

methodology	Number of rules	Number of inputs	Inputs	$1/MSE$
HGA-Control	25	6	$E(t)$, $E(t-1)$, $E(t-4)$, $\Delta E(t-1)$, $\Delta E(t-2)$, $\Delta E(t-4)$	0.6763
AHGA-Control	25	3	$E(t-4)$, $\Delta E(t-3)$, $\Delta E(t-4)$	0.8675

and it can also be observed that the AHGA-Control has the best initialization, performed by the INICONTROL-FCM algorithm, which outperforms the initialization obtained by the INICONTROL algorithm for the HGA-Control, showing the importance of using a good initialization method.

The membership functions and rules obtained by the INICONTROL and INICONTROL-FCM initialization methods are shown in Figure 5.7 (green) and Table 5.2, respectively. In Table 5.2(a), in the antecedent part (E and ΔE) the numbers 1 to 5 represent the linguistic terms of the partition sets of the respective antecedent variables, and in the consequent part ($\Delta KLa5$) the numbers 1 to 25 represent the linguistic terms of the partition set of the consequent variable, for the HGA-Control method. In Table 5.2(b), in the antecedent and consequent parts (E, ΔE, and $\Delta KLa5$) the numbers 1 to 20 represent the linguistic terms of the partition

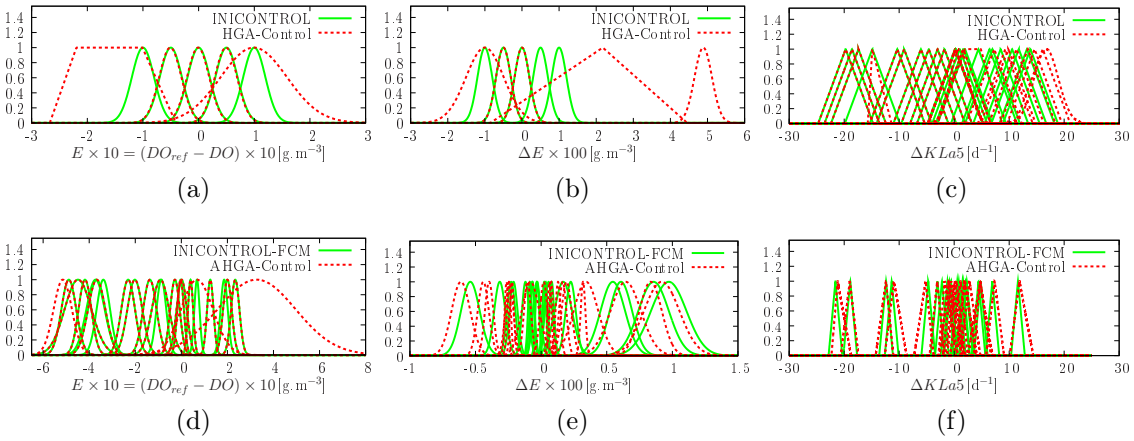
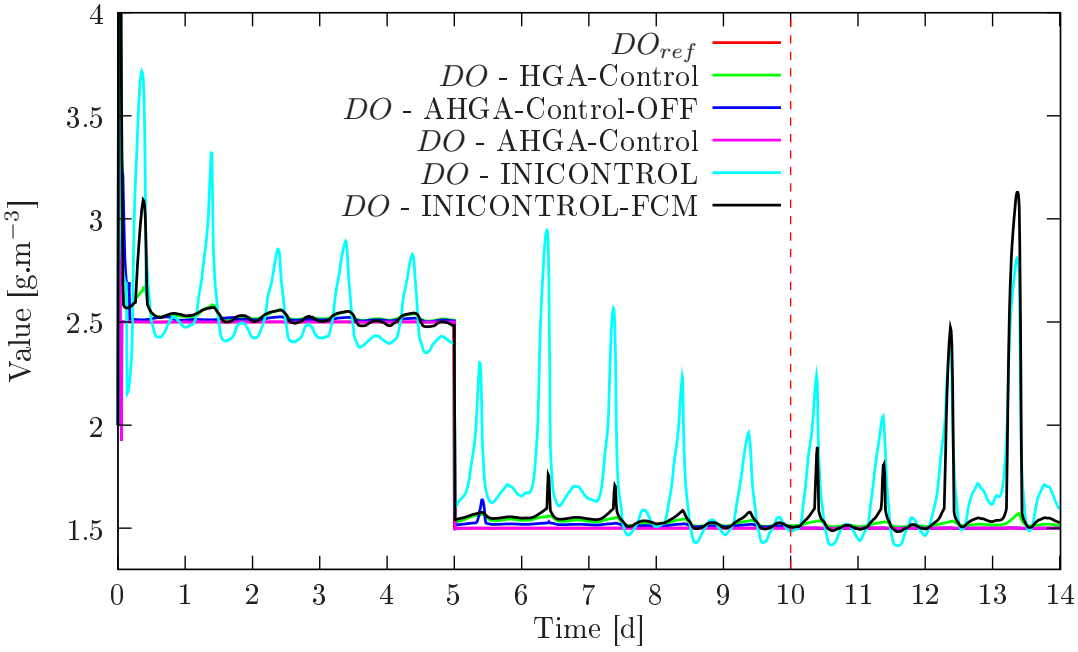


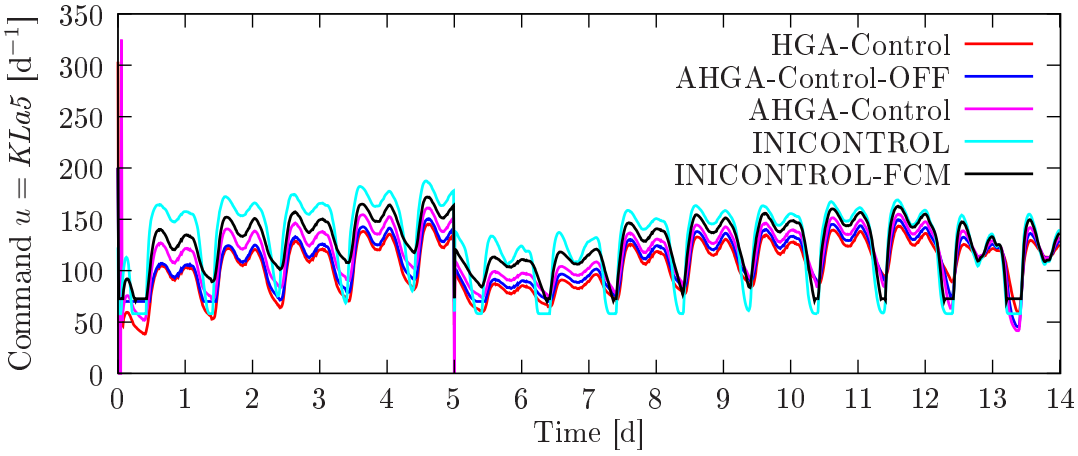
Figure 5.7: Membership functions of the proposed HGA-Control and AHGA-Control methods and of their initialization method (INICONTROL and INICONTROL-FCM, respectively), for all the delayed versions of the HGA-Control input variables (a) E; (b) ΔE, (c) $\Delta KLa5$; and the AHGA-Control input variables (d) E; (e) ΔE, (f) $\Delta KLa5$, for the BSM1 process.

sets of the respective variables, for the AHGA-Control method. The FLC's membership functions and fuzzy rules obtained from the operation of the HGA-Control and AHGA-Control methods are shown in Figure 5.7 (red), and Tables 5.3 and 5.4, respectively. In Tables 5.3 and 5.4 null values indicate the absence of membership function (absence of the corresponding variable in the rule).

Figure 5.8 and Table 5.5 show the results, including the time responses, obtained by the FLCs learned by the HGA-Control and AHGA-Control methodologies, by the respective initialization methods INICONTROL and INICONTROL-FCM, and by AHGA-Control method with the adaptation law turned off until $t = 10\,[\text{days}]$, a method named as AHGA-Control-OFF, for a reference signal different from the one used to generate the data set that was employed for the training, and the respective applied command signals. It can be seen that the proposed controllers are able to adequately (attain and) control the system output at the desired reference $DO_{ref}(t)$. It can also be seen in Figure 5.8 and in Table 5.5, on the AHGA-Control-OFF results that when the AHGA-Control methodology turns on the adaptation law at $t = 10\,[\text{days}]$ (red dashed line in Figure 5.8) the results are improved, and that the performance of AHGA-Control methodology with its adaptation law always



(a)



(b)

Figure 5.8: (a) *DO* results of the proposed HGA-Control and AHGA-Control methodologies, of the respective initialization methods INICONTROL and INICONTROL-FCM, and of the AHGA-Control method with the adaptation law turned off until $t = 10$ [days], a method named as AHGA-Control-OFF, for a reference different from the one used for obtaining the BSM1 data set used for training the HGA methodologies; and (b) the respective applied *KLa5* command signals. The red dashed line represents the time when the AHGA-Control methodology turns on the adaptation law (at $t = 10$ [days]), for the AHGA-Control-OFF case.

Table 5.2: Fuzzy rule structure obtained with the initialization methods: (a) INICONTROL (used in HGA-Control); and (b) INICONTROL-FCM (used in AHGA-Control), for the BSM1 process.

(a)

$E(t)$	$\Delta E(t)$	$\Delta KLa5(t)$
1	1	1
1	2	2
1	3	3
1	4	4
1	5	5
2	1	6
2	2	7
2	3	8
2	4	9
2	5	10
3	1	11
3	2	12
3	3	13
3	4	14
3	5	15
4	1	16
4	2	17
4	3	18
4	4	19
4	5	20
5	1	21
5	2	22
5	3	23
5	4	24
5	5	25

(b)

$E(t)$	$\Delta E(t)$	$\Delta KLa5(t)$
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12
13	13	13
14	14	14
15	15	15
16	16	16
17	17	17
18	18	18
19	19	19
20	20	20

turned on is better than AHGA-Control-OFF, concluding that the performance of the learned FLC can be improved using the adaption law. The results obtained with the HGA-Control and AHGA-Control methodologies have a response closer to the reference signal than the results obtained by the respective INICONTROL and INICONTROL-FCM initialization methods. This fact is also supported by the results of the time evolution of the respective fitness function of the proposed methods presented in Figure 5.6.

Table 5.3: Fuzzy rule structure obtained/learned by the HGA-Control method for the BSM1 process.

Selected FLC inputs						FLC output
$E(t)$	$E(t-1)$	$E(t-4)$	$\Delta E(t-1)$	$\Delta E(t-2)$	$\Delta E(t-4)$	$\Delta KLa5(t)$
2	0	0	2	0	0	7
5	1	2	5	3	4	19
2	0	0	3	0	0	13
2	0	0	5	0	0	10
2	0	0	2	0	0	12
4	3	2	2	0	0	17
3	0	0	3	0	0	13
2	3	0	4	0	0	19
4	0	0	4	0	0	19
2	0	0	3	0	0	13
1	0	0	5	0	0	5
1	0	0	5	0	0	5
2	3	0	4	0	0	19
5	0	0	1	0	0	21
2	3	0	3	0	0	13
3	0	0	3	0	0	13
3	0	0	2	0	0	12
4	0	0	2	0	0	17
1	0	0	5	0	0	5

5.5.2 Real-World Control of Two Coupled DC Motors

The two coupled DC motors process is the same of the one described in Subsection 4.6.3. The input/output data set is obtained by controlling the velocity of the DC motors process with the controller described in [Mendes *et al.*, 2013a], an adaptive fuzzy generalized predictive control named as Adaptive Fuzzy Generalized Predictive Control (AFGPC). The data set was obtained, while controlling the DC motor, by extracting the incremental command signal, $\Delta u(k)$, the tracking error of the DC motor velocity, $E(k) = r(k) - y(k)$, and the first difference of $E(k)$, $\Delta E(k)$, where $r(k)$ is the desired reference for the DC motor velocity. The first four delays of $E(k)$, and $\Delta E(k)$, i.e. $E(k-1), \ldots, E(k-4)$ and $\Delta E(k-1), \ldots, \Delta E(k-4)$, are also included in the learning data set, allowing a better selection of the FLC's input variables. The Levels 2 of the algorithms were configured to employ the variables grouping methodology (Section 4.3.3) as follows. The input variables were divided into two groups: one group for variables $[E(k), E(k-1), E(k-2), E(k-3), E(k-4)]$ and the other group for $[\Delta E(k), \Delta E(k-1), \Delta E(k-2), \Delta E(k-3), \Delta E(k-4)]$. For the

Table 5.4: Fuzzy rule structure obtained/learned by the AHGA-Control method for the BSM1 process.

Selected FLC inputs			FLC output
$E(t-4)$	$\Delta E(t-3)$	$\Delta E(t-4)$	$\Delta KLa5(t)$
12	0	0	12
6	0	0	6
10	0	0	10
18	0	0	18
6	0	0	6
6	0	0	6
3	0	0	3
1	0	0	1
3	0	0	3
9	0	0	9
2	0	0	2
19	0	0	19
1	4	16	11
11	0	0	11
7	0	0	7
4	0	0	4
18	0	0	18
7	0	0	7
4	0	0	4
4	0	0	4
4	0	0	4
5	0	0	5
20	0	0	20
16	0	0	16
18	0	0	18

AHGA-Control methodology, the number of clusters and the degree of fuzziness were chosen as $N = 20$, and $\eta = 2$, respectively, and for the HGA-Control methodology 5 membership functions were considered for each of the input variables and for the output variable. Also, for both methodologies $X_s = \{E(t), \Delta E(t)\}$ and $g = 5$ were used, and additionally, for the HGA-Control methodology, $\alpha_j = 1$, for all $1 \leqslant j \leqslant n$.

Table 5.5: Comparison of results of the proposed HGA-Control and AHGA-Control methodologies, of the respective initialization methods INICONTROL and INICONTROL-FCM, and of the AHGA-Control method with the adaptation law turned off until $t = 10\,[\text{days}]$, a method named as AHGA-Control-OFF, for the BSM1 test. Note that the $MSE = \frac{1}{T}\sum_{t=1}^{T}(DO_{ref}(t) - DO(t))^2$ is related with the tracking error instead of estimation error that is used in the train. T is the total number of sample on the test.

Methodology	$1/MSE = 1/(\frac{1}{T}\sum_{t=1}^{T}(DO_{ref}(t) - DO(t))^2)$
INICONTROL	10.459
INICONTROL-FCM	26.341
HGA-Control	503.61
AHGA-Control-OFF	740.33
AHGA-Control	1335.1

The reference signal used to obtain the data set was

$$r(k) = \begin{cases} 110, & 0 < k \leqslant 115, \\ 155, & 115 < k \leqslant 235, \\ 95, & 235 < k \leqslant 355, \\ 140, & 355 < k \leqslant 475, \\ 125, & 475 < k \leqslant 595, \\ 115, & 595 < k \leqslant 700. \end{cases} \tag{5.18}$$

Figure 5.9 shows the response obtained with the AFGPC proposed in [Mendes *et al.*, 2013a] that was used to construct the data set. Figure 5.10 shows the target command response (AFGPC), and the command responses obtained by the proposed HGA-Control and AHGA-Control methodologies. Table 5.6 presents the numbers of rules and inputs, and the selected input variables and the MSE that resulted from the application of the HGA-Control and AHGA-Control methodologies. As can be seen in Figure 5.10 and in Table 5.6 the responses of the FLCs attained by the proposed HGA-Control and AHGA-Control methodologies are good approximations to the target command signals. It is also verified that the result obtained with the AHGA-Control methodology has a response closer to the target command signal when compared to the result obtained by the HGA-Control methodology. This is

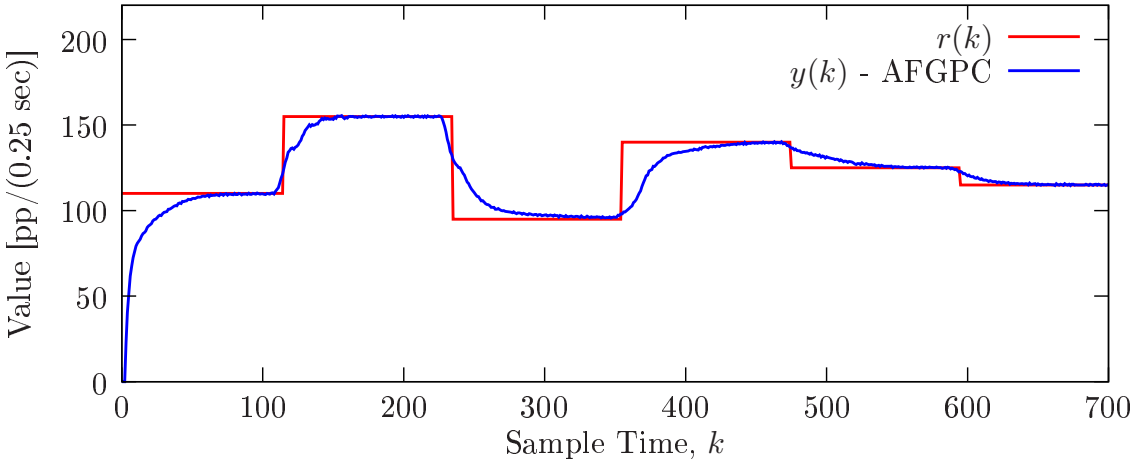


Figure 5.9: Performance of the AFGPC for the $r(k)$ trajectory (5.18) used to compile the learning data set of the DC motor process.

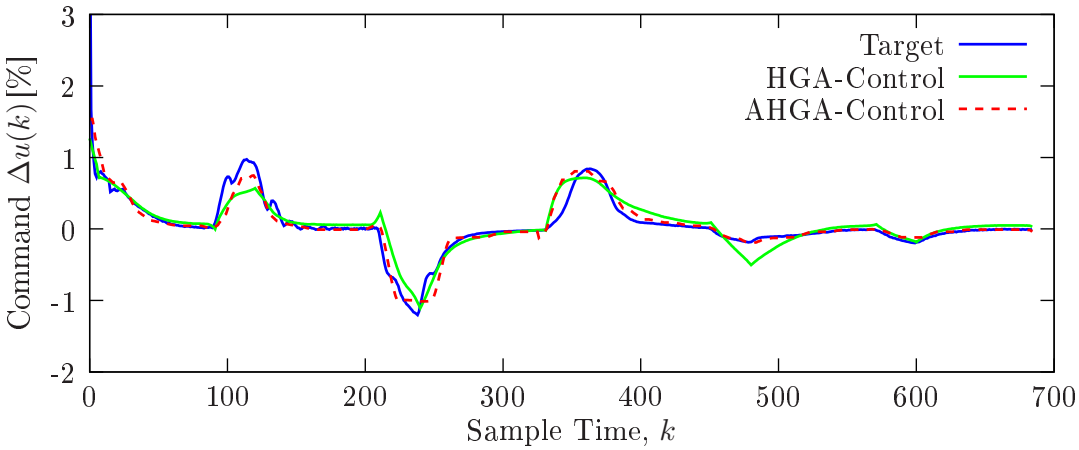


Figure 5.10: AFGPC target command signal, and command signals learned by the proposed HGA-Control and AHGA-Control methodologies on the DC motor process.

also supported by the results presented in Figure 5.11 which show the time evolutions of the fitness functions for the HGA-Control and AHGA-Control methodologies.

The membership functions and the rules obtained by the INICONTROL and INICONTROL-FCM initialization methods are shown in Figure 5.12 (green) and Table 5.7, respectively. In Table 5.7(a), in the antecedent part (E and ΔE) the numbers 1 to 5 represent the linguistic terms of the partition sets of the respective antecedent variables, and in the consequent part (Δu) the numbers 1 to 25 represent

Table 5.6: Comparison of results of the proposed HGA-Control and AHGA-Control methodologies on the Motor training data set.

methodology	Number of rules	Number of inputs	Inputs	$1/MSE$
HGA-Control	16	4	$E(k)$, $E(k-2)$, $\Delta E(k-1)$, $\Delta E(k-2)$	24.56
AHGA-Control	17	6	$E(k)$, $E(k-1)$, $E(k-4)$, $\Delta E(k)$, $\Delta E(k-2)$, $\Delta E(k-4)$	27.02

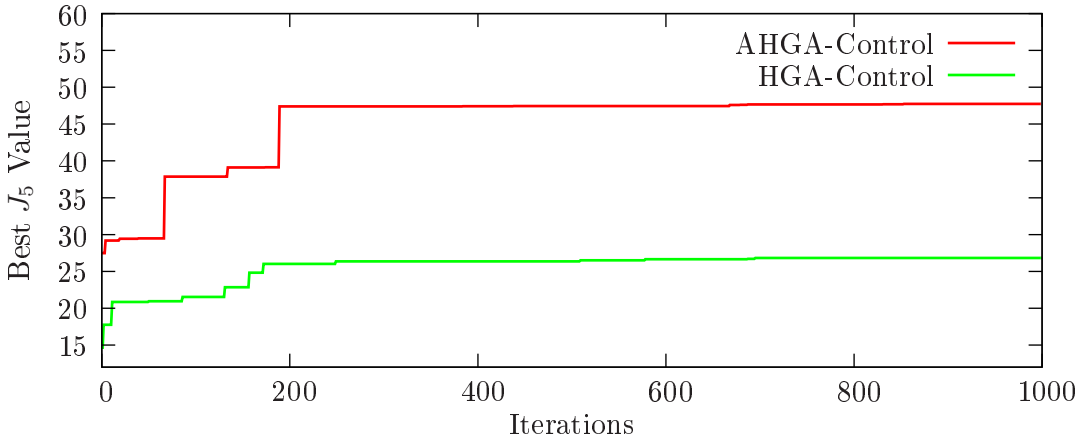


Figure 5.11: Evolution of the best individual fitness function value along the generations for the proposed HGA-Control and AHGA-Control methodologies for the DC motors process.

the linguistic terms of the partition set of the consequent variable, for the HGA-Control method. In Table 5.7(b), in the antecedent and consequent parts (E, ΔE, and Δu) the numbers 1 to 20 represent the linguistic terms of the partition sets of the respective variables, for the AHGA-Control method. The FLC's membership functions and fuzzy rules obtained from the operation of the HGA-Control and AHGA-Control methods are shown in Figure 5.12 (red), and Tables 5.8 and 5.9, respectively. In Tables 5.8 and 5.9, null values indicate the absence of membership function (absence of the corresponding variable in the rule).

Figure 5.13 and Table 5.10 compare the results, including the time responses, obtained by the FLCs evolved by the proposed HGA-Control and AHGA-Control methodologies, by the respective initialization methods INICONTROL and INICONTROL-FCM, and by the AHGA-Control method with the adaptation law

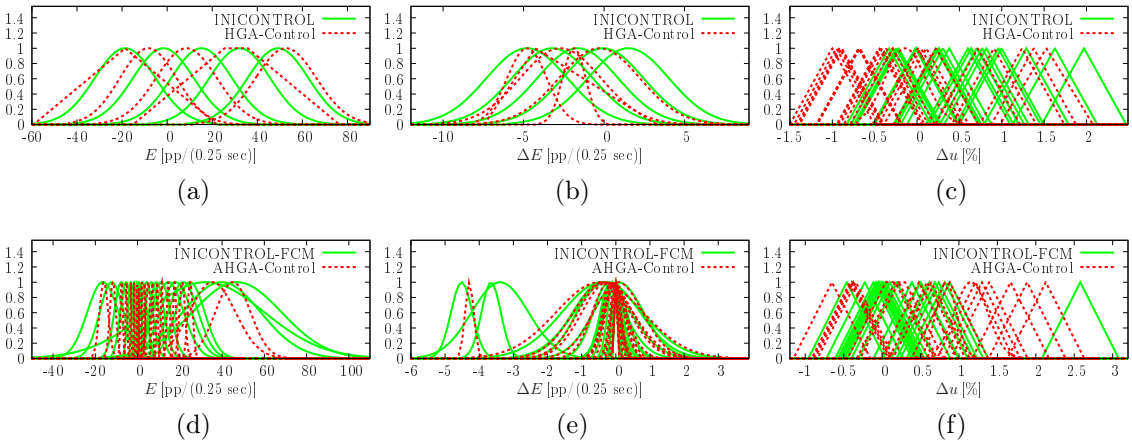


Figure 5.12: Membership functions of the proposed HGA-Control and AHGA-Control methods and of their initialization methods (INICONTROL and INICONTROL-FCM, respectively), for all the delayed versions of the HGA-Control input variables (a) E; (b) ΔE, (c) Δu; and the AHGA-Control input variables (d) E; (e) ΔE, (f) Δu, for the DC Motor process.

turned off until $k = 880$, a method named as AHGA-Control-OFF, for a reference signal different from the one used to generate the data set that was employed for training. It can be seen that the proposed controllers are able to adequately (attain and) control the system output at the desired reference $r(k)$. A load disturbance is applied by switching on the lamps for $360 \leqslant k \leqslant 680$. When the load disturbance is applied, there is an undershoot at $k = 360$ and an overshoot at $k = 680$ in the system responses. As can be seen, the controllers eliminate this disturbance. It can also be seen in Figure 5.13 and in Table 5.10 that the performance of AHGA-Control methodology with its adaptation law always turned on is better than AHGA-Control-OFF, concluding that the performance of the learned FLC can be improved using the adaption law. The results obtained with the HGA-Control and AHGA-Control methodologies have a response closer to the reference signal than the result obtained by the respective initialization methods INICONTROL and INICONTROL-FCM, a fact also supported by the results of the time evolution of the respective fitness function of the proposed methods presented in Figure 5.11.

Table 5.7: Fuzzy rule structure obtained with the initialization methods: (a) INICONTROL (used in HGA-Control); and (b) INICONTROL-FCM (used in AHGA-Control), for the DC Motor process.

(a)

$E(k)$	$\Delta E(k)$	$\Delta u(k)$
1	1	1
1	2	2
1	3	3
1	4	4
1	5	5
2	1	6
2	2	7
2	3	8
2	4	9
2	5	10
3	1	11
3	2	12
3	3	13
3	4	14
3	5	15
4	1	16
4	2	17
4	3	18
4	4	19
4	5	20
5	1	21
5	2	22
5	3	23
5	4	24
5	5	25

(b)

$E(k)$	$\Delta E(k)$	$\Delta u(k)$
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12
13	13	13
14	14	14
15	15	15
16	16	16
17	17	17
18	18	18
19	19	19
20	20	20

5.6 Conclusion

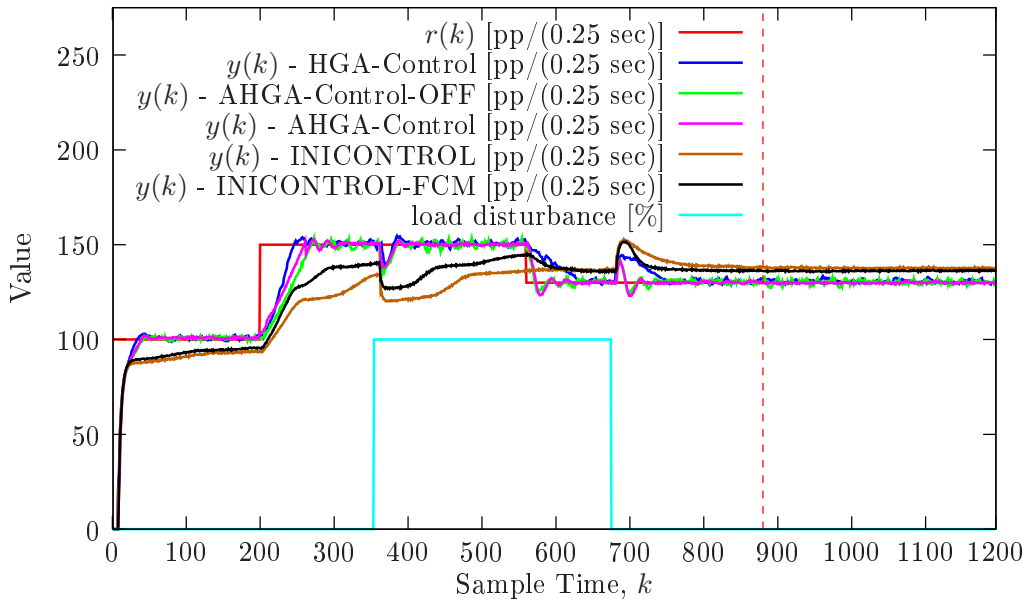
This chapter proposed two methodologies to automatically extract all fuzzy parameters and design the structure of a FLC in order to control nonlinear processes. The learning of the FLC is performed by the HGAs, using a set of input/output data, previously extracted from a process under control (e.g. it can be extracted from a process under manual control). These methodologies do not require any prior knowledge concerning the fuzzy rule structure, location or shape of member-

Table 5.8: Fuzzy rule structure obtained/learned by the HGA-Control method for the DC Motor process.

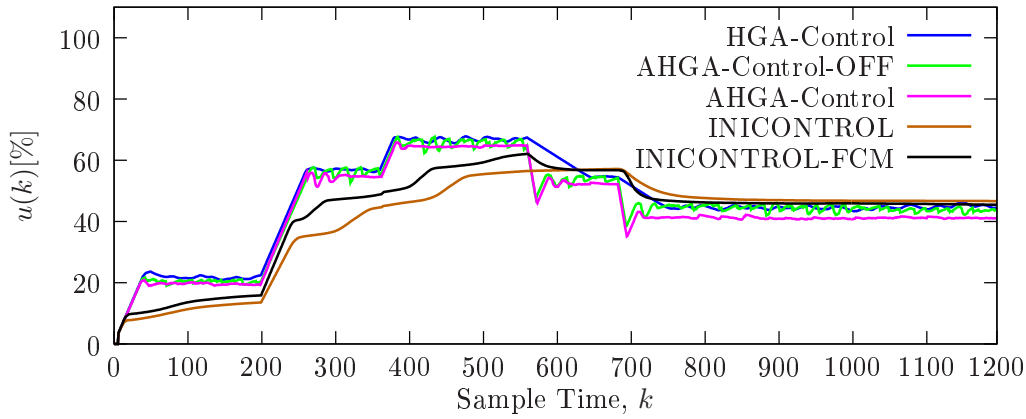
Selected FLC inputs				FLC output
$E(k-1)$	$E(k-2)$	$\Delta E(k-1)$	$\Delta E(k-2)$	$\Delta u(k)$
4	0	0	2	17
1	0	0	1	1
3	0	4	2	22
5	0	0	1	21
1	3	5	5	5
1	0	4	2	7
3	0	0	5	15
3	4	0	2	7
3	0	0	2	17
3	0	0	2	12
1	0	0	5	5
4	0	0	2	22
2	0	0	2	7
5	0	0	5	25
2	0	5	2	12
4	0	0	5	17

Table 5.9: Fuzzy rule structure obtained/learned by the AHGA-Control method for the DC Motor process.

Selected FLC inputs						FLC output
$E(k)$	$E(k-1)$	$E(k-4)$	$\Delta E(k)$	$\Delta E(k-2)$	$\Delta E(k-4)$	$\Delta u(k)$
19	0	0	19	0	0	19
14	0	0	14	0	0	14
6	11	0	11	0	0	11
16	0	0	16	0	0	16
8	0	0	8	0	0	8
15	8	12	13	9	11	5
6	0	0	6	0	0	6
13	0	0	13	0	0	13
2	10	17	8	17	1	3
2	0	0	15	0	0	15
15	0	0	15	0	0	15
18	0	0	18	0	0	18
20	0	0	20	0	0	20
7	0	0	7	0	0	7
1	0	0	1	0	0	1
5	0	3	3	0	0	3
10	0	0	10	0	0	10



(a)



(b)

Figure 5.13: (a) Results of the proposed HGA-Control and AHGA-Control methodologies, of the respective INICONTROL and INICONTROL-FCM initialization methods, and of the AHGA-Control method with the adaptation law turned off until $k = 880$, a method named as AHGA-Control-OFF, for a reference signal different from the one used for obtaining the DC motors data set used for training, and in the presence of load disturbances (lamps switched on for $360 \leqslant k \leqslant 680$) on the DC motors process; and (b) the respective applied command signals. The red dashed line represents the time when the AHGA-Control methodology turns on the adaptation law (at $k = 880$), for the AHGA-Control-OFF case.

Table 5.10: Comparison of results of the proposed HGA-Control and AHGA-Control methodologies, of the respective initialization methods INICONTROL and INICONTROL-FCM, and of the AHGA-Control method with the adaptation law turned off until $k = 880$, a method named as AHGA-Control-OFF, for the DC motors process test. Note that the $MSE = \frac{1}{T}\sum_{k=1}^{T}(r(k) - y(k))^2$ is related with the tracking error instead of estimation error that is used in the train. T is the total number of sample on the test.

Methodology	$1/MSE = 1/(\frac{1}{T}\sum_{k=1}^{T}(r(k) - y(k))^2)$
INICONTROL	3.3×10^{-3}
INICONTROL-FCM	5.6×10^{-3}
HGA-Control	14.4×10^{-3}
AHGA-Control-OFF	12.5×10^{-3}
AHGA-Control	15.3×10^{-3}

ship functions, implication and aggregation operators, defuzzification methods, or selection of adequate input variables and corresponding time delays.

The main purpose of the proposed HGAs is to develop a FLC with a response similar to the one used to compile the data set, or in less successful attempts, to develop a controller which constitutes a starting point for further adjustments. In order to obtain a better control error, if necessary the proposed algorithm could be easily applied to initialize the required fuzzy knowledge-base of adaptive controllers. Additionally, the methodologies may also be used to understand a process for which there is little or no information available, since they automatically extract all fuzzy parameters, and they are able to gather a knowledge-base about the process control.

Two methodologies have been proposed in this chapter. First, a HGA for control which uses the INICONTROL initialization method based on [Andersen *et al.*, 1997], and is named as HGA-Control (Section 5.4.1, Algorithm 5.4). Second, it was proposed a HGA, named as AHGA-Control, which uses the INICONTROL-FCM initialization method based on [Andersen *et al.*, 1997] and on a fuzzy *c*-means (FCM) clustering algorithm [Celikyilmaz and Trksen, 2009; Dovžan and Škrjanc, 2011], and where to improve the results of the learned FLC, a direct adaptive fuzzy controller used in [Mendes *et al.*, 2011] was applied (Subsection 5.4.2, Algorithm 5.5).

The proposed methodologies were studied and applied in the control of the dissolved oxygen in an activated sludge reactor within a simulated wastewater treat-

ment plant (WWTP), and in the control of the velocity on a real-world experimental setup composed of two coupled DC motors. The results have shown that the proposed methodologies extracted all the parameters of the FLC, and the extracted FLC was able to control the processes with success.

Thus, the aim of proposing methodologies to automatically extract all fuzzy parameters of a FLC in order to control nonlinear processes, by using data previously extracted from the process under control, and without any prior knowledge about the control of the process, was reached.

Chapter 6

Fuzzy Predictive Control

Contents

6.1 Introduction / State of the Art 123

6.2 Fuzzy Predictive Control Frameworks 125

6.2.1 Modelling Using T-S Fuzzy Models 127

6.2.2 Multiple Models 128

6.2.3 Predictive Control Law 129

6.2.4 Fuzzy Predictive Control Framework Algorithms 133

6.3 Experimental Results 134

6.3.1 Continuous-Stirred Tank Reactor 134

6.3.2 Real-World Control of Two Coupled DC Motors 139

6.4 Conclusion 142

6.1 Introduction / State of the Art

Model predictive control (MPC) is a popular control approach that is based on the use of a model of the process to predict the future behavior of the system over a prediction horizon. MPC is widely used in practice due to its high-quality control performance. One of the most popular and powerful MPC methods applied in industry has been the generalized predictive control (GPC) [Camacho and Bordons,

2007]. The GPC has been applied in various plants, and has shown good performance results [Clarke, 1988; Tham *et al.*, 1991] using linear plant models. However, the majority of physical systems contain complex nonlinear relations, which are difficult to model with conventional techniques. The results can be improved using an algorithm based on a nonlinear model. Furthermore, a disadvantage of GPC, as commonly in MPCs, is its assumption of the knowledge of an accurate model of the process to be controlled.

As already mentioned in this thesis, the assumption of the knowledge of an accurate model in MPCs, presents problems because many complex plants are difficult to be mathematically modelled based on physical laws, or have large uncertainties and strong nonlinearities. A suitable option, is the application of models based on fuzzy logic systems, which is theoretically supported by the fact that fuzzy logic systems are universal approximators [Wang and Mendel, 1992; Kosko, 1994].

In [Zhao *et al.*, 2010] it is proposed a methodology for automatically extracting T-S fuzzy models from data using particle swarm optimization. The structures and parameters of the fuzzy models are encoded into particles and evolve together so that the optimal structure and parameters can be achieved simultaneously. In [Yusof *et al.*, 2011], a technique for the modeling of nonlinear control processes using a fuzzy modeling approach based on the T-S fuzzy model with a combination of a genetic algorithm and the recursive least squares method is proposed. In [Kayadelen, 2011] the potential of genetic expression programming and an adaptive neuro-fuzzy computing paradigm is studied to forecast the safety factor for liquefaction of soils. In [Han *et al.*, 2012] a self-organizing radial basis function neural network model predictive control method is proposed for controlling the dissolved oxygen concentration in a wastewater treatment process. In [Wu *et al.*, 2012] a GPC strategy with closed-loop model identification for burn-through point control in the sintering process is proposed. In [Su *et al.*, 2012] an automatic methodology to extract T-S fuzzy models with enhanced performance from data is proposed. The idea of variable length genotypes is introduced to the artificial bee colony (ABC). The Fuzzy C-Means clustering technique based on the ABC algorithm is studied. In [Li *et al.*, 2013] a type-2 fuzzy method based on a data-driven strategy for the modeling and optimization of the thermal comfort and energy concumption at smart homes or intelligent buildings is presented. [Ren *et al.*, 2013] presents a fuzzy modelling method for modeling cutting forces based on subtractive clustering. The subtractive clus-

tering is used to partition the input space and extract a set of fuzzy rules, and then a least squares algorithm is used to find the optimal membership functions along with the consequent parameters of the rule base.

The methods of [Kayadelen, 2011; Han *et al.*, 2012; Wu *et al.*, 2012; Su *et al.*, 2012; Li *et al.*, 2013; Ren *et al.*, 2013], have the limitation of not being able to perform automatic selection of variables and delays: pre-selection is performed. The selection of the most adequate input variables and respective time delays is crucial since the use of the correct variables with the correct delays can lead to better prediction accuracy because they can contain more information about the output than incorrect variables and/or variables with incorrect delays [Souza *et al.*, 2013]. In [Zhao *et al.*, 2010; Yusof *et al.*, 2011] the employed pre-selection processes do not take advantage of taking into account the prediction models being learned.

In this chapter the problem of the assumption of knowledge about an accurate model of the process in MPC control architectures will be addressed. The methodologies proposed in Chapter 4 to automatically identify a T-S fuzzy model from input/output data to approximate unknown nonlinear processes will be integrated with, and used to learn the prediction model of, the generalized predictive control (GPC) algorithm. Two methodologies are proposed. There is the possibility of learning models offline and/or online. It is a fact that, in off-line training algorithms, the discrete-time T-S fuzzy model can be obtained from input-output data collected from a plant. However, such collected data set can be limited and the obtained T-S fuzzy model may not provide adequate accuracy in parts or the whole operating areas of the plant. Moreover, the behavior and model of the plant may be changing over time. This motivates the introduction of adaptive methodologies to solve these problems. Thus, another characteristic of one of the control methodologies proposed in this chapter is that, the fuzzy model adapts itself to new process conditions in order to maintain the quality of the identification/control even in situations such as for example when dealing with nonlinear plants, time-varying processes, disturbances or varying operating regions or varying parameters of the model.

6.2 Fuzzy Predictive Control Frameworks

In this chapter two fuzzy predictive control frameworks are proposed. The T-S fuzzy models learned by the identification methods proposed in Chapter 4 are integrated

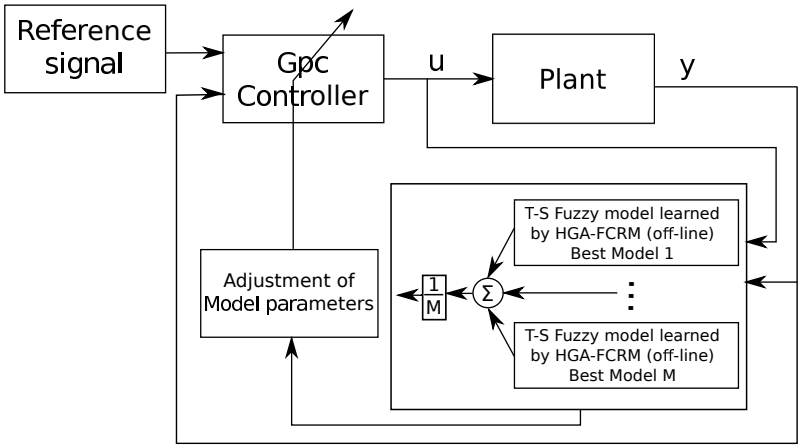


Figure 6.1: A generic schematic diagram of the proposed FMMGPC control architecture.

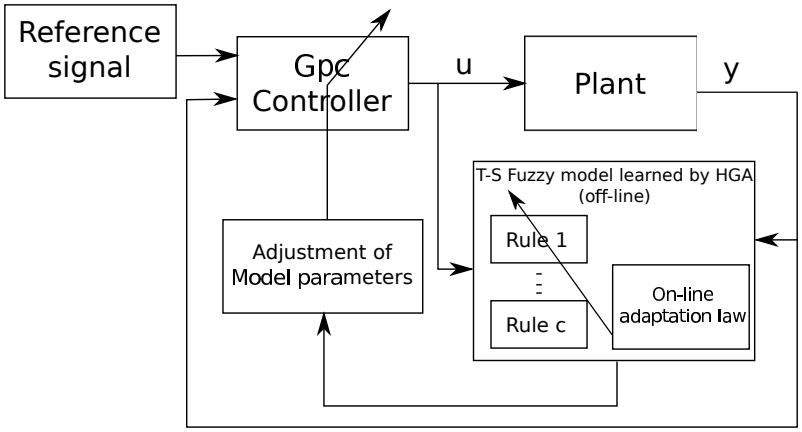


Figure 6.2: A generic schematic diagram of the proposed AFGPC control architecture.

in generalized predictive control (GPC). The first framework, the Fuzzy Multiple Models Generalized Predictive Control (FMMGPC) approach, is based on a GPC controller used in conjunction with a combination of multiple T-S fuzzy models learned by the identification methods proposed in Chapter 4. The second framework is based on the integration of a T-S fuzzy model learned by the identification methods proposed in Chapter 4, into an adaptive fuzzy GPC (AFGPC) controller.

A diagram of the FMMGPC approach is presented in Figure 6.1, and a diagram of the AFGPC approach is depicted in Figure 6.2. As can be seen in both diagrams, the

control scheme consists of the plant to be controlled, the controller that is composed of a model based predictive controller, namely the GPC, and a T-S fuzzy model. In the FMMGPC control architecture, the GPC controller integrates a combination of multiple T-S fuzzy models, learned off-line, according to the methodology presented in Chapter 4. In the AFGPC control architecture, the GPC controller integrates a T-S fuzzy model, learned off-line, according to the methodology presented in Chapter 4, and the consequent model parameters are adjusted on-line by the adaptation law studied in Section 4.4.3.

6.2.1 Modelling Using T-S Fuzzy Models

A large class of nonlinear processes can be represented by a model of the following type:

$$y(k) = f[y(k-1),\ldots,y(k-n_y),u(k-d-1),\ldots,u(k-d-n_u)], \quad (6.1)$$

where $u(\cdot):\ \mathbb{N}\to\mathbb{R}$, and $y(\cdot):\mathbb{N}\to\mathbb{R}$ are the process input and output, respectively, $n_u\in\mathbb{N}$, and $n_y\in\mathbb{N}$ are the orders of input and output, respectively, $d\in\mathbb{N}$ is the dead-time, and $d+1$ is the time-delay of the system. In the discrete-time nonlinear SISO plant (6.1), $f(\cdot):\mathbb{R}^{n_y+n_u}\to\mathbb{R}$ represents a nonlinear mapping which is assumed to be unknown. $f(\cdot)$ will be approximated by a T-S fuzzy system.

For the GPC controller, system (6.1) can be described by a T-S fuzzy model defined by the following fuzzy rules:

$$\begin{aligned} R_i:\ & \text{IF } x_1(k) \text{ is } A_1^i, \text{ and } \ldots \text{ and } x_n(k) \text{ is } A_n^i \\ & \text{THEN } y_i(k) = a_i(z^{-1})y(k-1) + b_i(z^{-1})u(k-d-1), \\ & \quad i=1,\ldots,N, \end{aligned} \quad (6.2)$$

where N is the number of rules, and $n=n_y+n_u$,

$$\begin{aligned} a_i(z^{-1}) &= a_{1i} + a_{2i}z^{-1} + \ldots + a_{n_yi}z^{-(n_y-1)}, \\ b_i(z^{-1}) &= b_{1i} + b_{2i}z^{-1} + \ldots + b_{n_ui}z^{-(n_u-1)}, \end{aligned} \quad (6.3)$$

and $u(k)$ is the control output (the command). $\mathbf{x}^T(k) = [x_1(k),\ldots,x_n(k)] = [y(k-1),\ldots,y(k-n_y),u(k-d-1),\ldots,u(k-d-n_u)]$ is the vector of input variables of the T-S fuzzy system. Considering a collection of M models of type (6.2),

the h-th such model can be written from (6.2) as

$$\begin{aligned} y^h(k) &= \sum_{i=1}^{N} \bar{\omega}_i^h[\mathbf{x}(k)] \left[a_i^h(z^{-1})y(k-1) + b_i^h(z^{-1})u(k-d-1)\right], \\ &= \sum_{i=1}^{N} \bar{\omega}_i^h[\mathbf{x}(k)]\mathbf{x}^T(k)\boldsymbol{\theta}_i^h, \\ &= [\boldsymbol{\Psi}^h(k)]^T\boldsymbol{\Theta}^h, \end{aligned} \tag{6.4}$$

for $h = 1, \ldots, M$, where superscript h is used to denote variables and parameters of model h, and for $i = 1, \ldots, N$,

$$\bar{\omega}_i^h[\mathbf{x}(k)] = \frac{\prod_{j=1}^{n} \mu_{A_j^{i^h}}(x_j(k))}{\sum_{p=1}^{N}\prod_{j=1}^{n} \mu_{A_j^{p^h}}(x_j(k))}, \tag{6.5}$$

$$\boldsymbol{\theta}_i^h = \left[a_{1i}^h, \ldots, a_{n_y i}^h, b_{1i}^h, \ldots, b_{n_u i}^h\right]^T, \tag{6.6}$$

$$\boldsymbol{\Theta}^h = \left[(\boldsymbol{\theta}_1^h)^T, (\boldsymbol{\theta}_2^h)^T, \ldots, (\boldsymbol{\theta}_N^h)^T\right]^T, \tag{6.7}$$

$$\boldsymbol{\Psi}^h(k) = \left[\left(\bar{\omega}_1^h[\mathbf{x}(k)]\right)\mathbf{x}^T(k), \ldots, \left(\bar{\omega}_N^h[\mathbf{x}(k)]\right)\mathbf{x}^T(k)\right]^T. \tag{6.8}$$

6.2.2 Multiple Models

Multiple T-S fuzzy models are combined on the GPC controller to improve the identification performance. The reason for using multiple T-S fuzzy models, instead of one, it is to reduce the uncertainty associated with the model parameters determined by the proposed hierarchical genetic algorithms fuzzy identification. The parameters of each of the T-S fuzzy models will be learned using the proposed hierarchical genetic algorithms fuzzy identification in Chapter 4. The output is the average of the predictions of the individual T-S fuzzy models.

By taking the output as the average of the predictions, the uncertainty associated with the model parameters is reduced, [Bishop, 2006]. In [Bishop, 2006], it is shown that the expected error of multiple models will not exceed the expected error of the individual models, so that $\mathrm{E}_{COM} \leqslant \mathrm{E}_{AV}$, where $\mathrm{E}_{AV} = \frac{1}{M}\sum_{h=1}^{M} \mathbb{E}_{\mathbf{x}}\left[(e^h(\mathbf{x}))^2\right]$ is the average squared errors made by the M models acting individually, $\mathbb{E}_{\mathbf{x}}$ is expectation operator, $\mathrm{E}_{COM} = \mathbb{E}_{\mathbf{x}}\left[\left(\frac{1}{M}\sum_{h=1}^{M} e^h(\mathbf{x})\right)^2\right]$ is the average error of the multiple models working together, $e^h(\mathbf{x}) = y_{real}(\mathbf{x}) - y_{est}(\mathbf{x})$, $y_{est}(\mathbf{x})$ is the model

output, and $y_{real}(\mathbf{x})$ is the real output of the system to be modeled.

Using M models and (6.4), the output of the model used on GPC is given by:

$$y(k) = \frac{1}{M}\sum_{h=1}^{M} y^h(k), \tag{6.9}$$

which can be rewritten as follows:

$$\bar{a}(z^{-1})y(k) = \bar{b}(z^{-1})u(k-d-1), \tag{6.10}$$

where

$$\bar{a}(z^{-1}) = 1 - \bar{a}_1 z^{-1} - \ldots - \bar{a}_{nY} z^{-nY}, \tag{6.11}$$

$$\bar{b}(z^{-1}) = \bar{b}_1 + \bar{b}_2 z^{-1} + \ldots + \bar{b}_{nU} z^{-(nU-1)}, \tag{6.12}$$

and

$$\bar{a}_s = \frac{1}{M}\sum_{h=1}^{M}\sum_{i=1}^{N} \bar{\omega}_i^h[\mathbf{x}(k)]a_{si}^h, \;\; s = 1, \ldots, nY, \tag{6.13}$$

$$\bar{b}_m = \frac{1}{M}\sum_{h=1}^{M}\sum_{i=1}^{N} \bar{\omega}_i^h[\mathbf{x}(k)]b_{mi}^h, \;\; m = 1, \ldots, nU, \tag{6.14}$$

$nY = \max_{h=1,\ldots,M}\{n_y^h\}$, where n_y^h is the order of the output (n_y) of model h, $nU = \max_{h=1,\ldots,M}\{n_u^h\}$, where n_u^h is the order of the input (n_u) of model h.

6.2.3 Predictive Control Law

Using the plant model in the form of (6.10), the GPC control law is obtained so as to minimize the following cost function

$$J(k) = \sum_{p=d+1}^{N_p} \left[\hat{y}(k+p|k) - r(k+p)\right]^2 + \sum_{p=d+1}^{d+N_u} \left[\lambda(z^{-1})\Delta u(k+p-d-1|k)\right]^2, \tag{6.15}$$

where $\hat{y}(k+p|k)$ is an p-step ahead prediction of the system on instant k, $r(k+p)$ is the future reference trajectory, $\Delta = 1 - z^{-1}$, and $\lambda(z^{-1}) = \lambda_0 + \lambda_1 z^{-1} + \ldots + \lambda_{N_p+n_U-1} z^{-(N_p+n_U-1)}$ is a weighting polynomial. N_p and N_u are the output and control horizons, respectively.

Consider the following Diophantine equation (6.16):

$$
\begin{aligned}
1 &= \Delta e_p(z^{-1})\bar{a}(z^{-1}) + z^{-p} f_p(z^{-1}), && (6.16)\\
e_p(z^{-1}) &= 1 + e_{p,1} z^{-1} + \ldots + e_{p,p-1} z^{-(p-1)}, && (6.17)\\
f_p(z^{-1}) &= f_{p,0} + f_{p,1} z^{-1} + \ldots + f_{p,n_Y} z^{-n_Y}, && (6.18)
\end{aligned}
$$

where $e_p(z^{-1})$ and $f_p(z^{-1})$ can be obtained by dividing 1 by $\Delta\bar{a}(z^{-1})$ until the remainder can be factorized as $z^{-p} f_p(z^{-1})$. The quotient of the division is the polynomial $e_p(z^{-1})$. A simple and efficient way to obtain polynomials $e_p(z^{-1})$ and $f_p(z^{-1})$ is to use recursion of the Diophantine equation as demonstrated in [Camacho and Bordons, 2007]. Polynomials $e_{p+1}(z^{-1})$ and $f_{p+1}(z^{-1})$ can be obtained from polynomials of $e_p(z^{-1})$ and $f_p(z^{-1})$, respectively. Polynomials $e_{p+1}(z^{-1})$ are given by

$$
e_{p+1}(z^{-1}) = e_p(z^{-1}) + z^{-p} e_{p+1,p}, \tag{6.19}
$$

where $e_{p+1,p} = f_{p,0}$. The coefficients of polynomial $f_{p+1}(z^{-1})$ can be obtained recursively as follows:

$$
f_{p+1,i} = f_{p,i+1} - f_{p,0}\,\Delta\bar{a}_{i+1}, \quad i = 0, \ldots, n_Y - 1, \tag{6.20}
$$

where $f_{p,n_Y} = 0$. Polynomial $g_{p+1}(z^{-1})$ is expressed as:

$$
\begin{aligned}
g_{p+1}(z^{-1}) &= e_{p+1}(z^{-1})\bar{b}(z^{-1}), && (6.21)\\
&= \left[e_p(z^{-1}) + z^{-p} f_{p,0}\right] \bar{b}(z^{-1}), && (6.22)\\
&= g_p(z^{-1}) + z^{-p} f_{p,0}\bar{b}(z^{-1}), && (6.23)
\end{aligned}
$$

where the coefficients of $g_{p+1}(z^{-1})$ are given by $g_{p+1,j} = g_{p,j}$ for $j = 0, \ldots, p-1$, and

$$
g_{p+1,p+i} = g_{p,p+i} + f_{p,0}\bar{b}_i, \quad i = 0, \ldots, n_U, \tag{6.24}
$$

where $g_{p,p+n_U} = 0$. $e_p(z^{-1})$, $f_p(z^{-1})$, and $g_p(z^{-1})$ are recursively computed for $p = d+1, \ldots, N_p$. To initialize the recursion (6.16), $p = d+1$, and

$$\begin{aligned} e_{d+1}(z^{-1}) &= 1, && (6.25)\\ f_{d+1}(z^{-1}) &= z(1-\tilde{a}(z^{-1})), \\ &= \tilde{a}_1 + \tilde{a}_2 z^{-1} + \ldots + \tilde{a}_{n_Y+1} z^{-n_Y}, && (6.26) \end{aligned}$$

where

$$\tilde{a}(z^{-1}) = \Delta\bar{a}(z^{-1}) = 1 - \tilde{a}_1 z^{-1} - \ldots - \tilde{a}_{n_Y+1} z^{-(n_Y+1)}.$$

Thus,

$$g_{d+1}(z^{-1}) = e_{d+1}(z^{-1})\bar{b}(z^{-1}) = \bar{b}(z^{-1}). \quad (6.27)$$

Multiplying (6.10) by $\Delta z^p e_p(z^{-1})$ yields

$$\Delta z^p e_p(z^{-1})\bar{a}(z^{-1})y(k) = \Delta z^p e_p(z^{-1})\bar{b}(z^{-1})u(k-d-1). \quad (6.28)$$

Defining

$$\begin{aligned} g_p(z^{-1}) &= e_p(z^{-1})\bar{b}(z^{-1}), && (6.29)\\ &= g_{p,0} + g_{p,1}z^{-1} + \ldots + g_{p,p+n_U-1}z^{-(p+n_U-1)}, \end{aligned}$$

and substituting (6.16) and (6.29) into (6.28) yields

$$y(k+p|k) = f_p(z^{-1})y(k) + g_p(z^{-1})\Delta u(k+p-d-1). \quad (6.30)$$

Thus, the best prediction of $y(k+p|k)$ is

$$\hat{y}(k+p|k) = f_p(z^{-1})y(k) + g_p(z^{-1})\Delta u(k+p-d-1). \quad (6.31)$$

Equation (6.31) can be rewritten as

$$\mathbf{y}(k) = \mathbf{G}\mathbf{u}(k) + \mathbf{F}(z^{-1})y(k) + \mathbf{L}(z^{-1}), \quad (6.32)$$

where

$$\begin{aligned}
\mathbf{y}(k) &= \begin{bmatrix} \hat{y}(k+d+1) \\ \hat{y}(k+d+2) \\ \vdots \\ \hat{y}(k+N_p) \end{bmatrix}, \ \mathbf{u}(k) = \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N_u-1) \end{bmatrix}, \\
\mathbf{F} &= \begin{bmatrix} f_{d+1}(z^{-1}) \\ f_{d+2}(z^{-1}) \\ \vdots \\ f_{N_p}(z^{-1}) \end{bmatrix}, \ \mathbf{G} = \begin{bmatrix} g_{1,0} & 0 & \dots & 0 \\ g_{2,1} & g_{2,0} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_{N_p,N_p-1} & g_{N_p,N_p-2} & \dots & g_{N_p,N_p-N_u} \end{bmatrix}, \\
\mathbf{L} &= \begin{bmatrix} \left[g_{d+1}(z^{-1}) - \bar{g}_{d+1}(z^{-1})\right] z\Delta u(k-1) \\ \left[g_{d+2}(z^{-1}) - \bar{g}_{d+2}(z^{-1})\right] z^2\Delta u(k-1) \\ \vdots \\ \left[g_{N_p}(z^{-1}) - \bar{g}_{N_p}(z^{-1})\right] z^{N_p}\Delta u(k-1) \end{bmatrix}, \\
\bar{g}_p(z^{-1}) &= g_{p,0} + g_{p,1}z^{-1} + \ldots + g_{p,p-d-1}z^{d+1-p}.
\end{aligned}$$

Using (6.32) and considering $\lambda(z^{-1})$ to be constant ($\lambda > 0$), (6.15) can be rewritten as

$$J_{eq}(k) = \left[\mathbf{F}y(k) + \mathbf{G}\mathbf{u}(k) + \mathbf{L} - \mathbf{R}\right]^T \left[\mathbf{F}y(k) + \mathbf{G}\mathbf{u}(k) + \mathbf{L} - \mathbf{R}\right] + \left[\lambda\mathbf{u}(k)\right]^2, \tag{6.33}$$

where

$$\mathbf{R} = \left[r(k+d+1), \ldots, r(k+N_p)\right]^T. \tag{6.34}$$

To minimize $J_{eq}(k)$ the following equation is solved

$$\frac{\partial J_{eq}(k)}{\partial[\Delta u(k)]} = 0. \tag{6.35}$$

By minimizing $J_{eq}(k)$ using (6.35), the following optimum control increment is obtained [Camacho and Bordons, 2007]:

$$\mathbf{u}^*(k) = \frac{\mathbf{G}^T(\mathbf{R} - \mathbf{F}y(k) - \mathbf{L})}{\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I}}, \tag{6.36}$$

Algorithm 6.1 Proposed FMMGPC algorithm.

(a) Design control parameters: N_p, N_u, λ, and d.

(b) Define which T-S fuzzy models, i.e. which rule bases (input variables, respective membership functions, fuzzy rules, and the final learned model parameters) learned by HGA-FCRM, Algorithm 4.5, will be combined; And initialize $u(0)$;

(c) For/using each newly arriving online sample, do:

i. Compute $\bar{a}(z^{-1})$ and $\bar{b}(z^{-1})$ using (6.11) and (6.12), respectively;

ii. Compute the control signal $\Delta u(k)$ with (6.37).

where $\mathbf{I}$ is the identity matrix.

As the control signal sent to the process is the first row of $\mathbf{u}^*(k)$, the $\Delta u^*(k)$ is given by:

$$\Delta u^*(k) = \mathbf{K}[\mathbf{R} - \mathbf{F}y(k) - \mathbf{L}], \tag{6.37}$$

where $\mathbf{K}$ is the first row of matrix $(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I})^{-1}\mathbf{G}^T$,

$$\mathbf{K} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \end{bmatrix}_{1\times N_u} (\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I})^{-1}\mathbf{G}^T. \tag{6.38}$$

6.2.4 Fuzzy Predictive Control Framework Algorithms

As mentioned at the beginning of Section 6.2, two fuzzy predictive control frameworks are proposed:

1. The first framework (FMMGPC) uses a combination of multiple T-S fuzzy models learned by the identification methods proposed in Chapter 4, and integrates the overall combined model with a GPC controller. Algorithm 6.1 summarizes the design and operation of the FMMGPC framework.

2. The second framework (AFGPC) uses a single T-S fuzzy model learned by the identification methods proposed in Chapter 4, and integrates the learned model with a GPC controller. Additionally, AFGPC integrates a capacity of adaptation of the consequent parameters by using the adaptation law presented in Subsection 4.4.3. Algorithm 6.2 summarizes the design and operation of the AFGPC framework.

Algorithm 6.2 Proposed AFGPC algorithm.

(a) Design control parameters: N_p, N_u, λ, and d.

(b) Use the fuzzy rule base (input variables, respective membership functions, fuzzy rules, and the final learned model parameters) learned by the AHGA-FCM, Algorithm 4.6.

(c) Design the identification parameters (ρ, φ_i, τ_i, ν_i, for all $1 \leqslant i \leqslant N$) of the recursive least squares method with adaptive directional forgetting (Subsection 4.4.3), with the same values as the ones defined in Algorithm 4.6 and initialize $u(0)$.

(d) For/using each newly arriving online sample, do:

   i. Compute $\bar{a}(z^{-1})$ and $\bar{b}(z^{-1})$, with $M = 1$ using (6.11) and (6.12), respectively;

   ii. Compute the control signal $\Delta u(k)$ with (6.37);

   iii. Adapt the T-S fuzzy model parameters (a_{ji} and b_{ji} of (6.3)) by performing one iteration of recursion (4.42).

6.3 Experimental Results

This section presents simulation and real-world results to demonstrate the feasibility, performance and effectiveness of the proposed fuzzy predictive control methodologies. The control of the product concentration of a simulated CSTR plant, and the control of the velocity of a real-world experimental setup composed of two coupled DC motors are studied. The proposed methodologies were implemented in the Matlab Software with the main functions being implemented in the C programming language to reduce computational time.

The prediction models that will be used on FMMGPC and AFGPC frameworks are the ones that were learned in the experiments described in Section 4.6.

6.3.1 Continuous-Stirred Tank Reactor

In this subsection, the control of the measured concentration of $y(t) = C_A(t)$ of the CSTR plant defined in Subsection 4.6.2, is studied. The control is performed by manipulating the coolant flow rate $u(t) = q_c(t)$. A stochastic disturbance $\vartheta(t)$, namely a Gaussian white noise, is also considered in the CSTR model (4.45).

The prediction models that are used on the FMMGPC and AFGPC frameworks,

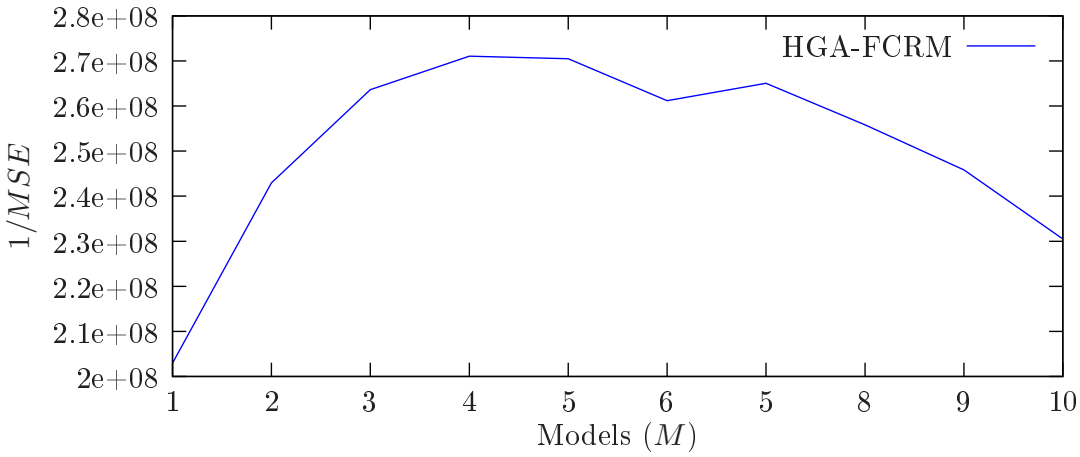


Figure 6.3: Performances of combinations among the M best models, $M = 1, \ldots, 10$, obtained by the HGA-FCRM method in the CSTR process, in Subsection 4.6.2 in Chapter 4.

in this subsection, are the same as the models learned in Subsection 4.6.2. Specifically, for the FMMGPC framework, the best combination of the best M models learned by the HGA-FCRM algorithm in Subsection 4.6.2 is used, assuming the possible values of $M = 1, \ldots, 10$. As described below, $M = 4$ is the optimum. For the AFGPC framework, the best model learned by the AHGA-FCM algorithm in Subsection 4.6.2 is used.

Figure 6.3 illustrates the prediction performance results of the combinations of the best M models trained by the HGA-FCRM algorithm, for $M = 1, \ldots, 10$. The first point in the figure corresponds to the best learned model, the second point corresponds to the combination of the two best learned models, until the tenth point that corresponds to the combination of the ten best learned models. As can be seen, the best performance is obtained by combining the best four learned models ($M = 4$).

For a better study, two cases were tested. In both cases, the following controller parameters were chosen by the user for the classic GPC controller in both the FMMGPC and the AFGPC frameworks: $N_p = 150$, $N_u = 1$, $\lambda = 0.05$, and $d = 5$. Additionally, the following identification parameters $\rho = 0.999$, $\varphi_i = 1$, $\tau_i = \nu_i = 10^{-9}$, of the recursive least squares method with adaptive directional forgetting (Subsection 4.4.3), for all $1 \leqslant i \leqslant N$, were used for the AFGPC framework. A Zero-mean

Gaussian white noise with a variance of 2×10^{-6} [(mol/l/min)^2] was also applied as a stochastic disturbance $\vartheta(t)$. The linear model parameters used in the GPC controller (independent GPC controller; not the FMMGPC nor the AFGPC) were obtained with the Reaction Curve Method from [Camacho and Bordons, 2007] which gave the polynomials $\bar{a}(z^{-1}) = 1 - 0.8507z^{-1}$ (6.11) and $\bar{b}(z^{-1}) = 0.5263 \times 10^{-4}z^{-1}$ (6.12) to be used on the model (6.10).

Case 1*:* The reference input was

$$r(t) = \begin{cases} 0.11, & 0 < t \leqslant 5\,[\text{hours}], \\ 0.07, & 5\,[\text{hours}] < t \leqslant 10\,[\text{hours}], \\ 0.1, & 10\,[\text{hours}] < t \leqslant 15\,[\text{hours}], \\ 0.08, & 15\,[\text{hours}] < t \leqslant 20\,[\text{hours}], \\ 0.12, & 20\,[\text{hours}] < t \leqslant 15\,[\text{hours}], \end{cases} \tag{6.39}$$

and a disturbance was defined as a change of the process flow rate q, where $q = 110$ for $13\,[\text{hours}] \leqslant t \leqslant 17\,[\text{hours}]$.

From the results presented in Figure 6.4, it can be seen that the proposed FMMGPC and AFGPC frameworks are able to adequately (attain and) control the system output at the desired reference $r(t)$. When the load disturbance is applied at $13\,[\text{hours}] \leqslant t \leqslant 17\,[\text{hours}]$, there is an undershoot at $t = 13\,[\text{hours}]$ and an overshoot at $t = 17\,[\text{hours}]$ in the system responses. As can be seen, both controllers eliminate this disturbance.

Case 2*:* In this case, the reference input was constant, $r(t) = 0.1$ for $0 < t \leqslant 25\,[\text{hours}]$, and a disturbance was defined as a change of the feed concentration C_{A0}, where $C_{A0} = 0.97$ for $6\,[\text{hours}] \leqslant t \leqslant 17\,[\text{hours}]$, and a change of the inlet coolant temperature T_{c0}, where $T_{c0} = 345$ for $12\,[\text{hours}] \leqslant t \leqslant 20\,[\text{hours}]$.

From the results presented in Figure 6.5, it can be seen that the proposed controllers are able to adequately (attain and) control the system output at the desired reference $r(t)$, also in in Case 2. As can be seen both the FMMGPC and AFGPC frameworks eliminate the disturbances.

From the results, it is concluded that the proposed FMMGPC and AFGPC frameworks can control the process using only a data set of the process to initialize the T-S fuzzy model(s). From the results evidenced in *Case* 1, it is also concluded that both the FMMGPC and AFGPC controllers, using the same values for the pa-

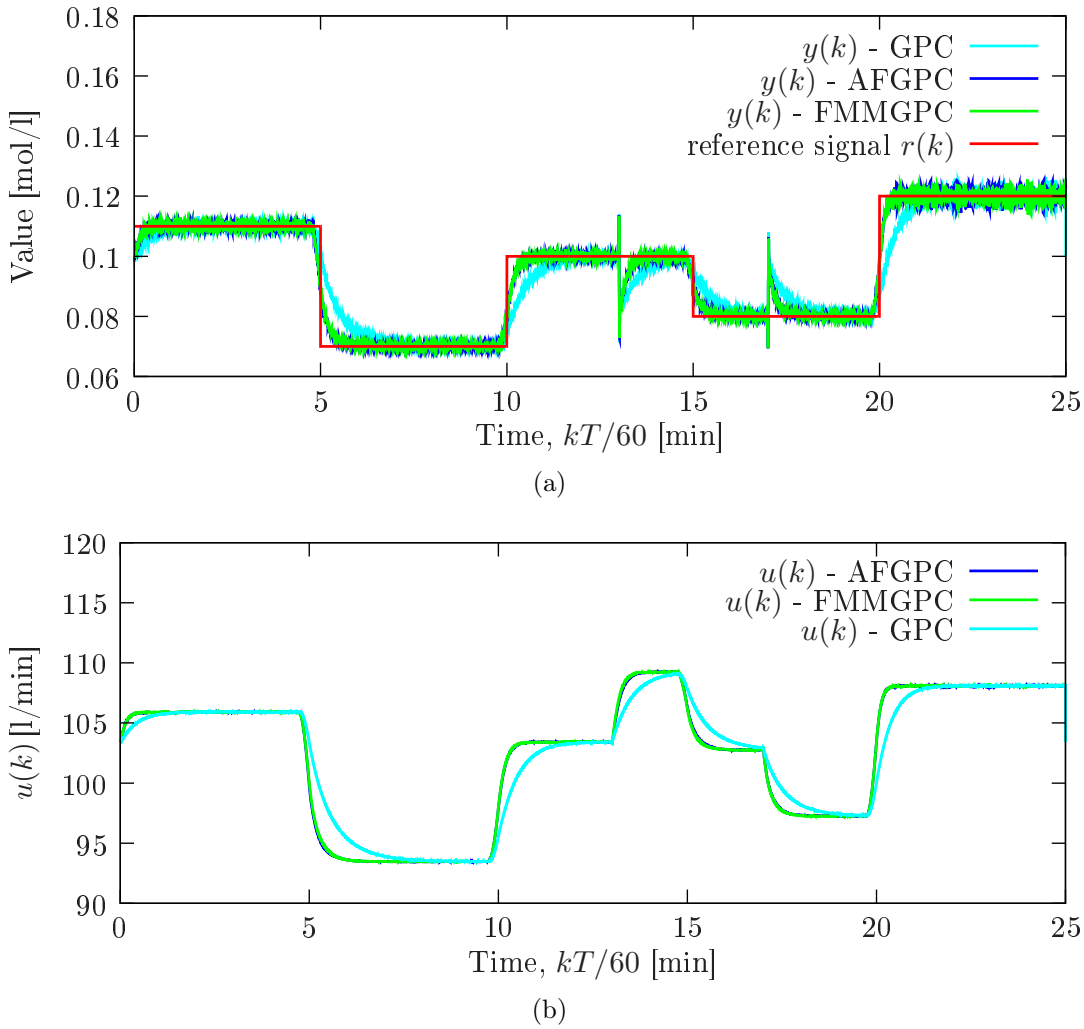


Figure 6.4: (a) Results of the proposed FMMGPC and AFGPC frameworks in the presence of disturbance in the process flow rate q (for $13\,[\text{hours}] \leqslant t \leqslant 17\,[\text{hours}]$) in the CSTR process for the study Case 1; and (b) the respective applied command signal.

rameters which are common with the GPC controller (N_p, N_u, λ, and d), outperform the GPC controller.

In terms of computational effort, the proposed FMMGPC and AFGPC frameworks have two stages of implementation. The first stage is off-line, where an identification of the process model is done. This first stage has a high computational effort. The second stage is performed on-line (in real-time), and is where the control

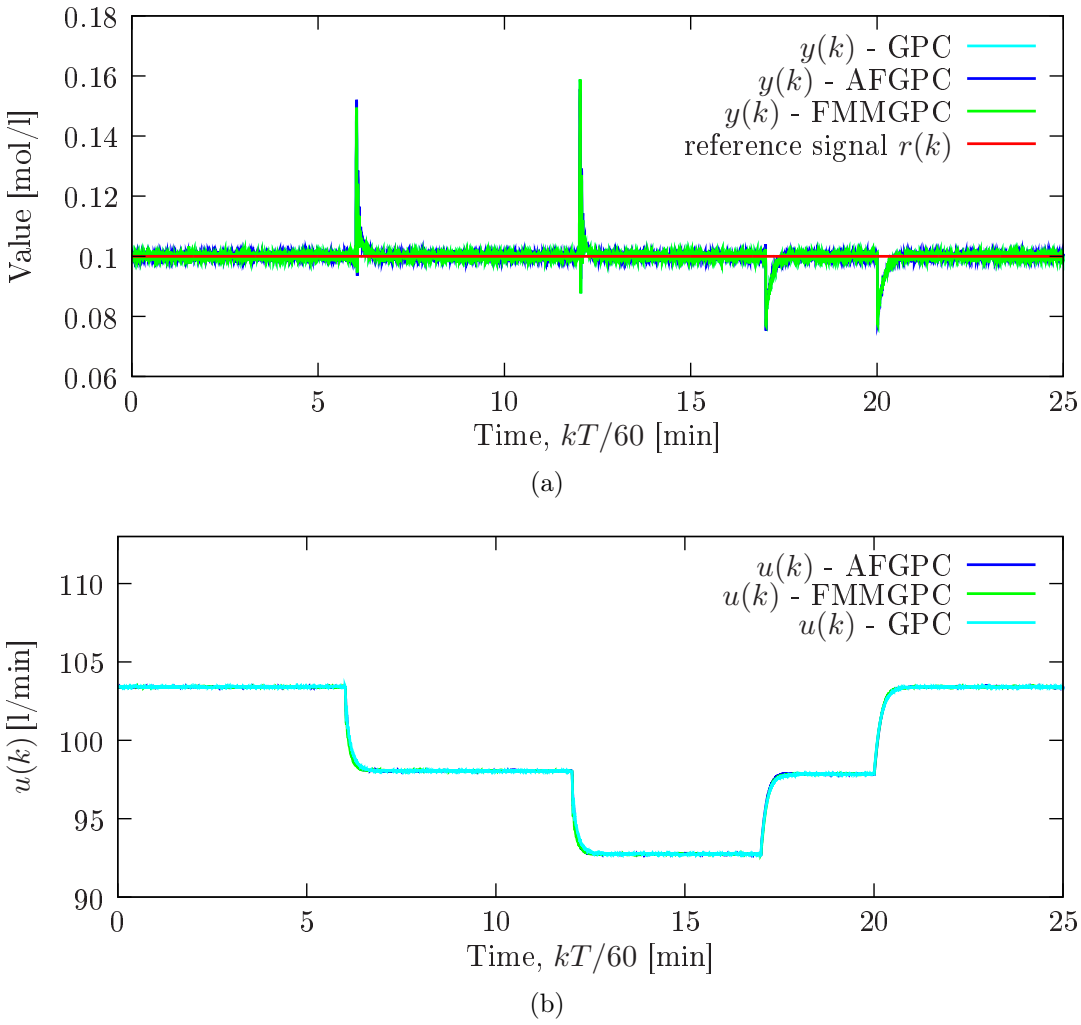


Figure 6.5: (a) Results of the proposed FMMGPC and AFGPC frameworks in the presence of disturbances on the feed concentration C_{A0} (for 6 [hours] $\leqslant t \leqslant$ 17 [hours]) and on the inlet coolant temperature T_{c0} (for 12 [hours] $\leqslant t \leqslant$ 20 [hours]) in the CSTR process for the study Case 2; and (b) the respective applied command signal.

is performed. The second stage is composed of the recursive computation sequence used to obtain the controller parameters (equations (6.19)-(6.27)) with $N_u = 1$, and then by the computation of the control commands (equations (6.29), (6.33), (6.34), (6.38)). In this second stage, the computational effort is very small (e.g. using Matlab with the some functions being implemented in the C programming language, the control computational time, using an Intel Core i5-760 Processor (8Mb Cache,

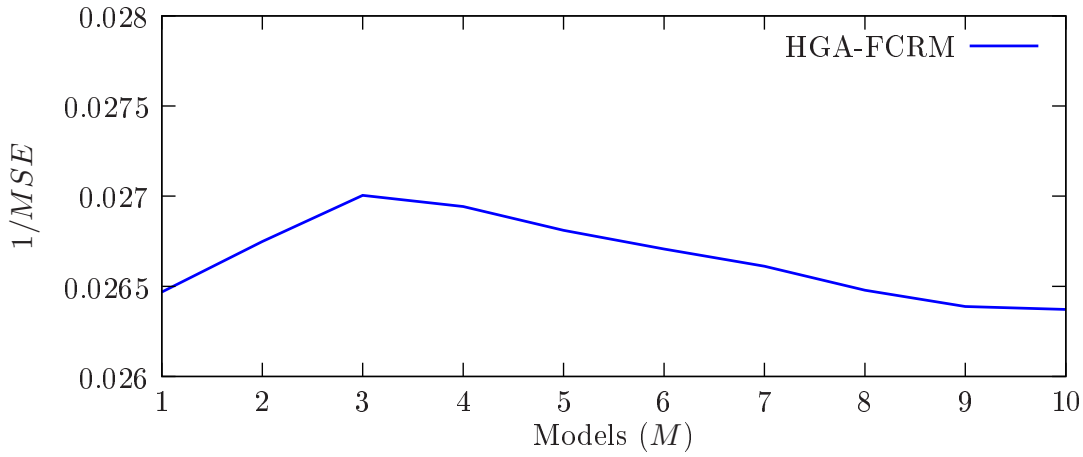


Figure 6.6: Performances of combinations among the M best models, $M = 1, \ldots, 10$, obtained by the HGA-FCRM method in the real DC motors process, in Subsection 4.6.3 in Chapter 4.

2.80 GHz) and 8Gb of RAM, for FMMGPC, using four models, is 0.0012 [s], and for AFGPC is 0.009 [s]).

6.3.2 Real-World Control of Two Coupled DC Motors

In this subsection, the main goal is to control the velocity, $y(k)$, of the real-world setup composed of two coupled DC motors (one working as a motor and the other working as a generator), defined in Subsection 4.6.3, by manipulating the voltage $u(k)$ applied to the motor. Additionally, the DC motor is exposed to load changes.

The prediction models that are used on the FMMGPC and AFGPC frameworks, in this subsection, are the same as the models learned in Subsection 4.6.3. Specifically, for the FMMGPC framework, the best combination of the best M models learned by the HGA-FCRM algorithm in Subsection 4.6.2 is used, assuming the possible values of $M = 1, \ldots, 10$. As described below, $M = 3$ is the optimum. For the AFGPC framework, the best model learned by the AHGA-FCM algorithm in Subsection 4.6.3 is used.

Figure 6.6 illustrates the prediction performance results of the combinations of the best M models obtained by the HGA-FCRM algorithm, for $M = 1, \ldots, 10$. The first point in the figure corresponds to the best learned model, the second point corresponds to the combination of the two best learned models, until the tenth point

that corresponds to the combination of the ten best learned models. As can be seen, the best performance is obtained by combining the best three learned models ($M = 3$).

The following controller parameters were chosen by the user for the classic GPC controller in both the FMMGPC and the AFGPC frameworks: $N_p = 10$, $N_u = 1$, $\lambda = 80$, and $d = 0$. Additionally, the following identification parameters $\rho = 0.93$, $\varphi_i = 1$, $\tau_i = 10^{-3}$, $\nu_i = 10^{-6}$, of the recursive least squares method with adaptive directional forgetting (Subsection 4.4.3), for all $1 \leqslant i \leqslant N$, were used for the AFGPC framework. The linear model parameters used in the (independent) GPC controller were obtained with the Reaction Curve Method from [Camacho and Bordons, 2007] which gave the polynomials $\bar{a}(z^{-1}) = 1 - 0.9460z^{-1}$ (6.11) and $\bar{b}(z^{-1}) = 0.0951z^{-1}$ (6.12) to be used on the model (6.10).

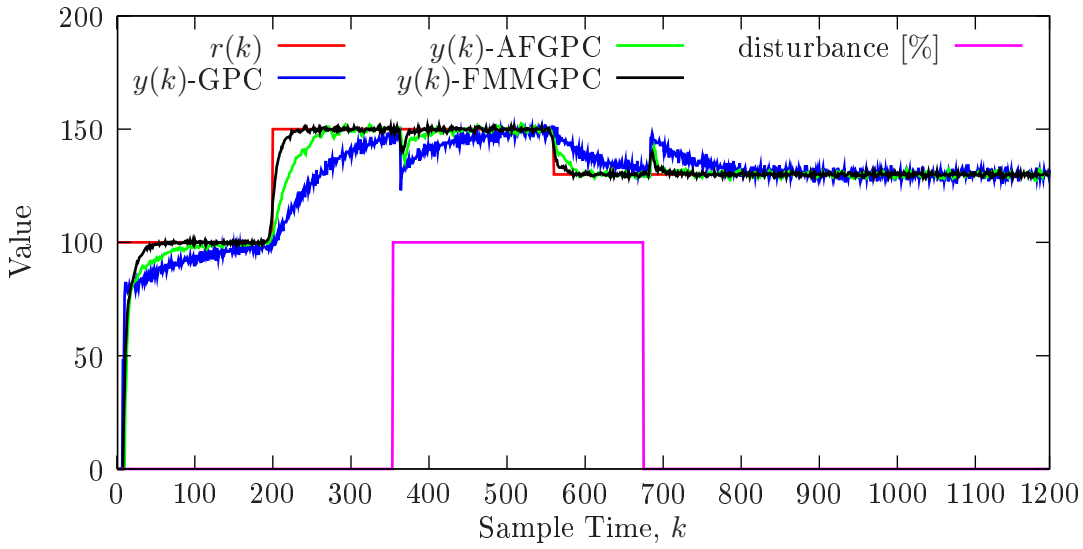
The reference input was

$$r(k) = \begin{cases} 100, & 0 < k \leqslant 120, \\ 150, & 120 < k \leqslant 320, \\ 130, & 320 < k \leqslant 600, \end{cases} \tag{6.40}$$

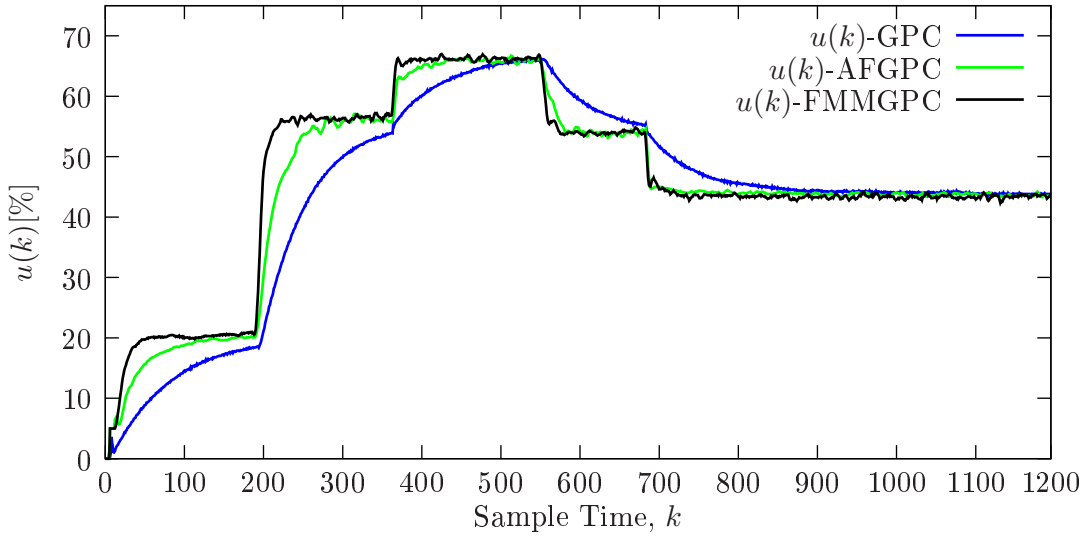
and the load disturbance was applied at $360 \leqslant k \leqslant 680$ (lamps switched-on).

From the results presented in Figure 6.7, it can be seen that the proposed FMMGPC and AFGPC frameworks are able to adequately (attain and) control the system output at the desired reference $r(k)$. When the load disturbance is applied at $360 \leqslant k \leqslant 680$, there is an undershoot at $k = 360$ and an overshoot at $k = 680$ in the system responses. As can be seen the FMMGPC and AFGPC frameworks eliminate this disturbance. By the results, it is concluded that the proposed FMMGPC and AFGPC frameworks can control the process using only a data set of the process to initialize the T-S fuzzy model(s), and that both the FMMGPC and AFGPC controllers, using the same values of the parameters which are common with with the GPC controller (N_p, N_u, λ, and d), outperform the GPC controller.

In terms of computational effort, with the same conditions as in Section 6.3.1 (using the recursive computation sequence and the same computer), the computation time for FMMGPC framework with three models is 0.0010 [s], and the computation time for AFGPC framework is 0.009 [s].



(a)



(b)

Figure 6.7: (a) Results of the proposed FMMGPC and AFGPC frameworks and the GPC controller, in the presence of load disturbances (lamps switched on for $360 \leqslant k \leqslant 680$) in the real-world DC motors process; and (b) the respective applied command signal. The units for $y(k)$ and $r(k)$ are [pp/(0.25 sec)].

6.4 Conclusion

In this chapter the problem of the assumption of knowledge about an accurate model of the process in MPC control architectures was investigated. This aspect can present problems because many complex plants are difficult to be mathematically modelled based on physical laws, or have large uncertainties and strong nonlinearities. The methodologies proposed in Chapter 4 to automatically identify a T-S fuzzy model from input/output data to approximate unknown nonlinear processes were used to learn the prediction model of the generalized predictive control (GPC) algorithm.

Taking into account the above mentioned problems, in this chapter two fuzzy predictive control frameworks were proposed. The T-S fuzzy models learned by the identification methods proposed in Chapter 4 were integrated into the generalized predictive control (GPC) approach. The first proposed framework, the Fuzzy Multiple Models Generalized Predictive Control (FMMGPC) approach, is based on a GPC controller used in conjunction with a combination of multiple T-S fuzzy models learned by the identification methods proposed in Chapter 4. The second framework is based on the integration of a T-S fuzzy model learned by the identification methods proposed in Chapter 4, into an adaptive fuzzy GPC (AFGPC) controller, an approach named as AFGPC.

To validate and demonstrate the performance and effectiveness of the proposed frameworks, they were tested on the control of a simulated continuous stirred tank reactor (CSTR), and on a real-world experimental setup composed of two coupled DC motors. The results have also shown that the proposed frameworks can control the processes using only a data set of the processes to design the respective process model.

Thus, the aim of proposing frameworks to control industrial processes without knowledge about the plant to be controlled (problem of the assumption of knowledge of an accurate model in MPCs), and using the methodologies proposed in Chapter 4 to design the process model, was reached.

Chapter 7

Conclusions

Industrial processes have faced major changes in the market during the past decades, due to the increasing world competition, and the environmental legislation, which resulted in hard constraints that increase the process complexity and the costs of production equipment. Many industrial systems exhibit nonlinear behaviors and frequently have many complex characteristics, such as unknown and time-varying dynamics, constraints, and disturbances.

Fuzzy logic have been used for a wide variety of industrial systems and consumer products, attracting, increasingly, the attention of many researchers. A major application of fuzzy theory has been in control of nonlinear systems which are typically difficult to model and control, when a mathematical model of the process is poorly understood or is unknown, and expert human knowledge about the process or about the control (e.g. experienced operators) is available. Thus, this thesis has proposed methodologies for addressing identification and control problems on nonlinear industrial processes using fuzzy logic theory, and learning and adaptive approaches.

Several topics about fuzzy logic for industrial applications were discussed in Chapter 2. From the point of view of the author of this thesis, the most emergent topics are: the problems about the assumption of knowledge of an accurate model in MPCs, because many complex plants are difficult to be mathematically modelled based on physical laws, or have large uncertainties and strong nonlinearities; and the difficulty of designing fuzzy systems to solve certain complex nonlinear problems, when the only available knowledge concerning the process is the empirical information transmitted by a human operator. Motivated by these problems, three

main research directions were addressed: automatic identification of T-S fuzzy models, automatic design of FLC, and design frameworks for predictive control without prior knowledge about a model of the plant to be controlled.

In Chapter 4 methodologies for identification of industrial processes were proposed. The learning of a T-S fuzzy model is performed from input/output data to approximate unknown nonlinear processes by a HGA, which optimize a large set of T-S fuzzy parameters encoded at five different hierarchical levels. The proposed methods are automatic tools for T-S fuzzy model design, with the following main characteristics: (1) automatically performing the optimization of the variable and delay selection jointly with the learning and optimization of the system model without the need for any prior human knowledge, (2) the T-S fuzzy model structure is constructed just according to the data characteristics, and (3) it is optimized by means of GAs. Three methodologies were proposed: a HGA without initialization method, named as HGA, a HGA with the FCRM initialization method, named as HGA-FCRM, and a HGA with the FCM initialization method and using an adaptive methodology to update the consequent parameters of the T-S fuzzy system, named as AHGA-FCM. The identification performance of the proposed methodologies were quantitatively compared with three non-adaptive approaches: MLP, ELM, and the HGA proposed in [Delgado *et al.*, 2009]; and two adaptive approaches: RPLS and ILLSA. Taking into account the results, the proposed methodologies are able to design all the parts of the T-S fuzzy prediction model, identifying nonlinear systems satisfactorily with appropriate input variables and delay selection, and with reasonable number of rules. Numerical results have shown that AHGA-FCM methodology has a superior performance when compared to the other state of art adaptive methods, and that the HGA-FCRM methodology has a superior performance when compared the other three state of art non-adaptive methods.

Off-line training approaches, that make use of a set of data previously collected from the plant, may not provide adequate accuracy in parts or the whole operating areas of the plant. The data set may be not sufficiently representative of the plant, or the plant may undergo dynamic changes. All algorithms proposed in Chapter 4 used off-line approaches for training, at least fo training part of the model, and in particular to train the antecedent part of the model. Therefore, self-adaptive strategies, more specifically related to the on-line optimization approaches for the antecedent membership functions, can be considered as a future research topic.

In Chapter 5, methodologies have been proposed to automatically extract all fuzzy parameters of a fuzzy logic controller from data collected from a given process while it is being controlled (e.g. a process under manual control), in order to control nonlinear industrial processes. The proposed methodologies do not require any prior knowledge concerning the fuzzy rule structure, location or shape of membership functions, implication and aggregation operators, defuzzification methods, or selection of adequate input variables and corresponding time delays. Two methodologies were proposed: a HGA for control which uses an initialization method based on [Andersen *et al.*, 1997], named as HGA-Control; and a HGA, named as AHGA-Control, which uses an initialization method based on [Andersen *et al.*, 1997] and on the FCM method, and where to improve the performance of the learned FLC, a direct adaptive fuzzy control methodology is applied for on-line adaptation of the consequent parameters of the fuzzy control rules. The presented results show that the proposed methodologies are able to extract all the parameters of the FLC, and to successfully control nonlinear processes using only a data set obtained from a process under control (e.g. under manual control).

Similarly to the future research topic suggested above, the development of on-line approaches for the optimization of the antecedent membership functions can be considered a promising future research focus in the continuation of the work in HGA-Control and AHGA-Control. Another possible topic of future research is to combine the proposed methods with iterative rule learning techniques.

In Chapter 6 the problem of the assumption of knowledge about an accurate model of the process in MPC control architectures was addressed. The methodologies proposed in Chapter 4 to automatically identify a T-S fuzzy model from input/output data to approximate unknown nonlinear processes were used to learn the prediction model of the generalized predictive control (GPC) algorithm. Two fuzzy predictive control frameworks were proposed: the FMMGPC framework that is based on a GPC controller and uses a combination of multiple T-S fuzzy models learned by the identification methods proposed in Chapter 4; and the AFGPC framework which integrate a T-S fuzzy model learned by the identification methods proposed in Chapter 4 and uses an adaptive method to adapt the consequent parameters of the model. Taking into account the results, the aim of proposing frameworks to control industrial processes without knowledge about the plant to be controlled, using the methodologies proposed in Chapter 4, was reached. Robustness

and stability are relevant possible topics for future research work in the proposed predictive control frameworks.

In this thesis, some of problems such as the ones mentioned above, have been studied and new methods have been proposed and developed in the course of this work for overcoming such problems. The proposed methods for identification have been compared with the state of the art methods, and two of the proposed methods, the AHGA-FCM and HGA-FCRM methodologies, had a superior performance when compared to the other state of the art adaptive and non-adaptive methods, respectively, for all the case studies (Chapter 4); the methods proposed to automatically extract all fuzzy parameters of a FLC in order to control nonlinear processes without any prior knowledge about the control process have been successfully tested with good results (Chapter 5); and the aim of proposing predictive control frameworks to control processes using only a data set of the processes to design the respective process model have been successfully reached (Chapter 6).

Bibliography

[Al-Hadithi *et al.*, 2012] Basil M. Al-Hadithi, Agustín Jiménez, and Fernando Matía. A New Approach to Fuzzy Estimation of Takagi-Sugeno Model and its Applications to Optimal Control for Nonlinear Systems. *Applied Soft Computing*, vol. 12, no. 1, pp. 280–290, 2012. (Cited in page 11).

[Alam and Tokhi, 2008] M.S. Alam and M.O. Tokhi. Hybrid Fuzzy Logic Control With Genetic Optimisation for a Single-Link Flexible Manipulator. *Engineering Applications of Artificial Intelligence*, vol. 21, no. 6, pp. 858–873, September 2008. (Cited in page 86).

[Ali and Ramaswamy, 2009] Sk. Faruque Ali and Ananth Ramaswamy. Optimal Fuzzy Logic Control for MDOF Structural Systems Using Evolutionary Algorithms. *Engineering Applications of Artificial Intelligence*, vol. 22, no. 3, pp. 407–419, April 2009. (Cited in page 86).

[Andersen *et al.*, 1997] H.C. Andersen, A. Lotfi., and A.C. Tsoi. A New Approach to Adaptive Fuzzy Control: The Controller Output Error Method. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 27, no. 4, pp. 686–691, August 1997. (Cited in pages 88, 94, 120, and 145).

[Anh and Ahn, 2009] Ho Pham Huy Anh and Kyoung Kwan Ahn. Identification of Pneumatic Artificial Muscle Manipulators by a MGA-Based Nonlinear NARX Fuzzy Model. *Mechatronics*, vol. 19, no. 1, pp. 106–133, February 2009. (Cited in page 41).

[Babuska and Verbruggen, 1995] R. Babuska and H.B. Verbruggen. Identification of Composite Linear Models Via Fuzzy Clustering. In: *Proceedings of the European Control Conference (ECC 95)*, pp. 1207–1212. 1995. (Cited in page 11).

[Babuska and Verbruggen, 1996] Robert Babuska and Henk B. Verbruggen. An Overview of Fuzzy Modeling for Control. *Control Engineering Practice*, vol. 4, no. 11, pp. 1593–1606, 1996. (Cited in page 11).

[Belchior *et al.*, 2012] Carlos Alberto Coelho Belchior, Rui Alexandre Matos Araújo, and Jorge Afonso Cardoso Landeck. Dissolved Oxygen Control of the Activated Sludge Wastewater Treatment Process Using Stable Adaptive Fuzzy Control. *Computers & Chemical Engineering*, vol. 37, pp. 152–162, 2012. (Cited in pages xx, 104, and 105).

[Bellman and Zadeh, 1970] R.E. Bellman and L.A. Zadeh. Decision-Making in a Fuzzy Environment. *Management Science*, vol. 17, 1970. (Cited in page 10).

[Ben-Israel and Greville, 2003] Adi Ben-Israel and Thomas N.E. Greville. *Generalized Inverses: Theory and Applications*. Springer-Verlag, New York, USA, second ed., 2003. (Cited in page 45).

[Bishop, 2006] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 1 ed., 2006. (Cited in page 128).

[Bobál *et al.*, 2005] Vladimír Bobál, Josef Böhm, Jaromír Fessl, and Jiří Macháček. *Self-tuning PID Controllers*. Advanced Textbooks in Control and Signal Processing. Springer London, 2005. (Cited in pages 59, and 63).

[Camacho and Bordons, 2007] E. F. Camacho and C. Bordons. *Model Predictive Control*. Advanced Textbooks in Control and Signal Processing. Springer-Verlag, London, 2nd ed., July 2007. (Cited in pages 20, 40, 123, 130, 132, 136, and 140).

[Cao *et al.*, 2013] Hui Cao, Lixin Jia, Gangquan Si, and Yanbin Zhang. A Clustering-Analysis-Based Membership Functions Formation Method for Fuzzy Controller of Ball Mill Pulverizing System. *Journal of Process Control*, vol. 23, no. 1, pp. 34–43, January 2013. (Cited in page 11).

[Cazarez-Castro *et al.*, 2010] Nohe R. Cazarez-Castro, Luis T. Aguilar, and Oscar Castillo. Fuzzy Logic Control With Genetic Membership Function Parameters Optimization for the Output Regulation of a Servomechanism With Nonlinear Backlash. *Expert Systems with Applications*, vol. 37, no. 6, pp. 4368–4378, June 2010. (Cited in page 16).

[Celikyilmaz and Trksen, 2009] Asli Celikyilmaz and I. Burhan Trksen. *Modeling Uncertainty with Fuzzy Logic: With Recent Theory and Applications*. Springer Publishing Company, Incorporated, 1st ed., 2009. (Cited in pages 53, 88, 94, and 120).

[Chen and Lin, 2007] Y.M. Chen and Chun-Ta Lin. Dynamic Parameter Optimization of Evolutionary Computation for On-Line Prediction of Time Series With Changing Dynamics. *Applied Soft Computing*, vol. 7, no. 4, pp. 1170–1176, August 2007. (Cited in pages 41, and 42).

[Clarke, 1988] D. W. Clarke. Application of Generalized Predictive Control to Industrial Process. *IEEE Control Systems Magazine*, vol. 8, no. 2, pp. 49–55, 1988. (Cited in pages 20, and 124).

[Coban and Can, 2010] Ramazan Coban and Burhanettin Can. A Trajectory Tracking Genetic Fuzzy Logic Controller for Nuclear Research Reactors. *Energy Conversion and Management*, vol. 51, no. 3, pp. 587–593, March 2010. (Cited in page 87).

[Cordón *et al.*, 2001] Oscar Cordón, Francisco Herrera, Frank Hoffmann, and Luis Magdalena. *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases*, vol. 19 of *Advances in Fuzzy Systems–Applications and Theory*. World Scientific Publishing Co. Pte. Ltd., 1 ed., 2001. (Cited in pages 6, and 37).

[Cutler and Ramaker, 1980] C. R. Cutler and B. L. Ramaker. Dynamic Matrix Control - a Computer Control Algorithm. In: *Proceedings of the Joint Automatic Control Conference*. San Francisco, California, 1980. (Cited in page 20).

[Darwin, 1859] Charles Darwin. *On The Origin of Species by Means of Natural Selection*. John Murray, John Murray, Albermarle Street, London, 1859. (Cited in page 16).

[Dayal and MacGregor, 1997] Bhupinder S. Dayal and John F. MacGregor. Recursive Exponentially Weighted PLS and its Applications to Adaptive Control and Prediction. *Journal of Process Control*, vol. 7, no. 3, pp. 169–179, 1997. (Cited in page 65).

[Delgado *et al.*, 2009] Myriam Regattieri Delgado, Elaine Yassue Nagai, and Lúcia Valéria Ramos de Arruda. A Neuro-Coevolutionary Genetic Fuzzy System to Design Soft Sensors. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 13, no. 5, pp. 481–495, 2009. (Cited in pages xxiii, 41, 42, 43, 44, 69, 75, 78, 80, 82, 83, 84, and 144).

[Delgado *et al.*, 2001] Myriam Regattieri Delgado, Fernando Von Zuben, and Fernando Gomide. Hierarchical Genetic Fuzzy Systems. *Information Sciences*, vol. 136, no. 1-4, pp. 29 – 52, 2001. (Cited in pages 42, and 46).

[Dounis *et al.*, 2013] Anastasios I. Dounis, Panagiotis Kofinas, Constantine Alafodimos, and Dimitrios Tseles. Adaptive Fuzzy Gain Scheduling {PID} Controller for Maximum power Point Tracking of Photovoltaic System. *Renewable Energy*, vol. 60, no. 0, pp. 202–214, December 2013. (Cited in page 14).

[Dovžan and Škrjanc, 2011] Dejan Dovžan and Igor Škrjanc. Recursive Fuzzy c-Means Clustering for Recursive Fuzzy Identification of Time-Varying Processes. *ISA Transactions*, vol. 50, no. 2, pp. 159–169, 2011. (Cited in pages 53, 54, 55, 88, 94, and 120).

[Du and Zhang, 2008] Haiping Du and Nong Zhang. Application of Evolving Takagi-Sugeno Fuzzy Model to Nonlinear System Identification. *Applied Soft Computing*, vol. 8, no. 1, pp. 676–686, January 2008. (Cited in page 17).

[Espinosa *et al.*, 2004] Jairo Espinosa, Joos Vandewalle, and Vincent Wertz. *Fuzzy Logic, Identification and Predictive Control (Advances in Industrial Control)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2004. ISBN 1852338288. (Cited in page 93).

[Fortuna *et al.*, 2006] Luigi Fortuna, Salvatore Graziani, and Alessandro Rizzo. *Soft Sensors for Monitoring and Control of Industrial Processes (Advances in Industrial Control)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 1846284791. (Cited in pages 40, and 65).

[Gómez-Skarmeta *et al.*, 1999] A. F. Gómez-Skarmeta, M. Delgado, and M. A. Vila. About the Use of Fuzzy Clustering Techniques for Fuzzy Model Identification. *Fuzzy Sets and Systems*, vol. 106, no. 2, pp. 179–188, September 1999. (Cited in page 11).

[Han *et al.*, 2012] Hong-Gui Han, Jun-Fei Qiao, and Qi-Li Chen. Model Predictive Control of Dissolved Oxygen Concentration Based on a Self-Organizing RBF Neural Network. *Control Engineering Practice*, vol. 20, no. 4, pp. 465–476, April 2012. (Cited in pages 124, and 125).

[Herrera *et al.*, 1995] F. Herrera, M. Lozano, and J. L. Verdegay. Tuning Fuzzy Logic Controllers by Genetic Algorithms. *International Journal of Approximate Reasoning*, vol. 12, no. 3-4, pp. 299–315, April-May 1995. (Cited in page 86).

[Herrera, 2008] Francisco Herrera. Genetic Fuzzy Systems: Taxonomy, Current Research Trends and Prospects. *Evolutionary Intelligence*, vol. 1, no. 1, pp. 27–46, 2008. (Cited in pages 6, and 37).

[Hojati and Gazor, 2002] Mehrdad Hojati and Saeed Gazor. Hybrid Adaptive Fuzzy Identification and Control of Nonlinear Systems. *IEEE Trans. Fuzzy Systems*, vol. 10, no. 2, pp. 198–210, April 2002. (Cited in page 13).

[Holland, 1975] John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, USA, 1975. (Cited in page 16).

[Holmblad and Østergaard, 1995] L. P. Holmblad and J.-J. Østergaard. The FLS Application of Fuzzy Logic. *Fuzzy Sets and Systems*, vol. 70, no. 2-3, pp. 135–146, March 1995. ISSN 0165-0114. (Cited in page 10).

[Homayouni *et al.*, 2009] Seyed Mahdi Homayouni, Tang Sai Hong, and Napsiah Ismail. Development of Genetic Fuzzy Logic Controllers for Complex Production Systems. *Computers & Industrial Engineering*, vol. 57, no. 4, pp. 1247–1257, November 2009. (Cited in page 86).

[Huang *et al.*, 2006] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme Learning Machine: Theory and Applications. *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006. ISSN 0925-2312. doi:10.1016/j.neucom.2005.12.126. (Cited in page 65).

[Jang, 1993] Jyh-Shing Roger Jang. ANFIS: Adaptive-Network-Based Fuzzy Inference System. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 23, no. 3, pp. 665–685, May 1993. (Cited in page 15).

[Jeppsson and Pons, 2004] Ulf Jeppsson and Marie-Noëlle Pons. The COST Benchmark Simulation Model-Current State and Future Perspective. *Control Engineering Practice*, vol. 12, no. 3, pp. 299–304, March 2004. (Cited in pages 104, and 105).

[Juang and Lin, 1999] Chia-Feng Juang and Chin-Teng Lin. A Recurrent Self-Organizing Neural Fuzzy Inference Network. *IEEE Transactions on Neural Networks*, vol. 10, no. 4, pp. 828–845, July 1999. (Cited in page 15).

[Kadlec and Gabrys, 2011] Petr Kadlec and Bogdan Gabrys. Local Learning-Based Adaptive Soft Sensor for Catalyst Activation Prediction. *AIChE Journal*, vol. 57, no. 5, pp. 1288–1301, May 2011. (Cited in page 65).

[Kalyoncu and Haydim, 2009] Mete Kalyoncu and Mustafa Haydim. Mathematical Modelling and Fuzzy Logic Based Position Control of an Electrohydraulic Servosystem With Internal Leakage. *Mechatronics*, vol. 19, no. 6, pp. 847–858, September 2009. (Cited in page 11).

[Kasabov, 1996] Nicola K. Kasabov. *Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering.* MIT Press, Cambridge, 1996. (Cited in page 38).

[Kayadelen, 2011] C. Kayadelen. Soil Liquefaction Modeling by Genetic Expression Programming and Neuro-Fuzzy. *Expert Systems with Applications*, vol. 38, no. 4, pp. 4080–4087, April 2011. (Cited in pages 124, and 125).

[Kim and Jeon, 2011] Dohyun Kim and Doyoung Jeon. Fuzzy-Logic Control of Cutting Forces in CNC Milling Processes Using Motor Currents as Indirect Force Sensors. *Precision Engineering*, vol. 35, no. 1, pp. 143–152, January 2011. (Cited in page 11).

[Kim *et al.*, 2006] Min-Soeng Kim, Chang-Hyun Kim, and Ju-Jang Lee. Evolving Compact and Interpretable Takagi-Sugeno Fuzzy Models With a New Encoding Scheme. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 36, no. 5, pp. 1006–1023, October 2006. (Cited in page 41).

[Kosko, 1994] B. Kosko. Fuzzy Systems as Universal Approximators. *IEEE Transactions on Computers*, vol. 43, no. 11, pp. 1329–1333, November 1994. (Cited in pages 11, 21, 40, and 124).

[Kulhavý, 1987] Rudolf Kulhavý. Restricted Exponential Forgetting in Real-Time Identification. *Automatica*, vol. 23, no. 5, pp. 589–600, September 1987. (Cited in pages 59, and 63).

[Kumar *et al.*, 2012] P. Ganesh Kumar, T. Aruldoss Albert Victoire, P. Renukadevi, and D. Devaraj. Design of Fuzzy Expert System for Microarray Data Classification Using a Novel Genetic Swarm Algorithm. *Expert Systems with Applications*, vol. 39, no. 2, pp. 1811–1821, February 2012. (Cited in page 87).

[Kwong *et al.*, 1994] K.M. Kwong, W.A.and Passino, E.G. Laukonen, and S. Yurkovich. Expert Supervision of Fuzzy Learning Systems With Applications to Reconfigurable Control for Aircraft. In: *Proceedings of the 33rd IEEE Conference on Decision and Control*, vol. 4, pp. 4116–4121. 1994. (Cited in page 13).

[Kwong and Passino, 1996] W.A. Kwong and K.M Passino. Dynamically Focused Fuzzy Learning Control. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 26, no. 1, pp. 53–74, February 1996. (Cited in page 13).

[Lam and Lauber, 2013] H. K. Lam and J. Lauber. Membership-Function-Dependent Stability Analysis of Fuzzy-Model-Based Control Systems Using Fuzzy Lyapunov Functions. *Information Sciences*, vol. 232, no. 0, pp. 253–266, May 2013. (Cited in page 12).

[Layne and Passino, 1996] Jeffery R. Layne and Kevin M. Passino. Fuzzy Model Reference Learning Control. *Journal of Intelligent and Fuzzy Systems*, vol. 4, no. 1, pp. 33–47, 1996. (Cited in page 13).

[Layne and Passino, 1993] J.R. Layne and K.M. Passino. Fuzzy Model Reference Learning Control for Cargo Ship Steering. *IEEE Control Systems Magazine*, vol. 13, no. 6, pp. 23–34, December 1993. (Cited in page 13).

[Lennon and Passino, 1999] William K. Lennon and Kevin M. Passino. Intelligent Control for Brake Systems. *IEEE Transactions on Control Systems Technology*, vol. 7, no. 2, pp. 188–202, 1999. (Cited in page 13).

[Li *et al.*, 2009] Chaoshun Li, Jianzhong Zhou, Xiuqiao Xiang, Qingqing Li, and Xueli An. T-S Fuzzy Model Identification Based on a Novel Fuzzy C-Regression Model Clustering Algorithm. *Engineering Applications of Artificial Intelligence*, vol. 22, no. 4-5, pp. 646–653, June 2009. (Cited in pages 53, 56, and 57).

[Li *et al.*, 2013] Chengdong Li, Guiqing Zhang, Ming Wang, and Jianqiang Yi. Data-driven Modeling and Optimization of Thermal Comfort and Energy Consumption Using Type-2 Fuzzy Method. *Soft Computing*, vol. 17, no. 11, pp. 2075–2088, November 2013. (Cited in pages 124, and 125).

[Lim, 1995] C. M. Lim. Implementation and Experimental Study of a Fuzzy Logic Controller for DC Motors. *Computers in Industry*, vol. 26, no. 1, pp. 93–96, April 1995. (Cited in page 10).

[Lin and Lee, 1991] Chin-Teng Lin and C. S. George Lee. Neural-Network-Based Fuzzy Logic Control and Decision System. *IEEE Transactions on Computers*, vol. 40, no. 12, pp. 1320–1336, December 1991. (Cited in page 14).

[Lin and Lee, 1994] Chin-Teng Lin and C.S.G. Lee. Reinforcement Structure/Parameter Learning for Neural-Network-Based Fuzzy Logic Control Systems. *IEEE Transactions on Fuzzy Systems*, vol. 2, no. 1, pp. 46–63, February 1994. (Cited in page 14).

[Lin and Wai, 2001] Faa-Jeng Lin and Rong-Jong Wai. Hybrid Control Using Recurrent Fuzzy Neural Network for Linear Induction Motor Servo Drive. *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 1, pp. 102–115, February 2001. (Cited in page 15).

[Lin and Cunningham, 1995] Yinghua Lin and George A. Cunningham, III. A New Approach to Fuzzy-Neural System Modeling. *IEEE Transactions on Fuzzy Systems*, vol. 3, no. 2, pp. 190 –198, May 1995. (Cited in page 42).

[Liu *et al.*, 2008] Jianfeng Liu, Zhiwu Huang, Weirong Liu, Yingze Yang, and Haitao Tong. Locomotive Brake Control Method Based on T-S Fuzzy Modeling Predictive Control. In: *Proceedings of the 2008 Second International Symposium on Intelligent Information Technology Application*, pp. 602–607. IEEE Computer Society, Washington, DC, USA, 2008. (Cited in page 21).

[Ljung, 1987] Lennart Ljung. *System Identification Theory for the User*. Prentice–Hall, Inc, Englewood Cliffs, NJ, USA, 1987. (Cited in page 21).

[Lo *et al.*, 2007] C.H. Lo, P.T. Chan, Y.K. Wong, A.B. Rad, and K.L. Cheung. Fuzzy-Genetic Algorithm for Automatic Fault Detection in HVAC Systems. *Applied Soft Computing*, vol. 7, no. 2, pp. 554–560, March 2007. (Cited in pages 41, and 42).

[Lu and Tsai, 2007] Chi-Huang Lu and Ching-Chih Tsai. Generalized Predictive Control Using Recurrent Fuzzy Neural Networks for Industrial Processes. *Journal of Process Control*, vol. 17, no. 1, pp. 83–82, January 2007. (Cited in page 21).

[Mamdani, 1974] E. H. Mamdani. Application of Fuzzy Algorithms for Control of Simple Dynamic Plant. *Procedings of IEEE*, vol. 121, no. 12, pp. 1585–1588, December 1974. (Cited in page 10).

[Mamdani and Assilian, 1975] E. H. Mamdani and S. Assilian. An Experiment in Linguistic Synthesis With a Fuzzy Logic Controller. *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1–13, January 1975. (Cited in page 10).

[Mendes *et al.*, 2011] Jérôme Mendes, Rui Araújo, Pedro Sousa, Filipe Apóstolo, and Luís Alves. An Architecture for Adaptive Fuzzy Control in Industrial Environments. *Computers in Industry*, vol. 62, no. 3, pp. 364–373, April 2011. (Cited in pages 14, 30, 31, 88, 92, 97, and 120).

[Mendes *et al.*, 2014] Jérôme Mendes, Rui Araújo, Tiago Matias, Ricardo Seco, and Carlos Belchior. Automatic Extraction of the Fuzzy Control System by a Hierarchical Genetic Algorithm. *Engineering Applications of Artificial Intelligence*, vol. 29, no. 1, pp. 70–78, March 2014. (Cited in page 7).

[Mendes *et al.*, 2013a] Jérôme Mendes, Rui Araújo, and Francisco Souza. Adaptive Fuzzy Identification and Predictive Control for Industrial Processes. *Expert Systems with Applications*, vol. 40, no. 17, pp. 6964–6975, December 2013a. (Cited in pages 7, 111, and 113).

[Mendes *et al.*, 2013b] Jérôme Mendes, Rui Araújo, and Francisco Souza. Hierarchical Fuzzy Genetic Algorithm for Identification and Predictive Control. *Submitted to Soft Computing*, 2013b. (Cited in page 7).

[Mendes *et al.*, 2012] Jérôme Mendes, Francisco Souza, Rui Araújo, and Nuno Gonçalves. Genetic Fuzzy System for Data-Driven Soft Sensors Design. *Applied Soft Computing*, vol. 12, no. 10, pp. 3237–3245, october 2012. (Cited in page 7).

[Morningred *et al.*, 1992] J. Duane Morningred, Bradley E. Paden, Dale E. Seborg, and Duncan A. Mellichamp. An Adaptive Nonlinear Predictive Controller. *Chemical Engineering Science*, vol. 47, no. 4, pp. 755–762, 1992. (Cited in pages xxiii, 72, and 73).

[Moudgal *et al.*, 1995] V.G. Moudgal, W.A. Kwong, K.M. Passino, and S Yurkovich. Fuzzy Learning Control For a Flexible-Link Robot. *IEEE Transactions on Fuzzy Systems*, vol. 3, no. 2, pp. 199–210, May 1995. (Cited in page 13).

[Nelles, 2000] Oliver Nelles. *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*. Springer, 1 ed., December 2000. (Cited in page 4).

[Nie, 1995] Junhong Nie. Constructing Fuzzy Model by Self-Organizing Counterpropagation Network. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, no. 6, pp. 963–970, June 1995. (Cited in page 41).

[Passino *et al.*, 1995] K.M. Passino, E.G. Laukonen, and S. Yurkovitch. Expert Supervision of Fuzzy Learning Systems For Fault Tolerant Aircraft Control. *Proceedings of the IEEE*, vol. 83, no. 3, pp. 466–483, March 1995. (Cited in page 13).

[Pettersson *et al.*, 2007] F. Pettersson, N. Chakraborti, and H. Saxén. A Genetic Algorithms Based Multi-Objective Neural Net Applied to Noisy Blast Furnace Data. *Applied Soft Computing*, vol. 7, no. 1, pp. 387–397, January 2007. (Cited in pages 17, 41, and 42).

[Pishvaie and Shahrokhi, 2006] Mahmoud Reza Pishvaie and Mohammad Shahrokhi. Control of pH Processes Using Fuzzy Modeling of Titration Curve. *Fuzzy Sets and Systems*, vol. 157, no. 22, pp. 2983–3006, November 2006. (Cited in page 11).

[Precup *et al.*, 2012] Radu-Emil Precup, Marius L. Tomescu, Mircea-Bogdan Rădac, Emil M. Petriu, Stefan Preitl, and Claudia-Adina Dragoş. Iterative Performance Improvement of Fuzzy Control Systems for Three Tank Systems. *Expert*

Systems with Applications, vol. 39, no. 9, pp. 8288–8299, July 2012. (Cited in page 11).

[Procyk and Mamdani, 1978] T. J. Procyk and E. H. Mamdani. A Linguistic Self-Organising Process Controller. *Automatica*, vol. 15, no. 1, January 1978. (Cited in page 13).

[Qin and Badgwell, 1997] S. Joe Qin and Thomas A. Badgwell. An Overview of Industrial Model Predictive Control Technology. In: Jeffrey C. Kantor, Carlos E. Garcia, and Brice Carnahan (eds.), *Chemical Process Control-V: Assessment and New Directions for Research: Proceedings of the Fifth International Conference on Chemical Process Control, Tahoe City, California, January 7-12, 1996*, vol. 93 of *AIChE Symposium Series 316*, pp. 232–256. American Institute of Chemical Engineers, New York, NY, USA, 1997. (Cited in pages 3, and 20).

[Qin and Badgwell, 2000] S. Joe Qin and Thomas A. Badgwell. An Overview of Nonlinear Model Predictive Control Applications. In: Frank Allgöwer and Alex Zheng (eds.), *Nonlinear Model Predictive Control*, vol. 26, pp. 369–392. Springer-Verlag, 2000. (Cited in pages xvii, 3, and 4).

[Qin and Badgwell, 2003] S. Joe Qin and Thomas A. Badgwell. A Survey of Industrial Model Predictive Control Technology. *Control Engineering Practice*, vol. 11, no. 7, pp. 733–764, July 2003. (Cited in pages 3, and 20).

[Ramírez *et al.*, 2004] Mercedes Ramírez, Rodolfo Haber, Víctor Peña, and Iván Rodríguez. Fuzzy Control of a Multiple Hearth Furnace. *Computers in Industry*, vol. 54, no. 1, pp. 105–113, May 2004. (Cited in page 10).

[Rechenberg, 1973] I. Rechenberg. *Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Fommann-Holzboog, 1973. (Cited in page 16).

[Ren *et al.*, 2013] Qun Ren, Marek Balazinski, Krzysztof Jemielniak, Luc Baron, and Sofiane Achiche. Experimental and Fuzzy Modelling Analysis on Dynamic Cutting Force in Micro Milling. *Soft Computing*, vol. 17, no. 9, pp. 1687–1697, September 2013. (Cited in pages 124, and 125).

[Richalet, 1993] J. Richalet. Industrial Applications of Model Based Predictive Control. *Automatica*, vol. 29, no. 5, pp. 1251–1274, September 1993. (Cited in pages 2, and 17).

[Richalet *et al.*, 1978] J. Richalet, A. Rault, J.L. Testud, and J. Papon. Model Predictive Heuristic Control: Applications to Industrial Processes. *Automatica*, vol. 14, no. 5, pp. 413–428, September 1978. (Cited in page 20).

[Rossiter *et al.*, 1991] J. A. Rossiter, B. Kouvaritakis, and R. M. Dunnett. Application of Generalised Predictive Control to a Boiler-Turbine Unit for Electricity Generation. *IEE Proceedings Part D*, vol. 138, no. 1, pp. 59–67, 1991. (Cited in page 21).

[Rubaai *et al.*, 2007] Ahmed Rubaai, Abdul R. Ofoli, and Donatus Cobbinah. DSP-Based Real-Time Implementation of a Hybrid H-Infinity in Adaptive Fuzzy Tracking Controller for Servo-Motor Drives. *IEEE Transactions on Industry Applications*, vol. 43, no. 2, pp. 476–484, March-April 2007. (Cited in page 14).

[Sadrabadi and Zarandi, 2011] Mahmood Rezaei Sadrabadi and M. Hossein Fazel Zarandi. Identification of the Linear Parts of Nonlinear Systems for Fuzzy Modeling. *Applied Soft Computing*, vol. 11, no. 1, pp. 807–819, January 2011. (Cited in page 11).

[Salehia *et al.*, 2009] Shahin Salehia, Mohammad Shahrokhib, and Ali Nejatic. Adaptive Nonlinear Control of pH Neutralization Processes Using Fuzzy Approximators. *Control Engineering Practice*, vol. 17, no. 11, pp. 1329–1337, November 2009. (Cited in page 14).

[Sharkawy, 2010] Abdel Badie Sharkawy. Genetic Fuzzy Self-Tuning PID Controllers for Antilock Braking Systems. *Engineering Applications of Artificial Intelligence*, vol. 23, no. 7, pp. 1041–1052, October 2010. (Cited in page 16).

[Sivanandam and Deepa, 2007] S. N. Sivanandam and S. N. Deepa. *Introduction to Genetic Algorithms*. Springer Publishing Company, Incorporated, 2007. (Cited in page 33).

[Skrjanc and Matko, 2000] Igor Skrjanc and Drago Matko. Predictive Functional Control Based on Fuzzy Model for Heat-Exchanger Pilot Plant. *IEEE Transac-*

tions on Fuzzy Systems, vol. 8, no. 6, pp. 705–712, December 2000. (Cited in page 21).

[Souza *et al.*, 2013] Francisco A. A. Souza, Rui Araújo, Tiago Matias, and Jérôme Mendes. A Multilayer-Perceptron Based Method for Variable Selection in Soft Sensor Design. *Journal of Process Control*, vol. 23, no. 10, pp. 1371–1378, November 2013. (Cited in pages 41, 87, and 125).

[Spooner *et al.*, 2002] Jeffrey T. Spooner, Manfredi Maggiore, and Raúl Ordó nez. *Stable Adaptive Control and Estimation for Nonlinear Systems*. John Wiley Sons, Inc, New York, USA, 2002. (Cited in page 13).

[Su *et al.*, 2006] Chengli Su, Ping Li, and Shuqing Wang. A Direct Adaptive Predictive Functional Control Based on T-S Fuzzy Model. In: *Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications*, pp. 92–97. IEEE Computer Society, Washington, DC, USA, 2006. (Cited in page 21).

[Su *et al.*, 2012] Zhi-Gang Su, Pei hong Wang, Jiong Shen, Yu fei Zhang, and Lu Chen. Convenient T-S Fuzzy Model With Enhanced Performance Using a Novel Swarm Intelligent Fuzzy Clustering Technique. *Journal of Process Control*, vol. 22, no. 1, pp. 108–124, January 2012. (Cited in pages 124, and 125).

[Sugeno and Nishida, 1985] M. Sugeno and M. Nishida. Fuzzy Control of Model Car. *Fuzzy Sets and Systems*, vol. 16, no. 2, pp. 103–113, July 1985. ISSN 0165-0114. (Cited in page 10).

[Sugeno and Yasukawa, 1993] Michio Sugeno and Takahiro Yasukawa. A Fuzzy-Logic-Based Approach to Qualitative Modeling. *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 1, pp. 7–31, February 1993. (Cited in page 42).

[Takagi and Sugeno, 1985] Tomohiro Takagi and Michio Sugeno. Fuzzy Identification of Systems and Its Applications to Modeling and Control. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 15, no. 1, pp. 116–132, February 1985. (Cited in pages 11, 32, and 40).

[Tan *et al.*, 2004] Leong Ping Tan, Ahmad Lotfi, Eugene Lai, and J. B. Hull. Soft Computing Applications in Dynamic Model Identification of Polymer Extrusion

Process. *Applied Soft Computing*, vol. 4, no. 4, pp. 345–355, September 2004. (Cited in page 41).

[Taylan and Darrab, 2011] Osman Taylan and Ibrahim A. Darrab. Determining Optimal Quality Distribution of Latex Weight Using Adaptive Neuro-Fuzzy Modeling and Control Systems. *Computers & Industrial Engineering*, vol. 61, no. 3, pp. 686–696, October 2011. (Cited in page 15).

[Tham *et al.*, 1991] M. T. Tham, F. Vagi, A. J. Morris, and R. K. Wood. Multivariable and Multirate Self-Tuning Control: a Distillation Column Case Study. *IEE Proceedings Part D*, vol. 138, no. 1, pp. 9–24, 1991. (Cited in pages 20, and 124).

[Tzeng, 2010] Shian-Tang Tzeng. Design of Fuzzy Wavelet Neural Networks Using the GA Approach for Function Approximation and System Identification. *Fuzzy Sets and Systems*, vol. 161, no. 19, pp. 2585–2596, October 2010. (Cited in pages 17, and 41).

[Waewsak *et al.*, 2010] Chaiwat Waewsak, Annop Nopharatana, and Pawinee Chaiprasert. Neural-Fuzzy Control System Application for Monitoring Process Response and Control of Anaerobic Hybrid Reactor in Wastewater Treatment and Biogas Production. *Journal of Environmental Sciences*, vol. 22, no. 12, pp. 1883–1890, December 2010. (Cited in page 15).

[Wakabayashi *et al.*, 2009] C. Wakabayashi, M. Embiruçu, C. Fontes, and R. Kalid. Fuzzy Control of a Nylon Polymerization Semi-Batch Reactor. *Fuzzy Sets and Systems*, vol. 160, no. 4, pp. 537–553, February 2009. (Cited in page 10).

[Wang, 1992] Li-Xin Wang. Stable Adaptive Fuzzy Control of Nonlinear Systems. In: *Proceedings of the 31st IEEE Conference on Decision and Control*, pp. 2511–2516. December 1992. (Cited in page 13).

[Wang, 1996] Li-Xin Wang. Stable Adaptive Fuzzy Controllers With Application to Inverted Pendulum Tracking. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 26, no. 5, pp. 677–691, October 1996. (Cited in pages 13, and 31).

[Wang, 1997] Li-Xin Wang. *A Course in Fuzzy Systems and Control*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1997. (Cited in pages 10, 12, 24, 27, 28, 29, 32, 45, 50, 87, and 92).

[Wang and Mendel, 1992] Li-Xin Wang and J.M. Mendel. Fuzzy Basis Functions, Universal Approximation, and Orthogonal Least-Squares Learning. *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 807–814, September 1992. (Cited in pages 11, 21, 40, and 124).

[Wang *et al.*, 2010] Xian-Zhong Wang, Tao Zhang, and Lei He. Application of Fuzzy Adaptive Back-Propagation Neural Network in Thermal Conductivity Gas Analyzer. *Neurocomputing*, vol. 73, no. 4-6, pp. 679–683, January 2010. (Cited in page 15).

[Warne *et al.*, 2004] K. Warne, G. Prasad, S. Rezvani, and L. Maguire. Statistical and Computational Intelligence Techniques for Inferential Model Development: a Comparative Evaluation and a Novel Proposition for Fusion. *Engineering Applications of Artificial Intelligence*, vol. 17, no. 8, pp. 871–885, 2004. (Cited in page 41).

[Wen and Liu, 2009] Ji-Wei Wen and Fei Liu. Robust Model Predictive Control for Fuzzy Systems Subject to Actuator Saturation. In: *Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 6, pp. 400–404. IEEE Computer Society, Los Alamitos, CA, USA, 2009. (Cited in page 21).

[Wu *et al.*, 2012] Min Wu, Chunsheng Wang, Weihua Cao, Xuzhi Lai, and Xin Chen. Design and Application of Generalized Predictive Control Strategy With Closed-Loop Identification for Burn-Through Point in Dintering Process. *Control Engineering Practice*, vol. 20, no. 10, pp. 1065–1074, October 2012. (Cited in pages 124, and 125).

[Yasunobu and Miyamoto, 1985] S. Yasunobu and S. Miyamoto. Automatic Train Operation by Fuzzy Predictive Control. *Industrial Applications of Fuzzy Control*, 1985. (Cited in page 10).

[Ying, 1997] H. Ying. General MISO Takagi-Sugeno Fuzzy Systems with Simplified Linear Rule Consequent as Universal Approximators for Control and Modeling

Applications. *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 2, pp. 1335–1340, 1997. (Cited in pages 11, and 32).

[Yusof *et al.*, 2011] Rubiyah Yusof, Ribhan Zafira Abdul Rahman, Marzuki Khalid, and Mohd Faisal Ibrahim. Optimization of Fuzzy Model Using Genetic Algorithm for Process Control Application. *Journal of the Franklin Institute*, vol. 348, no. 7, pp. 1717–1737, September 2011. (Cited in pages 41, 42, 124, and 125).

[Zadeh, 1965] L. A. Zadeh. Fuzzy Sets. *Information and Control*, vol. 8, no. 3, pp. 338–353, June 1965. (Cited in page 9).

[Zadeh, 1971] L. A. Zadeh. Similarity Relations and Fuzzy Orderings. *Information Science*, vol. 3, no. 2, pp. 177–200, 1971. ISSN 0020-0255. (Cited in page 10).

[Zadeh, 1973] L. A. Zadeh. Outline of a New Approach to the Analysis of Complex Systems and Decision Processes. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, pp. 28–44, January 1973. (Cited in page 10).

[Zadeh, 1968] Lotfi A. Zadeh. Fuzzy Algorithms. *Information and Control*, vol. 12, no. 2, pp. 94–102, 1968. (Cited in page 10).

[Zhang and Morris, 1999] Jie Zhang and A.J. Morris. Recurrent Neuro-Fuzzy Networks for Nonlinear Process Modeling. *IEEE Transactions on Neural Networks*, vol. 10, no. 2, pp. 313–326, March 1999. (Cited in page 21).

[Zhao *et al.*, 2011] Dezong Zhao, Chunwen Li, and Jun Ren. Fuzzy Speed Control and Stability Analysis of a Networked Induction Motor System With Time Delays and Packet Dropouts. *Nonlinear Analysis: Real World Applications*, vol. 12, no. 1, pp. 273–287, February 2011. (Cited in page 11).

[Zhao *et al.*, 2010] Liang Zhao, Feng Qian, Yupu Yang, Yong Zeng, and Haijun Su. Automatically Extracting T-S Fuzzy Models Using Cooperative Random Learning Particle Swarm Optimization. *Applied Soft Computing*, vol. 10, no. 3, pp. 938–944, June 2010. (Cited in pages 41, 124, and 125).

[Zhou *et al.*, 2013] Q. Zhou, C.W. Chan, P. Tontiwachwuthikul, R. Idem, and D. Gelowitz. Application of Neuro-Fuzzy Modeling Technique for Operational Problem Solving in a CO2 Capture Process System. *International Journal of Greenhouse Gas Control*, vol. 15, no. 0, pp. 32–41, July 2013. (Cited in page 15).

[Zhou and Xub, 2001] Shang Ming Zhou and Li Da Xub. A New Type of Recurrent Fuzzy Neural Network For Modeling Dynamic Systems. *Knowledge-Based Systems*, vol. 14, no. 5-6, pp. 243–251, August 2001. (Cited in page 15).

[Zhu *et al.*, 1991] Q. M. Zhu, K. Warwick, and J. L. Douce. Adaptive General Predictive Controller for Nonlinear Systems. *IEE Proceedings Part D*, vol. 138, no. 1, pp. 33–40, 1991. (Cited in page 21).

[Ziaii *et al.*, 2012] Mansour Ziaii, Faramarz Doulati Ardejani, Mahdi Ziaei, and Ali A. Soleymani. Neuro-Fuzzy Modeling Based Genetic Algorithms for Identification of Geochemical Anomalies in Mining Geochemistry. *Applied Geochemistry*, vol. 27, no. 3, pp. 663–676, March 2012. (Cited in page 17).

[Zumberge and Passino, 1998] Jon Zumberge and Kevin M. Passino. A Case Study in Intelligent vs. Conventional Control For a Process Control Experiment. *Control Engineering Practice*, vol. 6, no. 9, pp. 1055–1075, September 1998. (Cited in page 13).

[Öztürk and Çelik, 2012] Nihat Öztürk and Emre Çelik. Speed Control of Permanent Magnet Synchronous Motors Using Fuzzy Controller Based on Genetic Algorithms. *International Journal of Electrical Power & Energy Systems*, vol. 43, no. 1, pp. 889–898, December 2012. (Cited in page 17).


FCT
Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA EDUCAÇÃO E CIÊNCIA


mais
CENTRO
Programa Operacional Regional do Centro


QUADRO
DE REFERÊNCIA
ESTRATÉGICO
NACIONAL


UNIÃO EUROPEIA
Fundo Europeu
de Desenvolvimento Regional


adi
agência de inovação