Neural - Fuzzy Approach for System Identification

Promotor : **Dr. ir. G. van Straten**
hoogleraar in de meet -, regel -, en systeemtechniek

B. - T. Tien

田 秉 才

Neural - Fuzzy Approach for System Identification

PROEFSCHRIFT

ter verkrijging van de graad van doctor
op gezag van de rector magnificus
van de Landbouwuniversiteit Wageningen,
dr. C. M. Karssen,
in het openbaar te verdedigen
op maandag 8 september 1997
des namiddags te vier uur in de Aula

Thesis title : Neural - Fuzzy Approach for System Identification
Author : Tien, Biing-Tsair
Total pages : 155
ISBN number : 90 - 5485 - 726 - 9
Subject headings : neural-fuzzy model / modeling / system identification / agriculture

This dissertation has been completed in partial fulfillment of the requirements of the Dutch Institute of Systems and Control DISC for graduate study.

NNO8201, 2308

Stellingen 【箴言】

\- 壹 -

Both from a structural and from a functional viewpoint, fuzzy systems can be regarded as a special type of feedforward neural networks.

(Chapter 2, this thesis)

\- 貳 -

Fortunetellers are one special type of fuzzy-rule-base model predictors. However, in contrast to control practice, when they feel well, most people don't need a predictor.

\- 參 -

Translation of foreign terms into Chinese is a kind of art, which shall try to keep both the meaning *and* the pronunciation as close as possible to the original ones. For example, 'fuzzy' can be translated as '乏晰', because '乏' means 'be short of' and '晰', the 'clearness'. Hence, '乏晰' in Chinese gives you a clear impression that fuzzy is short of clearness. Another interesting example of the translation of 'Holland' in Chinese is '荷蘭', where '荷' means the lotus and '蘭' the orchid. Surely, '荷蘭' presents you a beautiful imagination that Holland is a country full of flowers everywhere.

\- 肆 -

The orthogonal least squares algorithm makes it possible to prune redundant fuzzy rules from the prototype rule base and to assess the remaining weight parameters of the neural-fuzzy model by one-pass estimation. (Chapter 3, this thesis)

\- 伍 -

Computer related products that are MIT (Made In Taiwan) are as renowned as the Massachusetts Institute of Technology, due to the fact that R.O.C. (Republic Of China in Taiwan) also stands for Republic Of Computer in Taiwan.

- 陸 -

A good model accuracy can be achieved by just tuning the consequent weights of the neural-fuzzy model. (Chapter 4, this thesis)

- 柒 -

It is easier to split off an atom than to break down the bias of people. (Albert Einstein)

- 捌 -

The severely fluctuating weather in Holland stimulates the development of advanced climate control technology for Dutch greenhouses.

- 玖 -

Defuzzification of a Mamdani type of fuzzy model offers a clue to link the Takagi-Sugeno fuzzy model and the Mamdani fuzzy model, and thus enables linguistic interpretation of crisp consequents of the Takagi-Sugeno fuzzy rules in the same manner as Mamdani fuzzy rules. (Chapter 5, this thesis)

- 拾 -

Some officials are not really aware that 'it is nice to be important, but it is more important to be nice', so that they can easily destroy the good fame of all other nice and important officials.

Propositions attached to the Ph.D. thesis
Neural-Fuzzy Approach for System Identification
by Biing-Tsair Tien,
September 8, 1997, Wageningen, The Netherlands.

應用類神經網路及乏晰方法之系統辨識
田秉才

E-mail: TIENBT@CCMS.NTU.EDU.TW

應用類神經網路及乏晰方法
之系統辨識

作　　者　田　　秉　　才

指導教授　黑勒特　方　史達頓

博士論文

本論文之博士學位由
瓦赫尼根農業大學校長 卡森教授 授予

論文之公開答辯於
一九九七年九月八日下午四時
在大學禮堂歐臘舉行

ABSTRACT

Tien, B. - T. (1997) *Neural - Fuzzy Approach for System Identification.*
Ph.D. Thesis, Wageningen Agricultural University, Wageningen, The Netherlands
Key words : neural-fuzzy model, modeling, system identification, agriculture

In the real-world most processes have nonlinear and complex dynamics. Conventional modeling methods based on first principles are often cumbersome and time consuming, and approximations by linearized models are not always suitable. Thus, a nonlinear system identification procedure from observational data using artificial neural network and fuzzy models for black-box and gray-box modeling, respectively, can be an attractive alternative. In this thesis we consider the combination of both approaches to perform function approximation of unknown dynamic systems.

An integrated neural-fuzzy model, named NUFZY, is developed in this thesis, which combines advantages of both neural network and fuzzy modeling, and compensates for their weaknesses. The NUFZY system is a special type of neural network, which is characterized by partial connection in its first and second layers. Through its network connections the NUFZY system carries out a particular type of fuzzy reasoning. The transparency of network structure and the self-explanatory representation of fuzzy rules can be obtained from the NUFZY system. Moreover, it is functionally equivalent to a zeroth-order Takagi-Sugeno (T-S) fuzzy model, so that it can be seen as an universal function approximator to perform nonlinear mapping. Two existing learning methods are used to train the model parameters of the NUFZY system. One is the orthogonal least squares algorithms, which is used to find redundant fuzzy rules from the prototype rule base and to find the weight parameters of the NUFZY model by one-pass estimation. The other is the prediction error algorithms, which gives a fast adaptation of parameters of the NUFZY model. The developed NUFZY system is used to model several agricultural problems and results in sound performance, showing its capability for function approximation to deal with the real world modeling problems.

In this thesis we also discuss the possibility of obtaining linguistic interpretations of the crisp consequent from T-S fuzzy rules. This is relevant because the NUFZY model is a special case of the zeroth-order T-S fuzzy model. Promising results on the interpretability of the T-S fuzzy model have been attained. Besides, we investigate how to incorporate the *a priori* knowledge into the T-S fuzzy model in a systematic way. It has been shown that, when the qualitative *a priori* knowledge is taken into account in modeling, the resultant T-S fuzzy model becomes more robust in the extrapolation domain. This approach can be extended to neural-fuzzy modeling without difficulty.

ACKNOWLEDGEMENTS

The present Ph.D. study is financed by the Ministry of Education, Republic of China in Taiwan under subjects relating to 'Agricultural Automation' from September 1993 to August 1997. I would like express my gratitude to my government supporting me to study in Europe, where I choose the Netherlands, one of the most beautiful countries in the world.

I would like to thank my supervisor, Professor Gerrit van Straten, for his consistent support and encouragement. It is very joyful to discuss with him as I am always inspired by his constructive suggestions based on his abundant experience. His patience and thoughtfulness have been reflected in the way that he corrects my mistakes and leads me to the correct direction. I am very much deeply impressed by his rigorous and dedicated attitude in the research works. His intellectual guidance smooths away obstacles on the road of my Ph.D. programme and helps me succeed in completing this thesis. I owe him very much that is beyond my words.

I'd also like to thank my colleagues in the Systems and Control group of the Department of Agricultural, Environmental and Systems Technology (former Department of Agricultural Engineering and Physics), who create a comfortable atmosphere for discussion and offer efficient support to my Ph.D. research. They are Dr. Jan Bontsema, Dr. Ton van Boxtel, Dr. Karel Keesman, Dr. Gerard van Willigenburg, Dr. Eldert van Henten, Ing. Henk Veen, Ing. Rachel van Oosteghem, Ing. Kees van Asselt, Ir. Frank Tap, Ir. Leo Lukasse, Ir. Ilse Quirijns, and Mr. Paul van Espelo.

I couldn't forget Ard Duits, my Dutch room mate in the Rijnsteeg student flat when I first arrived the Netherlands. Except my Ph.D. study, Ard is so hearty to help me on any trouble I met. I had a very pleasant time with him in the stay of Rijnsteeg. The precious friendship will be remembered for ever.

Finally, I would like to dedicate this thesis to my wife, Yu-Tsai, and my son, Genso, as well as to our parents. From Taiwan to the Netherlands, Yu-Tsai was so brave to come alone to me in 1993. We made it true to marry in Wageningen by overcoming all possible resistance and difficulty. I want to express my sincere gratitude to Yu-Tsai's understanding and patience for these years, and to Genso's cooperation during the work of my thesis. Without their unlimited support and stimulation, I would not have come through this thesis.

獻給我的家人 — 玉猜、耕碩

以及我們的父母

CONTENTS

CONTENTS I

LIST OF FIGURES V

LIST OF TABLES VI

GLOSSARY VII

Variables for fuzzy systems VII

Indices and constants VIII

Operators and abbreviations VIII

Notations VIII

Vectors and matrices IX

1. GENERAL INTRODUCTION 1

1.1 Motivation - why neural-fuzzy modeling 2

1.1.1 Neural network modeling 2
1.1.2 Fuzzy modeling 3
1.1.3 Neural-fuzzy modeling 3

1.2 Objectives of this thesis 4

1.3 Thesis overview 5

2. CONSTRUCTION OF THE NEURAL-FUZZY SYSTEM - NUFZY 7

2.1 Basics of neural networks 8

2.2 Basics of fuzzy logic 10

2.2.1 Mamdani type of fuzzy rule 10
2.2.2 Takagi-Sugeno type of fuzzy rule 21

2.3 Construction of the NUFZY system 23

2.3.1 Architecture of the antecedent part of NUFZY 25
2.3.2 Architecture of the consequent part of NUFZY 27

2.4 CONCLUDING REMARKS 28

3. BATCH LEARNING OF THE NUFZY SYSTEM 29

3.1 INTRODUCTION 30

3.2 ORTHOGONAL LEAST SQUARES LEARNING 31

3.3 FUZZY RULE REDUCTION 33

3.4 EXAMPLES OF NONLINEAR STATIC SYSTEMS 35

3.4.1 Example 1 - synthetic nonlinear system 35
3.4.2 Example 2 - tomato production 39

3.5 EXAMPLES OF NONLINEAR DYNAMIC SYSTEMS - APPLICATIONS TO AGRICULTURAL PROBLEMS 40

3.5.1 Identification of lettuce growth process 41
3.5.2 Identification of greenhouse temperature dynamics 47

3.6 DISCUSSION 51

3.7 CONCLUSIONS 53

4. RECURSIVE LEARNING OF THE NUFZY SYSTEM 55

4.1 INTRODUCTION 56

4.2 SENSITIVITY DERIVATIVES OF THE NUFZY SYSTEM 57

4.2.1 Nodes and derivatives in Layer 1 of the NUFZY system 57
4.2.2 Nodes and derivatives in Layer 2 of the NUFZY system 60
4.2.3 Nodes and derivatives in Layer 3 of the NUFZY system 60
4.2.4 Sensitivity derivatives of the NUFZY system 61

4.3 PREDICTION ERROR ALGORITHM 63

4.3.1 Batch prediction error learning 63
4.3.2 Recursive prediction error learning 65

4.4 EXAMPLES 69

4.4.1 Example 1 - synthetic nonlinear system 69
4.4.2 Example 2 - prediction of tomato dry weight production 73

4.5 CONCLUSIONS 75

5. KNOWLEDGE INTERPRETATION FROM TAKAGI-SUGENO FUZZY MODELS 77

5.1 INTRODUCTION 78

5.2 COMPARISON OF TWO TYPES OF FUZZY RULES AND THEIR MODELS 79

5.2.1 Takagi-Sugeno type of fuzzy rule and model 79

5.2.2 Mamdani type of fuzzy rule and model 81

5.3 EXTENDED MAMDANI FUZZY RULE AND MODEL 89

5.4 INTERPRETATION OF THE IDENTIFIED T-S FUZZY RULES BY EM FUZZY RULES 97

5.4.1 To link the T-S fuzzy rule and the extended Mamdani fuzzy rule 97

5.4.2 To solve the parameters w and ρ 97

5.5 EXAMPLE 98

5.6 CONCLUSION AND DISCUSSION 101

6. INCORPORATING *A PRIORI* KNOWLEDGE INTO T-S FUZZY MODELING 103

6.1 INTRODUCTION 104

6.2 OPTIMIZATION BASED APPROACH 105

6.2.1 Formulation of the performance criterion 107

6.2.2 Estimation of the penalty weighting parameters 109

6.3 EXAMPLE 114

6.4 RESULTS 117

6.5 CONCLUSIONS 119

7. CONCLUSIONS 121

7.1 GENERAL CONCLUSIONS 122

7.2 FUTURE PERSPECTIVES 123

APPENDIX A OPTIMALITY OF THE GENERALIZED TAKAGI-SUGENO FUZZY MODEL 125

APPENDIX B DERIVATION OF THE SENSITIVITY DERIVATIVES OF THE NUFZY SYSTEM 127

B.1 NODE DERIVATIVES IN LAYER 1 OF THE NUFZY SYSTEM 127

B.2 NODE DERIVATIVES IN LAYER 2 OF THE NUFZY SYSTEM 129

B.3 NODE DERIVATIVES IN LAYER 3 OF THE NUFZY SYSTEM 131

B.4 THE SENSITIVITY DERIVATIVE OF THE NUFZY SYSTEM WRT TUNING PARAMETERS 132

APPENDIX C DERIVATION OF DERIVATIVES OF σ^2_{GCV} WITH RESPECT TO p 135

C.1 THE DERIVATIVE OF A^{-1} WITH RESPECT TO PENALTY WEIGHTING PARAMETERS p 135

C.2 THE DERIVATIVE OF e^Te WITH RESPECT TO PENALTY WEIGHTING PARAMETERS p 138

REFERENCES 141

SUMMARY 149

SAMENVATTING 151

結 論 153

CURRICULUM VITAE 155

LIST OF FIGURES

Figure 2.1: The neuron of neural networks 8

Figure 2.2: The structure of the NUFZY system 24

Figure 3.1: The development of SSE of the NUFZY model during the identification 36

Figure 3.2: Validation of the NUZFY system with Gaussian membership function in inputs 37

Figure 3.3: Validation of the NUZFY system with IMQ membership function in inputs . 37

Figure 3.4: Validation results by neural networks 38

Figure 3.5: Tomatoes growth process identified by the NUFZY model 40

Figure 3.6: Prediction of the lettuce dry weight by the first kind of NUFZY model 45

Figure 3.7: Prediction of the lettuce dry weight by the second kind of NUFZY model.... 46

Figure 3.8: Simulation results of the greenhouse temperature by the NUFZY model 50

Figure 4.1: Predicted output of the NUZFY system in a time-invariant case..................... 71

Figure 4.2: Variation of the consequent weight of the NUFZY model in the time - invariant case when $\mathbf{w}_{ini} = \mathbf{w}_{ols}$ 71

Figure 4.3: Variation of the consequent weight of the NUFZY model in the time- invariant case when $\mathbf{w}_{ini} = \mathbf{0}$ 72

Figure 4.4: Predicted output of the NUFZY model in the time-variant case..................... 73

Figure 4.5: The measured TDW of tomatoes with 95% confidence interval and the predicted TDW by the NUFZY model 74

Figure 5.1: Result of the extended Mamdani fuzzy model 100

Figure 6.1: The re-estimation procedure of the optimal penalty weighting parameters 113

Figure 6.2: The input/output membership functions and outputs of the T-S fuzzy model and the default Mamdani fuzzy model 116

Figure 6.3: The validation results of T-S fuzzy models incorporating different *a priori* knowledge 118

LIST OF TABLES

Table 2.1: The commonly used T-norm and S-norm operators 12

Table 6.1: The optimization result of T-S fuzzy models incorporating different *a priori* knowledge 117

GLOSSARY

The following general symbols and notations are used throughout this thesis. Some may have different meanings locally, but they shall be clear from the text.

VARIABLES FOR FUZZY SYSTEMS

x_i	the i^{th} input variable
X_i	universe of discourse of x_i
X	joint universe of discourse of all X_i
A^r_i	fuzzy set for X_i in the antecedent of the r^{th} fuzzy rule
$A^r_{ki}(x_i)$	the ki^{th} fuzzy set of input x_i in the antecedent of the r^{th} fuzzy rule
y_n	the n^{th} output variable
Y_n	universe of discourse of y_n
Y	joint universe of discourse of all Y_n
B^r_n	fuzzy set for Y_n in the consequent of the r^{th} fuzzy rule
y	single output variable
$B^r_j(y)$	the j^{th} fuzzy set of output y in the consequent of the r^{th} fuzzy rule
x_i'	the i^{th} numerical/measured input variable
A_i'	fuzzy set with singleton membership function resulted by x_i'
A'	joint fuzzy set of all premise fuzzy sets A_i'
$B^{r}{}'(y)$	resultant consequent fuzzy set of y based on implication of all premise fuzzy relations of the r^{th} fuzzy rule
B'	resultant consequent fuzzy set based on aggregation of all $B^{r}{}'(y)$
$fr_{i,r}(x_i)$	the i^{th} fuzzy relation of the fuzzy proposition 'x_i is $A^r_{ki}(x_i)$' in the antecedent of the r^{th} fuzzy rule
$fr_{j,r}(y)$	the j^{th} fuzzy relation of the fuzzy proposition 'y is $B^r_j(y)$' in the consequent of the r^{th} fuzzy rule
frp_r	conjunctive/disjunctive fuzzy relation of the antecedent part of the r^{th} fuzzy rule
FR^r	implicated fuzzy relation of the r^{th} fuzzy rule
FR	aggregative fuzzy relation of all fuzzy rules
a^r_j	the integrated area of the j^{th} fuzzy set of output y in the r^{th} fuzzy rule
a'	resultant active area of all active fuzzy sets of B(y)
m^r_j	the integrated first moment of the j^{th} fuzzy set of output y in the r^{th} fuzzy rule
m'	resultant active first moment of all active fuzzy sets of B(y)

INDICES AND CONSTANTS

i	denotes the i^{th} input variable x_i; $i = 1, .., ni$
ni	the total number of input variables
ki	denotes the ki^{th} membership function of x_i; $ki = 1, .., N_i$
N_i	the total number of membership functions of input x_i
n	denotes the n^{th} output variable y_n; $n = 1, .., nb$
nb	the total number of output variables
j	denotes the j^{th} membership function of single output y; $j = 1, .., N_b$
N_b	the total number of membership functions of output y
r	denotes the r^{th} fuzzy rule; $r = 1, .., R$
R	the total number of all fuzzy rules, is equal to $\prod_{i=1}^{ni} N_i$
m	denotes the m^{th} membership function of the set that stacks all input membership functions; $m = 1, .., M$
M	the total number of all input membership functions, is equal to $\sum_{i=1}^{ni} N_i$
t	denotes the t^{th} pattern of input $\mathbf{x}$ from the training set; $t = 1, .., np$
np	the total number of training set of input $\mathbf{x}$
q	denotes the q^{th} pattern of input $\mathbf{x}_s$ from the training set; $q = 1, .., ns$
ns	the total number of training set of input $\mathbf{x}_s$
k	denotes the k^{th} pattern of input $\mathbf{x}_a$ from the training set; $k = 1, .., na$
na	the total number of training set of input $\mathbf{x}_a$
nv	the total number of validation data set

OPERATORS AND ABBREVIATIONS

T	fuzzy T-norm operation
S	fuzzy S-norm operation
S_a	fuzzy S-norm operation used for aggregation
$\mathbf{I}$	fuzzy implication
$\mathbf{I}_{CI}$	fuzzy implication complies with classical implication
$\mathbf{I}_{CC}$	fuzzy implication complies with classical conjunction
cog	centroid of gravity method for defuzzification
T-S	Takagi-Sugeno type of fuzzy rule / model
EM	extended Mamdani type of fuzzy rule / model

NOTATIONS

$\Phi_{ki}(x_i)$	the ki^{th} membership node of the input x_i
$\mu_{ki}(x_i)$	the ki^{th} membership function of the input x_i; or denoted by α_m
$c_{ki}(x_i)$	the ki^{th} center of the membership function $\mu_{ki}(x_i)$

$\sigma_{ki}(x_i)$	the kith bandwidth of the membership function $\mu_{ki}(x_i)$
$\Re^r$	the rth rule node
ν_r	the firing strength of the rth fuzzy rule
$\bar{\nu}_r$	the normalized ν_r
w_{rn}	the nth consequent weight of the output with respect to the rth fuzzy rule
$\hat{y}_n$	the nth prediction output of the NUFZY system
$\boldsymbol{R}^r_{(M)}$	the rth fuzzy rule of the Mamdani type of fuzzy rule
$\boldsymbol{R}^r_{(TS)}$	the rth fuzzy rule of the Takagi-Sugeno type of fuzzy rule
$\boldsymbol{R}^r_{(EM)}$	the rth fuzzy rule of the extended Mamdani type of fuzzy rule
ρ_{rj}	the jth consequent significant level in the rth fuzzy rule
ζ	the ratio of np to ns
η	the ratio of np to na
σ^2_{GCV}	the generalized cross-validation criterion
γ	the update gain in the prediction error algorithm
λ	the penalty weighting parameter accounts for penalty caused by non-smoothness of the T-S fuzzy model; or, a forgetting factor in the prediction error algorithm
β	the penalty weighting parameter accounts for penalty caused by violating soft constraints
α	the penalty weighting parameter accounts for penalty caused by mismatch between the T-S fuzzy model and a default model; or, the stacked membership value of $\mu_{ki}(x_i)$

VECTORS AND MATRICES

Vector is denoted by bold font with lower case letter, for example,

$\mathbf{c}$	$= [c_1 .. c_m .. c_M]^T \; (= \theta_c)$
$\boldsymbol{\sigma}$	$= [\sigma_1 .. \sigma_m .. \sigma_M]^T \; (= \theta_\sigma)$
$\mathbf{w}_n$	$= [w_{1n} .. w_{rn} .. w_{Rn}]^T$
$\boldsymbol{\varpi}$	$= [\mathbf{w}_1^T .. \mathbf{w}_n^T .. \mathbf{w}_{nb}^T]^T \; (= \theta_\varpi)$
$\boldsymbol{\theta}$	$= [\boldsymbol{\varpi}\; \mathbf{c}\; \boldsymbol{\sigma}]^T$
$\mathbf{v}$	$= [\nu_1 .. \nu_r .. \nu_R]^T$
$\boldsymbol{\rho}_r$	$= [\rho_{r1} .. \rho_{rj} .. \rho_{rNb}]^T$
$\mathbf{x}$	$= [x_1 .. x_i .. x_{ni}]^T$; or, $= [x(1) .. x(t) .. x(np)]^T$
$\mathbf{y}$	$= [y_1 .. y_n .. y_{nb}]^T$; or, $= [y(1) .. y(t) .. y(np)]^T$
$\mathbf{x}_a$	$= [x_a(1) .. x_a(k) .. x_a(na)]^T$
$\mathbf{y}_a$	$= [y_a(1) .. y_a(k) .. y_a(na)]^T$
$\mathbf{x}_s$	$= [x_s(1) .. x_s(q) .. x_s(ns)]^T$
$\mathbf{y}_s$	$= [y_s(1) .. y_s(q) .. y_s(ns)]^T$
$\mathbf{z}$	$= [z_1 .. z_j .. z_{Nb}]^T$
$\mathbf{p}$	$= [\lambda\; \beta\; \alpha]^T$

Matrix is denoted by bold font with upper case letter, for example,

$\mathbf{W}$ $= [\mathbf{w}_1 \,; .. \, \mathbf{w}_n \,; .. \, \mathbf{w}_{nb}]_{(R\times nb)}$ with $\mathbf{w}_n = [w_{1n} \, .. \, w_{rn} \, .. \, w_{Rn}]^T$

$$\boldsymbol{\Psi}_{\varpi} = \left[\frac{\partial \hat{\mathbf{y}}}{\partial \theta_{\varpi}}\right] = \begin{bmatrix} \bar{\mathbf{v}} & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \vdots & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \bar{\mathbf{v}} & \cdots & \vdots \\ \vdots & \cdots & \vdots & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \cdots & \bar{\mathbf{v}} \end{bmatrix}_{((nb\cdot R)\times nb)}$$

$$\boldsymbol{\Psi}_{c} = \left[\frac{\partial \hat{\mathbf{y}}}{\partial \theta_{c}}\right]_{(M\times nb)}$$

$$\boldsymbol{\Psi}_{\sigma} = \left[\frac{\partial \hat{\mathbf{y}}}{\partial \theta_{\sigma}}\right]_{(M\times nb)}$$

$\boldsymbol{\Psi}$ $= [\boldsymbol{\Psi}_{\varpi} \,; \boldsymbol{\Psi}_{c} \,; \boldsymbol{\Psi}_{\sigma}]_{((nb\cdot R+2M)\times nb)}$

$\mathbf{I}_R$ R×R identity matrix (note, $\mathbf{I}_{R\times 1}$ is a R×1 vector of elements of ones)

RM R×M relationship matrix (note, denoted by Italic and bold font)

1. GENERAL INTRODUCTION

The Master said "Is it not pleasant to learn with a constant perseverance and application? Is it not delightful to have friends coming from distant quarters? Is he not a man of complete virtue, who feels no discomposure though men may take no note of him?".

- Confucius (Kung Fu-Tze), The Lun Yu, Analects of Confucius

子曰：學而時習之，不亦悅乎？有朋自遠方來，不亦樂乎？人不知而不慍，不亦君子乎？

論語學而篇

1.1 MOTIVATION - WHY NEURAL-FUZZY MODELING

A model is used to represent the essential aspects of an existing system (or process) in a usable form, with which the underlying input-output relations can be approximated. Many modern control designs demand precise and reliable models of the controlled processes in order to achieve a good performance. Most real-world processes have nonlinear and complex dynamics. Hence, if the system is to be operated over a wide range of operating conditions, the common linearization approach is not appropriate. Conventional methods of constructing nonlinear models from first principles are time consuming and require a level of knowledge about the internal functioning of the system that is often not available. Consequently, in such cases a nonlinear system identification procedure from observational data is a more attractive alternative. In such a procedure, first parameterized models (i.e. model structures) have to be postulated, the best model can then be found by optimizing parameters with respect to certain criteria across a set of input-output observations. If the model structures to be investigated are purely chosen from a set of mathematically convenient structures (e.g., autoregressive moving average, ARMA model), without incorporation of knowledge about the internal functioning, this is called black-box modeling. A black-box model is a model with flexible model structure which can be used to approximate a large variety of different unknown systems [54] . More specifically, the goal of black-box modeling is to perform function approximation of the unknown dynamic system using a set of observation data. In case that some qualitative *a priori* information can be used in the above modeling procedure, it is sometimes called gray-box modeling. Artificial neural network modeling and fuzzy modeling are typical examples of black-box and gray-box modeling, respectively. In the following, we will first analyze these two types of modeling.

1.1.1 Neural network modeling

Artificial neural networks (or 'neural networks' for short) have been attracting growing interest in the past decade and have been successful in various applications of nonlinear system identification and control problems, e.g., [7] [9] [10] [12] [48] [76] . Good surveys are given in [28] [29] . A thorough study of viewing neural network modeling as nonlinear system identification can be found in [52] .

Neural networks perform nonlinear mapping from the space of independent variables to the space of dependent variables by parallel architectures, which comprise processing units that communicate the data flow through weighted connections. The appealing features of neural network modeling lies in its approximation ability and learning capabilities. Based on the Stone-Weierstrass theorem, it has been shown in [27] [82] that multilayer feedforward networks can approximate any continuous function to arbitrary accuracy, provided sufficient hidden units are available. In this sense, multilayer feedforward networks belong to a class of universal approximators. Besides, several algorithms of learning/tuning model parameters have been developed and can be readily applied to neural networks [1] [39] [52] . However, it is difficult to interpret the information representation from the internal configuration of neural networks directly. Their homogeneous structure also impedes the use of any qualitative *a priori* knowledge.

1.1.2 Fuzzy modeling

In contrast to neural network modeling, fuzzy modeling is capable of processing available expert knowledge or experience which can be expressed in the form of a set of linguistic '*IF - THEN*' fuzzy rules and graded membership functions. It is therefore user-friendly and provides comprehensive knowledge representation. In terms of approximation ability, it has also been shown in [6] [80] that fuzzy systems can be universal approximators, like neural networks. So, fuzzy systems can be used to pursuit a high precision of function approximation. Yet, another appealing characteristic of the fuzzy model which is different from the neural network model is often neglected, i.e., the ability to deal with imprecise information by means of fuzzy rules generated from accumulated experience of human beings. This means that the fuzzy model is equipped with advantages over the neural network model, both in the transparent representation of knowledge and the ability to deal with imprecise information. Fuzzy models have been considered useful when confronted with systems whose underlying dynamics are unknown or too complex for analysis by conventional mathematical methods, e.g., [58] [60] [61] [62] [66] [67] .

Since conventional fuzzy reasoning is performed by a set of fixed fuzzy rules given by experts in order to carry out the function of static mapping, it is, however, usually difficult to modify the fuzzy rules. This indicates that the learning/tuning ability of conventional fuzzy systems is restricted, a characteristic opposite to the case of neural networks. Moreover, in the literature the handling of *a priori* knowledge of problems under study is *ad hoc*, and its use is unclear, so that it is often hard to set up prototype fuzzy rules for modeling. Alternatively, one can use a set of observation data to generate fuzzy rules, resulting in a data-driven fuzzy model [79] [81] . This approach creates the possibility of training fuzzy models, in the same spirit of training neural networks. It is worthy to note that, among the data-driven fuzzy models, the Takagi-Sugeno type of fuzzy model with crisp terms in its consequent is commonly adopted. Because of the crisp terms in the consequent, however, it is not easy to associate a full linguistic interpretation to the fuzzy rules from the Takagi-Sugeno fuzzy model.

The above comparison indicates that, in addition to the functional equivalence between the fuzzy model and the neural network model, one may try to seek the similarities between their structures and hopefully, to make use of their advantages and to make up for their weaknesses. This brings to mind the idea of combining both paradigms to create an integrated neural-fuzzy model.

1.1.3 Neural-fuzzy modeling

Although neural networks and fuzzy systems stem from different origins, they share the same property of parallel processing, and both can serve as universal approximators to perform nonlinear mapping. The recognition of the functional equivalence of both as universal approximators has prompted a new research to inject new driving forces from the field of neural networks into the 'fuzzy' discipline, and vice versa. That is, one attempts to combine the transparent representation of the fuzzy system and the learning capability of the neural networks in a unified framework, thus giving rise to an integrated neural-fuzzy or fuzzy-neural model, like [26] [31] [42] .

What can one gain from the integrated model? The integrated neural-fuzzy model, as proposed by [21] , is expected to be able to carry out the so-called 'IQ^2' reasoning (intelligent qualitative and quantitative reasoning). This means that the qualitative reasoning is based on fuzzy logic, and the adaptive numerics is quantitatively processed via neural networks. In other words, the integrated system can be seen as either an advanced state in the evolution of conventional fuzzy systems - being able to perform data-driven optimization - or as an extension of neural networks - realizing the integration of rule based knowledge [23] .

This thesis is primarily motivated by the benefits to be gained when the integrated neural-fuzzy model combines advantages of both paradigms and concurrently compensates for their weaknesses. We will, next, state further details of our objective and highlight some required properties of our integrated neural-fuzzy model, which are different from the existing ones.

1.2 OBJECTIVES OF THIS THESIS

One of the objectives of this thesis is first to construct an integrated neural-fuzzy model in order to perform function approximation of an unknown system via given input-output observations. In addition to obtaining a good accuracy of the modeling, the neural-fuzzy model shall fulfill the following requirements:

- Efficient implementation - the developed modeling techniques from either theoretical or computational aspects shall be easily and readily applicable to the developed neural-fuzzy model. This suggests that the construction of the model and the tuning of its parameters must be kept simple.

- Good generalization - the integrated neural-fuzzy model shall be able to deal with unseen inputs. By means of using correct fuzzy rules in the modeling, the resultant model will become robust and be capable of having good extrapolation to some distance.

- Transparency and interpretability - the knowledge representation of the integrated model shall be transparent to help users to understand the underling characteristics of the unknown system. Fuzzy rules deduced from the internal structure of the neural-fuzzy model can be interpreted in a linguistic way such that they benefit the validation of the local behaviors of the model.

- Ability to incorporate *a priori* knowledge - since there is much useful qualitative information concerning certain aspects of system behavior and operation, the neural-fuzzy model shall be able to utilize these different knowledge sources as much as possible.

It will be shown that the developed integrated neural-fuzzy model can, in fact, be deemed as a zeroth-order Takagi-Sugeno fuzzy model. Hence, issues of interpretability of its fuzzy rules and the incorporation of *a priori* knowledge are examined with the zeroth-order Takagi-Sugeno fuzzy model. Therefore, the second objective of this thesis is devoted to investigate the feasibility of obtaining transparent interpretations of the Takagi-Sugeno fuzzy rules and to investigate how to incorporate *a priori* knowledge into the Takagi-Sugeno fuzzy model.

1.3 THESIS OVERVIEW

This thesis comprises published papers and internal reports, some of which have been rewritten for the sake of easy reading. They are arranged in sequential chapters to be in line with the above objectives. In this section, we outline the organization of this thesis and point out the contributions made in each chapter.

Chapter 2 describes the fundamentals of neural networks and fuzzy logic. The contribution is the systematic establishment of an integrated neural-fuzzy system, named NUFZY. The NUFZY system is a simplified fuzzy system represented by the zeroth-order Takagi-Sugeno fuzzy model. It is a special type of neural network characterized by partial connections in its first and second layers. Through the network connections the NUFZY system performs a particular type of fuzzy reasoning. Yet, this does not restrict its ability of function approximation.

Chapter 3 is devoted to the estimation of weight parameters of the NUFZY model in an off-line fashion via a batch of observation data. Our contribution in this chapter is to use the orthogonal least squares algorithm to detect redundant fuzzy rules in the prototype fuzzy rule base, while, at the same time, finding the weight parameters of the NUFZY model by one-pass estimation. We also use several agricultural examples to illustrate the identification ability of the NUFZY model.

Chapter 4 demonstrates the use of the prediction error algorithm to tune parameters of the NUFZY model in a recursive manner (which is useful for on-line applications). The contribution of this chapter is in obtaining the sensitivity derivatives of the NUFZY system so that they can be easily applied to the recursive prediction error algorithm to attain a fast adaptation of model parameters. Examples are presented to show that good model accuracy can be obtained by merely tuning the consequent weight parameters of the NUFZY model.

Chapter 5 compares two types of fuzzy rules and their models. The result evokes the possibility to interpret fuzzy rules deduced from the zeroth-order Takagi-Sugeno (T-S) fuzzy model in a linguistic way. In our analysis, it is found that a fuzzy model has a natural property of dual representations, i.e., the defuzzified output can be represented as a linear function either of system inputs (like the T-S fuzzy model), or, of system outputs (like the Mamdani fuzzy model). This applies to both the

Takagi-Sugeno fuzzy model and the Mamdani fuzzy model. This property implicitly allows the transfer of the above two types of models to each other, thus enabling a linguistic interpretation of the T-S fuzzy rule. We also introduce a new parameter, named consequent significance level, to the ordinary Mamdani fuzzy model. This results in an extended Mamdani fuzzy model, which has a more flexible modeling ability compared to the ordinary Mamdani fuzzy model.

Chapter 6 illustrates an optimization approach to systematically incorporate the *a priori* knowledge into a Takagi-Sugeno fuzzy model. Our contribution lies in the application of the idea to formulate additional *a priori* knowledge as constraint terms imposed to the criterion function to be minimized. In particular, it is shown that if the knowledge about the system behavior outside the identification data range is expressed in the form of a qualitative Mamdani fuzzy model, then this model can be incorporated in the objective function of the parameter estimation problem as an additional penalty term. Thus, the estimation of the parameters of the T-S fuzzy model from the identification data is constrained by the involvement of *a priori* knowledge. As a consequence, the resultant fuzzy model becomes more robust in the extrapolation domain.

Chapter 7 concludes this thesis and suggests future research prospects.

2. CONSTRUCTION OF THE NEURAL-FUZZY SYSTEM - NUFZY

There are three friendships which are advantageous, and three which are injurious. Friendship with the uplight; friendship with the sincere; and friendship with the man of much observation : - these are advantageous. Friendship with the man of specious airs; friendship with the insinuatingly soft; and friendship with the glib-tongued : - these are injurious.

- Confucius (Kung Fu-Tze), The Lun Yu, Analects of Confucius

子曰：益者三友，損者三友；友直，友諒，友多聞；益矣。友便辟，友善柔，友便佞；損矣。 論語季氏篇

This chapter gives the basics of neural networks and fuzzy logic. Fuzzy reasoning can be cast into a feedforward network structure to perform the fuzzy inference procedure. The present study will regard the role of neuron units and node connections of the neural networks as joints and bonds which form the body of fuzzy reasoning. In other words, the neural network performs as a vehicle, in which fuzzy logic based reasoning is embedded, to achieve the goal of function approximation. First, a brief introduction of neural networks is given in section 2.1, followed by a more detailed, but conceptual, review of fuzzy logic reasoning in section 2.2. Based on the understanding of both neural networks and fuzzy logic, we will construct an integrated neural-fuzzy system, named NUFZY system, in section 2.3. Concluding remarks are addressed in section 2.4. Readers who are familiar with neural nets and fuzzy reasoning may proceed directly to section 2.3.

2.1 BASICS OF NEURAL NETWORKS

A typical neural network consists of a basic unit called 'neuron', which drives some finite inputs of connections represented by weighted values from the preceding layer of units and whose output is connected to the next layer of units. The i^{th} neuron of layer k is depicted in Figure 2.1.

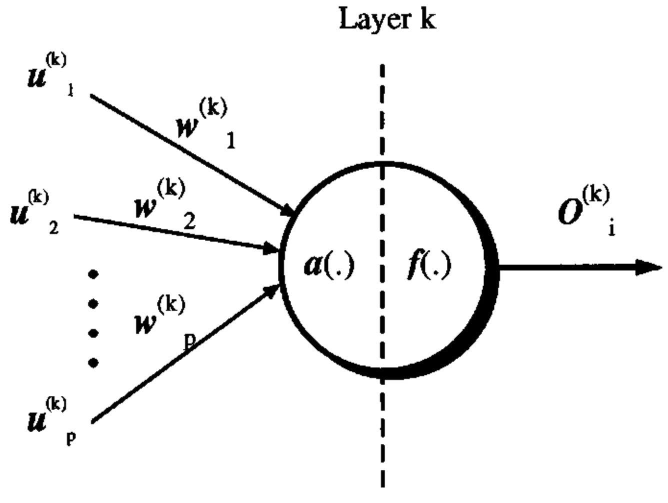

Figure 2.1: The fundamental unit of the neural networks - neuron.

Where $u^{(k)}_j$ represents the j^{th} input, associated with the weight $w^{(k)}_j$, to the i^{th} neuron of layer k. Subscript $j = 1, .., p$, p is the total number of inputs of the preceding layer that connects to the neuron of this layer. Superscript (k) denotes the layer number and $o^{(k)}_i$ denotes the i^{th} neuron output of layer k. The neuron of a feedforward neural network consists of a summator and a nonlinear activation function. The summator associated with p inputs of the preceding units is a function $a(\cdot)$ which serves to combine information from all nodes of the preceding layer.

This function then provides the net-input to the nonlinear activation function in this node, i.e.,

$$\text{net-input} = a^{(k)}(u^{(k)}_1, u^{(k)}_2, \ldots, u^{(k)}_p ; w^{(k)}_1, w^{(k)}_2, \ldots, w^{(k)}_p) = \Sigma_{(i=1,\ldots,p)} u^{(k)}_i w^{(k)}_i \tag{2.1}$$

The nonlinear activation function denoted as $f(\cdot)$ maps the net-input onto a bounded interval. Several nonlinear activation functions have been proposed for a neuron in the literature [1] [76] . Typical nonlinear activation functions, for instance, are

(1) threshold function (or, hard limitor)

$$f(a) = \begin{cases} 1 & \text{if } a>0 \\ 0 & \text{else} \end{cases} \tag{2.2}$$

(2) sigmoidal function

$$f(a) = (1 + exp\,(-a))^{-1} \tag{2.3}$$

(3) radial basis function (RBF)*

Some different choices of radial basis functions are possible [9] [11] [40] , such as

$$f(r) = r^2 \log(r) \quad : \text{thin-plate-spline function} \tag{2.4.a}$$

$$f(r) = exp(-(r^2/\sigma^2)/2) \quad : \text{Gaussian function} \tag{2.4.b}$$

$$f(r) = (r^2+\sigma^2)^{1/2} \quad : \text{multiquadratic function} \tag{2.4.c}$$

$$f(r) = (r^2+\sigma^2)^{-1/2} \quad : \text{inverse multiquadratic function (IMQ)} \tag{2.4.d}$$

where $r = \| x - c \|$ is the Euclidean distance from a point x of the input to the center c of the RBF. The parameters c and σ denote the center and width of RBF, respectively.

Hence, the node output is then given by

$$\text{output} = O^{(k)}_i = f(a) \;\; (\text{or } f(r))$$

It is noted that, among the above nonlinear activation functions, the Gaussian and inverse multiquadratic functions tend to zero outside the region where they are centered, and are therefore most suitable for the purpose of bounded output. In the present work, we will use these two radial basis functions as nonlinear activation functions to develop the network-like fuzzy logic system because they offer an additional advantage that they can represent the fuzzy membership functions in a fuzzy system.

* Neural networks that use radial basis functions are usually referred to as radial basis function networks.

2.2 BASICS OF FUZZY LOGIC

Systems based on fuzzy logic [85] are characterized by a set of *IF-THEN* fuzzy rules, in which the system behaviors can be described and analyzed by linguistic variables [86] . Each fuzzy rule, either gathered from accumulated experience or identified from empirical data, represents the local input-output underlying relations of systems. Based on the consequence used in fuzzy rules, we can distinguish two types of fuzzy rules : Mamdani type of fuzzy rule and Takagi-Sugeno type of fuzzy rule. The Mamdani type of fuzzy rule was the first reported fuzzy application in control in the literature since 1975 [45] . The later development of the Mamdani type of fuzzy reasoning is based on the composition of fuzzy relations, which forms the foundation of fuzzy inference by Mamdani type of fuzzy rules. Hence, in what follows, we will start with an introduction of the important concepts used in the Mamdani type of fuzzy rule. Then the variant fuzzy rule, the Takagi-Sugeno type of fuzzy rule [62] , is discussed. The term Mamdani type used here does not refer to the original operators used by Mamdani and his co-workers, but refers to the fact that the fuzzy rules have fuzzy propositions as their consequence, and that the implication is represented by T-norm conjunction (stated later). Furthermore, in this section we do not attempt to give all details of fuzzy logic, but will focus on introducing some concepts that might help to understand the spirit of fuzzy logic and to construct the integrated neural-fuzzy system. Therefore, most definitions and terminologies used in this section are adapted from the excellent work of [30] . Of course, many other references of fuzzy logic can be found, for instance, in [22] [37] [38] [39] [41] [50] [51] [66] [67] [83] [89] . Besides, the extension of fuzzy logic to approximation reasoning can be found in [87] [88] .

2.2.1 Mamdani type of fuzzy rule

A fuzzy rule consists of two parts, the antecedent and the consequent, which correspond to *IF* and *THEN* statements in the fuzzy rule, respectively. Consider a multi-input-multi-output (MIMO) fuzzy system, which has ni input variables x_i , i = 1, .. , ni, and nb output variables y_n , n = 1, .. , nb. A typical Mamdani type of fuzzy rule can be expressed as

$$\begin{array}{lll} \boldsymbol{R}_{(M)}: & \textbf{\textit{IF}} & x_1 \text{ is } A_1 \textbf{ AND} \,..\, x_i \text{ is } A_i \textbf{ AND} \,..\, x_{ni} \text{ is } A_{ni} \\ & \textbf{\textit{THEN}} & y_1 \text{ is } B_1, .., y_n \text{ is } B_n, .., y_{nb} \text{ is } B_{nb} \end{array}$$

where A_i and B_n are fuzzy sets for the universe of discourses of input and output, X_i and Y_n, respectively. The subscript (M) denotes a Mamdani type of fuzzy rule and distinguishes it from the Takagi-Sugeno fuzzy rule to be discussed later. Based on fuzzy relation analysis, it was shown that the above MIMO fuzzy system can be decomposed into nb multi-input-single-output (MISO) fuzzy systems, in which the fuzzy rule has only one output variable in its consequent [41] . Hence, the fuzzy rule of such a MISO fuzzy system can be simply expressed as

$$\boldsymbol{R}_{(M)}: \textbf{\textit{IF}}\ x_1 \text{ is } A_1 \textbf{ AND} \,..\, x_i \text{ is } A_i \textbf{ AND} \,..\, x_{ni} \text{ is } A_{ni}\ \textbf{\textit{THEN}}\ y \text{ is } B$$

Without loss of generality, in this section we will only consider the MISO fuzzy system to describe the following properties of fuzzy logic. Using more detailed notations to denote the above fuzzy rule of a MISO fuzzy system, it can be rewritten as below.

$$\boldsymbol{R}^{r}_{(M)}: \textbf{\textit{IF}}\ x_1 \text{ is } A^{r}_{k1}(x_1)\ \textbf{AND} \,..\, x_i \text{ is } A^{r}_{ki}(x_i)\ \textbf{AND} \,..\, x_{ni} \text{ is } A^{r}_{kni}(x_{ni})\ \textbf{\textit{THEN}}\ y \text{ is } B^{r}_{j}(y) \tag{2.5}$$

where $\boldsymbol{R}^r$ denotes the r^{th} fuzzy rule of the rule base with a total number of R, and r = 1, .. , R. Superscript r corresponds to the r^{th} fuzzy rule. Each input variable x_i has its own N_i fuzzy sets A_{ki} with subscript ki = 1, .. , N_i, denoting the ki^{th} fuzzy set of input x_i for i = 1, .. , ni. Output variable y has N_b fuzzy sets, B_j , with subscript j = 1, .. , N_b. Fuzzy sets A_{ki}'s and B_j use input x_i and output y as their arguments, respectively. Further explanations of terminology used will be given in the following subsections. We start with 'fuzzy relation'.

2.2.1.1 Fuzzy relation of a fuzzy rule

In Eq.(2.5), the linguistic expression A (or B) is a *fuzzy set* which maps the input x_i (or output y) into a bounded interval [0,1] via a membership function $\mu_A(x_i)$ (or $\mu_B(y)$), where the input x_i can be either a fuzzy number or a crisp number. The term, x_i is $A^r_{ki}(x_i)$, is a *fuzzy proposition* which defines a *fuzzy relation* $fr_{i,r}(x_i)$, or simply $fr_{i,r}$, that associates the input x_i with the linguistic descriptor $A^r_{ki}(x_i)$. Similarly, y is $B^r_j(y)$ also forms a fuzzy relation denoted as $fr_{j,r}$ (y) or simply $fr_{j,r}$. 'AND' in the antecedent part of Eq.(2.5) plays the role of *fuzzy conjunction* of all fuzzy relations $fr_{i,r}$ in the r^{th} rule and then forms a conjunctive fuzzy relation, denoted as frp_r, of the antecedent part of a fuzzy rule. The conjunction can be achieved by the *triangular norm* (T-norm) as defined by

$$\begin{aligned} frp_r &= T\,(fr_{1,r}, fr_{2,r}, ..., fr_{i,r}, ..., fr_{ni,r}) \\ &= fr_{1,r} \wedge fr_{2,r} \wedge .. \wedge fr_{i,r} \wedge .. \wedge fr_{ni,r} \\ &= T\,(fr_{i,r})\ ;\ \text{for } i = 1, .., ni \end{aligned} \tag{2.6}$$

An alternative operation named the *triangular conorm* (T-conorm or S-norm) may be used for linguistic disjunction 'OR',

$$\begin{aligned} frp_r &= S\,(fr_{1,r}, fr_{2,r}, ..., fr_{i,r}, ..., fr_{ni,r}\,) \\ &= fr_{1,r} \vee fr_{2,r} \vee .. \vee fr_{i,r} \vee .. \vee fr_{ni,r} \\ &= S\,(fr_{i,r})\ ;\ \text{for } i = 1, .., ni \end{aligned} \tag{2.7}$$

The commonly used operation of T-norms and S-norms are illustrated in the table below.

Table 2.1: The commonly used T-norm and S-norm operators.

	$T(p, q)$	$S(p, q)$
Zadeh	$\min(p, q)$	$\max(p, q)$
Bandler and Kohout	pq	$p + q - pq$
Łukasiewicz and Giel	$\max(p + q - 1, 0)$	$\min(p + q, 1)$

Arguments p and q in Table 2.1 can be either fuzzy numbers or crisp numbers. For example, when a set of numerical inputs is given, the corresponding numerical value of fuzzy relation can be obtained from either Eq.(2.6) or Eq.(2.7) accompanied by any of the operators listed in Table 2.1. Such a process is called fuzzification, as it maps a crisp point x' in X, the universe of discourse of input x, into a fuzzy set A(x') in X via a chosen membership function. Several choices of this mapping can be made. Among them, for a crisp input x', the most commonly used is singleton fuzzification, i.e., fuzzy set A(x') is characterized by its membership function for $\mu_A(x') = 1$ when x' = x, and $\mu_A(x') = 0$ for all other $x' \in X$ but $x' \neq x$. Hence, such a fuzzy set A(x') is called singleton and it is represented by a crisp number (either 1 or 0) rather than by a fuzzy number. Other possible choices of mapping can result in a non-singleton fuzzy set, such as triangular or Gaussian shaped membership functions.

2.2.1.2 *Fuzzy implication of a fuzzy rule*

Fuzzy implication, denoted as **I**, is performed by fuzzy relations of the antecedent part of a fuzzy rule together with the fuzzy relation of the consequent part of a fuzzy rule, (i.e., all $fr_{i,r}$'s and $fr_{j,r}$), and then generates an implicated *fuzzy relation* FR^r, where superscript r represents the r^{th} fuzzy rule. Hence the fuzzy relation FR^r of the r^{th} fuzzy rule, Eq.(2.5), can be constructed as follows

$$\begin{aligned} FR^r &= \mathbf{I}\,(frp_r(\mathrm{x})\,,\ fr_{j,r}(\mathrm{y})) \\ &= \mathbf{I}\,(\,\mathrm{T}\,(fr_{i,r}(\mathrm{x}_i))\,,\ fr_{j,r}(\mathrm{y})\,) \qquad \text{for } i = 1, .., ni \end{aligned} \tag{2.8}$$

There are two types of fuzzy implications, One is the fuzzy implication complying with *classical implication*, i.e.

$$\text{type I:} \qquad p \to q \equiv \mathbf{I}_{CI}(p, q) \equiv \neg p \vee q \tag{2.9}$$

The other is the fuzzy implication complying with *classical conjunction*, i.e.

$$\text{type II:} \qquad p \to q \equiv \mathbf{I}_{CC}(p, q) \equiv p \wedge q. \tag{2.10}$$

Based on these two implications one can generalize the following five fuzzy implications:

$$\text{(1) S - norm implication:} \qquad \mathbf{I}(p, q) = \mathrm{S}(\mathrm{c}(p), q) \tag{2.11}$$

where $c(p) = 1 - p$ means the complement of argument p; S represents a S - norm operation (refer to in Table 2.1). For example, Kleene - Dienes implication, $\mathbf{I}(p, q) = \min(1-p, q)$.

(2) Quantum Logic implication: $\mathbf{I}(p, q) = S(c(p), T(p, q))$ (2.12.a)

where T is a T-norm operation. This implication is also called *'prepositional calculus'*. When p is replaced by $1 - q$ and q is replaced by $1 - p$, then an *'extended prepositional calculus'* is obtained

$$\mathbf{I}(p, q) = S(T(c(p), c(q)), q) \tag{2.12.b}$$

(3) Residuated implication:

$$\mathbf{I}(p, q) = \begin{cases} 1 & \text{if } p \le q \\ 0 & \text{if } p = 1 \wedge q = 0 \\ \in [0, 1\rangle & \text{otherwise} \end{cases} \tag{2.13.a}$$

This implication is also referred to as *generalization of modus ponens* (GMP), which is expressed as

$$\mathbf{I}(p, q) = \sup\{\lambda \in [0,1] \,|\, T(p, \lambda) \le q\} \tag{2.13.b}$$

Similarly, if argument p is replaced by $1 - q$ and q is replaced by $1 - p$, a *generalization of modus tollens* (GMT) is expressed as

$$\mathbf{I}(p, q) = 1 - \inf\{\lambda \in [0,1] \,|\, S(q, \lambda) \le p\} \tag{2.13.c}$$

(4) T - norm implication: $\mathbf{I}(p, q) = T(p, q)$ (2.14)

Where T stands for the T-norms operation (refer to in Table 2.1). For instance, Mamdani's minimal implication $\mathbf{I}(p, q) = \min(p, q)$ and Larsen's product implication $\mathbf{I}(p, q) = pq$.

(5) classical intersection: $\mathbf{I}(p, q) = \inf\{\lambda \in [0,1] \,|\, S(1-p, \lambda) \le p\}$ (2.15)

Based on the above definitions of implications, a fuzzy relation FR^r obtained by Eq.(2.8) can be used for aggregation (stated next), or for the use of inference of a fuzzy rule, see subsection 2.2.1.4.

2.2.1.3 Aggregation of a set of fuzzy rules

A fuzzy rule base consists of a set of different fuzzy rules, where each fuzzy rule connects to the others by means of a linguistic connective term 'ALSO' to form a complete fuzzy rule base.

Still, based on fuzzy implication, each fuzzy rule (in Eq.(2.5)) forms a fuzzy relation FR^r (from Eq.(2.8)). Therefore, for a set of fuzzy rules with a total number of R, the resultant fuzzy relation FR can be aggregated by

$$FR = \begin{cases} \bigcup_{r=1}^{R} FR^r = \mathop{\mathrm{S}}_{r=1}^{R} FR^r & \textit{if rule connective 'ALSO' is interpreted as disjunction 'OR'} \\ \bigcap_{r=1}^{R} FR^r = \mathop{\mathrm{T}}_{r=1}^{R} FR^r & \textit{if rule connective 'ALSO' is interpreted as conjunction 'AND'} \end{cases} \tag{2.16}$$

where S and T represent the S-norm and T-norm operations, respectively.

2.2.1.4 *Inference of a fuzzy rule*

Based on the above fuzzy relation analysis, and given a premise proposition to an existing fuzzy rule, one can infer a consequent output fuzzy set by applying Zadeh's *compositional rule of inference* to compose the fuzzy relation of the given premise proposition and the fuzzy rule implication. The composition of fuzzy relation is done with a T-norm operation. This can be explained with the following example. If only two inputs x_1 and x_2 and one output y are considered, the Mamdani's type of fuzzy rule is expressed as

$$R^r_{(M)}: \textit{IF } x_1 \text{ is } A^r_1(x_1) \textbf{ AND } x_2 \text{ is } A^r_2(x_2) \textit{ THEN } y \text{ is } B^r(y) \tag{2.17}$$

where Eq.(2.17) forms a fuzzy relation FR^r as defined by Eq.(2.8). Given a set of fuzzy input (A_1', A_2') and the 'AND' conjunction, all fuzzy rule implications are based on the T-norm operation. The resultant output fuzzy set $B^{r'}$ (with respect to the r^{th} fuzzy rule) can be inferred by composition of A' and FR^r as below, where fuzzy set A' is a conjunction of A_1' and A_2' and is denoted as $T_c(A_1', A_2')$.

$$\begin{aligned} B^{r'} &= A' \circ_{T_I} FR^r \\ &= T_c(A_1', A_2') \circ_{T_I} FR^r \\ &= T_c(A_1', A_2') \circ_{T_I} T_I(\, T_c(A^r_1, A^r_2), B^r) \\ &= T_I(\, T_c(A_1', A_2'), T_I(\, T_c(A^r_1, A^r_2), B^r)\,)^{*} \\ &= T_I(\, \mathrm{hgt}(\, T_I(\, T_c(A_1', A_2'), T_c(A^r_1, A^r_2))\,), B^r)^{\dagger} \end{aligned} \tag{2.18}$$

---

* Since T (p , T (q,r)) = T(T(p , q), r) in the above expression, if we regard argument p as $T_c(A_1', A_2')$ and T(q , r) as $T_I(\, T_c(A^r_1, A^r_1), B^r)$, then the next expression can be derived accordingly.

† The term 'hgt' stands for the highest value of the argument.

where '$\circ_{T_I}$' denotes a *T-norm composition operation*, and T_c and T_I denote the T-norm conjunction and T-norm implication, respectively. When the *min* operator is used as the T-norm operation, the inference fuzzy output of Eq.(2.18) can be expressed in terms of membership function by

$$\begin{aligned}
\mu_{B^{r'}}(y) &= \sup_{x_1,x_2}\{[\mu_{A'_1}(x_1)\wedge(\mu_{A'_2}(x_2)]\wedge\mu_{FR^r}(x_1,x_2,y)\} \\
&= \sup_{x_1,x_2}\{[\mu_{A'_1}(x_1)\wedge\mu_{A'_2}(x_2)]\wedge[\mu_{A^r_1}(x_1)\wedge(\mu_{A^r_2}(x_2)\wedge\mu_{B^r}(y)]\} \\
&= \{\sup_{x_1,x_2}[(\mu_{A'_1}(x_1)\wedge(\mu_{A'_2}(x_2)\wedge\mu_{A^r_1}(x_1)\wedge\mu_{A^r_2}(x_2)]\}\wedge\mu_{B^r}(y) \qquad (2.19) \\
&= \{\sup_{x_1}[\mu_{A'_1}(x_1)\wedge\mu_{A^r_1}(x_1)]\wedge\sup_{x_2}[\mu_{A'_2}(x_2)\wedge\mu_{A^r_2}(x_2)]\}\wedge\mu_{B^r}(y) \\
&= hgt(A'_1\cap A^r_1)\wedge hgt(A'_2\cap A^r_2)\wedge\mu_{B^r}(y)
\end{aligned}$$

This resultant inference output fuzzy set of the r^{th} fuzzy rule can be used to derive the final output of the fuzzy system based on all fuzzy rules; see next subsection and defuzzification procedure.

2.2.1.5 Inference of a set of fuzzy rules

There are two approaches to derive the resultant inference of a set of fuzzy rules. One is the *local inference approach* that first performs inference with individual rules and then aggregates the results afterwards. The other is referred to as the *global inference approach* where a fuzzy relation *FR* is first obtained by aggregating all the fuzzy relations FR^r, then the result is inferred from this resultant fuzzy relation *FR*. The difference in these two approaches lies in the implication method on which the fuzzy rule is based. For example, if the fuzzy rule applies the classical-conjunction-based implication I_{CC} (see Eq.(2.10)), the disjunction '*OR*' (see Eq.(2.16)) is then used as the rule connective to aggregate all fuzzy rules. This will result in no difference between the global and the local inference approaches. Taking Eq.(2.17) as an example, the aggregated output fuzzy set B' can be expressed as follows

$B' = A' \circ FR$ (this is a global approach, since *FR* is used, rather than FR^r) (2.20.a)

$= A' \circ \{\cup_r FR^r\}$ (*FR* is a disjunction of FR^r based on I_{CC})

$= \cup_r \{A' \circ FR^r\}$ (this is a local approach, since FR^r is used) (2.20.b)

$= \cup_r B^{r'}$

where FR^r is implicated according to the classical-conjunction :

$$FR^r = I_{CC}(T_c(A^r_1, A^r_2), B^r) = T_c(A^r_1, A^r_2) \wedge B^r \qquad (2.20.c)$$

Hence, the above example using the T-norm as implication makes no difference between the result of global and local inference.

In contrast, if the fuzzy rule applies the classical-implication-based implication $\mathbf{I}_{CI}$ (see Eq.(2.9)), then the conjunction '*AND*' (see Eq.(2.16)) is used as the rule connective to aggregate all fuzzy rules. The result of the global inference will differ from that of the local inference. In this case, the output fuzzy set B' becomes

$$B' = A' \circ FR \quad \text{(global approach)} \tag{2.21.a}$$

$$= A' \circ \{\cap_r FR^r\} \quad (FR \text{ is a conjunction of } FR^r \text{ based on } \mathbf{I}_{CI})$$

$$\subseteq \cap_r \{A' \circ FR^r\} \quad \text{(local approach)} \tag{2.21.b}$$

where FR^r is implicated according to the classical-implication :

$$FR^r = \mathbf{I}_{CI}(\ T_c(A^r_1\ ,\ A^r_2),\ B^r\ ) = (\ 1 - T_c(A^r_1\ ,\ A^r_2)\ ) \vee B^r \tag{2.21.c}$$

Eq.(2.21.b) indicates that the results of local inference are less restrictive (informative) than those obtained from the global inference.

However, in cases where numerical input is used and the fuzzy input A' is replaced by a singleton value τ', there is no difference between the two approaches. The following derivation explains this situation.

$$B' = \tau' \circ FR \quad \text{(global approach)} \tag{2.22.a}$$

$$= \tau' \circ \{\cap_r FR^r\}$$

$$= \cap_r \{\tau' \circ FR^r\} \quad \text{(local approach)} \tag{2.22.b}$$

$$= \cap_r \{\tau' \circ \mathbf{I}\ (T_c(A^r_1\ ,\ A^r_2),\ B^r)\}$$

$$= \cap_r \mathbf{I}\ (\ \text{hgt}\ (\tau' \cap T_c(A^r_1\ ,\ A^r_2))\ ,\ B^r)\} \tag{2.22.c}$$

2.2.1.6 Commonly used inference methods

After introducing the fuzzy inference based on fuzzy relations of the Mamdani type of fuzzy rule, a summary is given in this subsection of some commonly used inference methods. The fuzzy inference based on fuzzy relations includes two main factors: construction of the fuzzy relation *FR* (model) based on implication of all fuzzy rules and, the use of the *FR* to actually infer the output from the inputs by composition. The construction of *FR* from all fuzzy rules is mainly achieved by linguistic *conjunction* (or *disjunction*) and *implication*, whilst the conjunction, in fact, refers to *aggregation* based on the choice of rule connective. Moreover, composition consists of two phases: a *combination* and a *projection* phase. These concepts are explained by the following example, where we assume a new set of inputs **x'** (=[x'$_1$ .. x'$_i$ .. x'$_{ni}$]^T) is given and is described by a fuzzy set A'(**x'**) (denoted as A' for short in the following).

(1) *Max-min* method

$$\mu_{B'}(y) = \overbrace{\max_{r}}^{aggregation} \underbrace{\min}_{implication} (\nu_r(\mathbf{x}), \mu_{B^r}(y)) \qquad r = 1, .., R \qquad (2.23.a)$$

where

$$\nu_r(\mathbf{x}) = \underbrace{\min_{i}}_{conjunction} \underbrace{[\overbrace{\sup}^{projection} \overbrace{\min}^{combination}}_{composition} (\mu_{A'}(x_i'), \mu_{A^r}(x_i))] \qquad i = 1, .., ni \qquad (2.23.b)$$

$\nu_r(\mathbf{x})$ (or, ν_r for short), as defined in Eq.(2.23.b), is a firing strength or degree of fulfillment (DOF) with respect to the r^{th} fuzzy rule. The projection on Eq.(2.23.b) means the result of combination, $\min(\mu_{A'}(x_i'), \mu_A^r(x_i))$, is projected onto bounded interval [0,1], and then it conjugates with results of other inputs (x_i). Finally, a firing strength, $\nu_r(\mathbf{x})$, with respect to the r^{th} fuzzy rule of all inputs x_i is thus obtained. This method is used by Mamdani and his co-workers, where the term *Max-min* comes from the fact that the implication uses the *min* operation and aggregation uses the *max* operation. It is also called *sup-min* method.

(2) *Max-prod* method

$$\mu_{B'}(y) = \overbrace{\max_{r}}^{aggregation} (\nu_r(\mathbf{x}) \underbrace{\cdot}_{implication} \mu_{B^r}(y)) \qquad r = 1, .., R \qquad (2.24.a)$$

where

$$\nu_r(\mathbf{x}) = \begin{cases} \overbrace{\min_{i}}^{conjunction} [\overbrace{\sup \min}^{composition} (\mu_{A'}(x_i'), \mu_{A^r}(x_i))] \\ or \\ \underbrace{\prod_{i}}_{conjunction} [\underbrace{\sup \min}_{composition} (\mu_{A'}(x_i'), \mu_{A^r}(x_i))] \end{cases} \qquad i = 1, .., ni \qquad (2.24.b)$$

ν_r, as defined in Eq.(2.24.b), is similar to the definition of ν_r defined in Eq.(2.23.b), but the conjunction operation could be chosen either as a *min* operation or as a *product* operation. The term *Max-prod* comes from the fact that the implication uses the *product* operation in Eq.(2.24.a) and aggregation uses the *max* operation. It is also called *max-dot* or *sup-prod* method.

(3) *Sum-prod* method

$$\mu_{B'}(y) = \overbrace{\sum_{r}}^{aggregation} (v_r(x) \underbrace{\cdot}_{implication} \mu_{B^r}(y)) \qquad r = 1, .., R \qquad (2.25.a)$$

where v_r is defined in the same manner as Eq.(2.24.b) and the conjunction operation could be either *min* or *product* operation. The term *Sum-prod* comes from the fact that the implication uses the *product* operation and aggregation uses the *summation* operation based on all R fuzzy rules. It is noted that the summation of all product terms ($v_r \cdot \mu_B^r$) will likely result in a supernormal fuzzy set* on the output universe, that does not conform to the fuzzy set theory. Therefore, a bounded summation may be used to modify Eq.(2.25.a) in order to alleviate the supernormal situation,

$$\mu_{B'}(y) = \min(\sum_{r} (v_r \cdot \mu_{B^r}(y)), 1) \qquad r = 1, .., R \qquad (2.25.b)$$

Some remarks of the above inference methods are made below.

(R.1) If the input x' is fuzzified by a singleton membership function $\mu_{A'}(x')$, the result of fuzzification becomes crisp, then the result of the combination of $\mu_{A'}(x')$ and $\mu_A^r(x)$ (see Eq.(2.23.b) and Eq.(2.24.b)) is crisp rather than fuzzy. By virtue of the sup-min operation, the result of composition remains as crisp as it is projected onto interval [0,1]. Therefore, for any crisp input x' (for instance, any crisp measurement signals), the result of composition can be simplified by just evaluating the membership value of $\mu_A^r(x')$ directly (see Eq.(2.27.a), shown next). On the other hand, if x' is characterized by a non-singleton fuzzy set A', the resultant conjunction will still be crisp because the sup-min operation projects the fuzzy combination of $\mu_{A'}(x')$ and $\mu_A^r(x)$ onto [0,1]. Hence, a crisp value is obtained as a result of composition. To put it briefly, irrespective of the use of crisp or fuzzy inputs, thanks to the sup-min composition, the resultant DOF, v_r, is crisp.

(R.2) One might wonder which of the various implication methods is a good one to be implemented. The author in [78] defines a number of intuitive criteria and shows that the *min*-implication and the *product*-implication (both are T-norm implications) fulfill many of these criteria. From the computational point of view, among these two T-norm implications, it can be seen that the *product*-implication is much easier to manipulate than the *min*-implication. Furthermore, in [46] simulation results show that the sum-product method

* A supernormal fuzzy set means that its maximum membership value is greater than 1; a normal fuzzy set, its maximum membership value reaches 1; a subnormal fuzzy set, less than 1.

together with centroid of gravity (COG) defuzzification (shown later), is more intuitive, simple in nature and performs better, in contrast to the max-min method together with COG defuzzification.

(R.3) Two types of aggregation are mainly used to infer the result, *viz. max*-aggregation and *sum*-aggregation. It should be noted that this aggregation procedure implicitly relates to the defuzzification procedure, as will be shown in the next subsection. Furthermore, the *max*-aggregation causes a nonlinear result of $\mu_B'(y)$ - which is not preferable - in contrast to a linear $\mu_B'(y)$ which results from the *sum*-aggregation. As indicated in Eq.(2.25.b), a bounded summation can be used to avoid the occurrence of supernormal fuzzy sets. Nevertheless, it is interesting to note that, if the DOF had been normalized in advance, then an ordinary *sum*-aggregation can be used without the supernormality problem.

2.2.1.7 Defuzzification

Defuzzification defuzzifies the inference output when a quantitative result of the fuzzy reasoning is required. It should be noted that the defuzzification method actually integrates aggregation and defuzzification into one operation implicitly. Several possible defuzzification methods can be employed, such as centroid of gravity, mean of maximum, indexed defuzzification and center of area. Details can be found in [30] and [41] . We will only introduce the centroid of gravity (COG) defuzzification as it is the most commonly used defuzzification method up to date. Once the individual inference output of each fuzzy rule is obtained, the defuzzified output can be obtained with the COG method after aggregation. The defuzzified output $y_{(M)}$, denoted as cog(B'), is obtained from the membership function $\mu_{B'}(y)$ as,

$$y_{(M)} = \mathrm{cog}(B') = \frac{\int_Y \mu_{B'}(y) y\,dy}{\int_Y \mu_{B'}(y)\,dy} \tag{2.26.a}$$

and the discrete version is

$$y_{(M)} = \mathrm{cog}(B') = \frac{\sum_{d=1}^{N_d} \mu_{B'}(y_d) y_d}{\sum_{d=1}^{N_d} \mu_{B'}(y_d)} \tag{2.26.b}$$

where y_d is the equi-distant quantization, in a total number N_d, used to discretize the membership function $\mu_{B'}(y)$ of the fuzzy output B' on the universe of discourse Y.

2.2.1.8 *Practical fuzzy inference procedure*

This subsection summarizes the fuzzy inference procedure from a practical point of view. The term 'practical' is used because the input and output are quantitative values rather than fuzzy values in most control applications. Hence the results obtained from the local inference approach are similar to those of the global inference approach. Assuming a new input **x'** is given with elements x_i', and fuzzy rules according to Eq.(2.5), the inference procedure, which is based local inference, can be summarized in the following five steps :

Step 1 : find the matching degree $\xi_r(x_i')$ (or denoted as $\xi_{i,r}$) of each input with respect to each fuzzy rule

$$\xi_{i,r}=\begin{cases}\mathrm{hgt}(A_i',A_i^r) & \text{in general}\\ \mu_{A_i^r}(x_i') & \text{numerical input}\end{cases} \tag{2.27.a}$$

It can be seen that this step only performs a composition of $A'(x_i')$ and $A^r(x_i)$ (denoted as A'_i and A^r_i, respectively, in above Eq.(2.27.a)) or, in fact, finds the membership value of input x_i'.

Step 2 : perform conjunction of each input in the antecedent part of a fuzzy rule to get a firing strength $v_r(\mathbf{x}')$, denoted by v_r for short,

$$\begin{aligned} v_r &= \mu_{A_1^r}(x_1')\wedge\mu_{A_2^r}(x_2')\wedge\cdots\wedge\mu_{A_i^r}(x_i')\wedge\cdots\wedge\mu_{A_{ni}^r}(x_{ni}')\\ &= \mathop{T}_{i=1}^{ni}\mu_{A_i^r}(x_i') \\ &= \mathop{T}_{i=1}^{ni}\xi_{i,r}=\begin{cases}\prod_{i=1}^{ni}\xi_{i,r} & \text{T}-\text{norm is the } \textit{product} \text{ operation}\\ \min_i(\xi_{i,r}) & \text{T}-\text{norm is the } \textit{min} \text{ operation}\end{cases}\end{aligned} \tag{2.27.b}$$

Step 3 : find the implication of each rule

$$\mu_{B'^r}(y) = I\,(v_r, \mu_{B^r}(y)) \qquad r = 1, .., R \tag{2.27.c}$$

The implication operation **I** is defined by a choice from Eq.(2.11) to Eq.(2.15).

Step 4 : aggregate all fuzzy rules

$$\mu_{B'}(y)=\begin{cases}\bigcup_r \mu_{B'^r}(y)=\bigcup_r I_{CC}(v_r,\mu_{B^r}(y)) & \text{Implication based on classical conjunction}\\ \bigcap_r \mu_{B'^r}(y)=\bigcap_r I_{CI}(v_r,\mu_{B^r}(y)) & \text{Implication based on classical implication}\end{cases} \tag{2.27.d}$$

Step 5 : defuzzification, when COG is employed

$$y_{(M)} = \text{cog}(B') = \frac{\int_Y \mu_{B'}(y) y dy}{\int_Y \mu_{B'}(y) dy} \tag{2.27.e}$$

It can be seen that the processes in step 1 to 3 apply to each individual fuzzy rule. The defuzzification in step 5 implicitly includes the aggregation of step 4 and then defuzzifies the aggregated results from all fuzzy rules.

2.2.2 Takagi-Sugeno type of fuzzy rule

The previous subsections have considered the Mamdani type of fuzzy rule and analyzed the fuzzy reasoning based on the fuzzy relation. In this subsection, we will discuss the type of fuzzy rule which was first introduced by Takagi and Sugeno [56] [62] and further developed by Sugeno and his co-workers [61] . We will call it the Takagi-Sugeno fuzzy rule, or T-S fuzzy rule for short, which, in fact, is a variant of the ordinary Mamdani fuzzy rule. The consequent is expressed differently, where the output fuzzy set is replaced by a function, denoted as g_n below. Hence, the 'generalized T-S fuzzy rule' can be written as

$$\begin{aligned} \boldsymbol{R}_{(TS)}: \textbf{\textit{IF}} \quad & x_1 \text{ is } A_1 \text{ AND} .. x_i \text{ is } A_i \text{ AND} .. x_{ni} \text{ is } A_{ni} \\ \textbf{\textit{THEN}} \quad & y_1 = g_1(x), .., y_n = g_n(x), .., y_{nb} = g_{nb}(x) \end{aligned} \tag{2.28}$$

In what follows, we will introduce two special cases of such a fuzzy rule, the so-called first-order and zeroth-order T-S fuzzy rule [32] . The first-order T-S fuzzy rule of a MISO fuzzy system uses a linear function as its consequence and can be expressed by

$$\begin{aligned} \boldsymbol{R}^r_{(TS)}: \textbf{\textit{IF}} \quad & x_1 \text{ is } A^r_{k1}(x_1) \text{ AND} .. x_i \text{ is } A^r_{ki}(x_i) \text{ AND} .. x_{ni} \text{ is } A^r_{kni}(x_{ni}) \\ \textbf{\textit{THEN}} \quad & y = a_0^r + \sum_{i=1}^{ni} a_i^r x_i \end{aligned} \tag{2.29}$$

where constants a^r_0, a^r_1, ..and a^r_{ni} in the consequent linear function with respect to the r^{th} T-S fuzzy rule are unknown parameters to be identified. There are several successful control and modeling applications using this type of fuzzy rule, e.g., [57] [58] [59] [60] . At first glance, one can easily distinguish the difference between a T-S fuzzy rule and a Mamdani fuzzy rule. Both may share the same structure in the antecedent part, but differ in the consequent part. This common feature allows us to obtain the firing strength v_r of a T-S fuzzy rule in the same way as for a Mamdani fuzzy rule. Besides, the resulting firing strength v_r, Eq.(2.27.b), performs a T-norm conjunction of all input fuzzy sets A's which use all inputs x_i's as arguments to their input membership functions, $\mu_{A'}(x)$. Hence, the information content of input variables x_i's is already embedded in the firing strength v_r. Consequently, input variables need not necessarily be involved (or, appear) again in the consequent linear function. Based on this consideration, another variant and more simple type of T-S fuzzy rule, the so-called zeroth-order T-S fuzzy rule, which has only one constant denoted by w_r as its singleton membership function in the rule consequent, can be represented as below,

$$R^{r}_{(TS)}: \textit{IF } x_1 \text{ is } A^r_{k1}(x_1) \text{ AND .. } x_i \text{ is } A^r_{ki}(x_i) \text{ AND .. } x_{ni} \text{ is } A^r_{kni}(x_{ni}) \textit{ THEN } y = w_r \tag{2.30}$$

Because the consequent is crisp, whether a linear function or a constant, we can easily employ the weighted sum as the defuzzification method to the T-S fuzzy rules in order to get the resultant output of the T-S fuzzy model. In the case of the linear consequent function as in Eq.(2.29), this will result in

$$\begin{aligned} y_{(TS)}(\mathbf{x}) &= \frac{\sum_{r=1}^{R} v_r(\mathbf{x}) \cdot (a_0^r + \sum_{i=1}^{ni} a_i^r x_i)}{\sum_{p=1}^{R} v_p(\mathbf{x})} \\ &= \sum_{r=1}^{R} \bar{v}_r(\mathbf{x}) \cdot (a_0^r + \sum_{i=1}^{ni} a_i^r x_i) \end{aligned} \tag{2.31}$$

where the term, $\bar{v}_r(\mathbf{x})$, represents the normalized firing strength, as defined by

$$\bar{v}_r(\mathbf{x}) = \frac{v_r(\mathbf{x})}{\sum_{p=1}^{R} v_p(\mathbf{x})} \tag{2.32}$$

In the case of a constant consequent as in Eq.(2.30), this gives

$$y_{(TS)}(\mathbf{x}) = \frac{\sum_{r=1}^{R} v_r(\mathbf{x}) \cdot w_r}{\sum_{p=1}^{R} v_p(\mathbf{x})} = \sum_{r=1}^{R} \bar{v}_r(\mathbf{x}) \cdot w_r \tag{2.33}$$

Eq.(2.33) has the attractive property that the output is linear-in-the-parameters.

Moreover, it is also interesting to note that when the consequent functions, g(**x**)'s in Eq.(2.28), are continuous, the output of the generalized T-S fuzzy model obtained by weighted sum defuzzification can optimally represent a global model [33] . See Appendix A.

In the development of an integrated neural-fuzzy model following next, we will prefer the zeroth-order T-S fuzzy rule, because of its simplicity. Also, several advantages can be gained using the zeroth-order T-S fuzzy model. One is that the problem of over-parameterization is less likely to occur than the first-order T-S fuzzy model when the number of system inputs and the total number of fuzzy rules are large. Second, the output of a zeroth-order T-S fuzzy model is linear-in-the-parameters, allowing a very fast estimation of the unknown consequent parameters, w_r's in Eq.(2.33). Third, the zeroth-order T-S fuzzy model is functionally equivalent to a radial basis function network. This facilitates the construction of the neural-fuzzy model and gives it the property of a universal function approximator.

2.3 CONSTRUCTION OF THE NUFZY SYSTEM*

A fuzzy system consists of four basic elements: (1) *fuzzifier*, (2) *fuzzy rule base*, (3) *fuzzy inference mechanism*, and (4) *defuzzifier*. These elements can be represented in various forms; for example, see [41] . Options for executing these basic elements for fuzzy reasoning have been described in the previous sections. In this section, we will establish an integrated neural-fuzzy system, called the NUFZY system, to carry out fuzzy reasoning and to achieve the goal of function approximation. In order to obtain functional equivalence between the fuzzy system and a neural network structure, the functions of each corresponding element of the fuzzy system are cast into network terms and are thus represented by neurons as well as weighted connections. Without loss of generality, we confine ourselves to the radial basis function as the membership function, the algebraic product as T-norm operation for the AND conjunction in the rule antecedent, and the centroid of gravity as the method of defuzzification. To construct a multi-input-multi-output NUFZY system with ni inputs x_i, i = 1,.., ni, and nb outputs y_n, n = 1,.., nb, the following assumptions are made:

(A.1) Each input x_i has Ni membership functions, each associated with its own linguistic label A_{ki} with index ki = 1, .. , N_i, and i = 1,.., ni. The number of membership functions, and the shape and location of the membership functions for each input x_i can be determined *a priori* by the users.

(A.2) The fuzzy rules take the form of a zeroth-order Takagi-Sugeno fuzzy rule by taking the consequent as a singleton value, denoted as w_{rn} (a constant term), rather than a linear function of system inputs. Hence, the fuzzy rule is expressed as

$$\begin{aligned} R^r_{(TS)}: \quad & \textbf{\textit{IF}}\ x_1 \text{ is } A^r_{k1}(x_1) \text{ AND} \,..\, x_i \text{ is } A^r_{ki}(x_i) \text{ AND} \,..\, x_{ni} \text{ is } A^r_{kni}(x_{ni}) \\ & \textbf{\textit{THEN}}\ y_1 = w_{r1}, \,..\,, y_n = w_{rn}, \ , y_{nb} = w_{r\,nb} \end{aligned} \tag{2.34}$$

where $A^r_{ki}(x_i)$ represents the kith linguistic label of x_i with respect to the rth fuzzy rule R^r, and w_{rn} the consequent weight of output y_n with respect to the rth fuzzy rule.

* This section of construction of the NUFZY system is extracted from the paper of [72] , titled 'A neuro-fuzzy approach to identify lettuce growth and greenhouse climate'. To appear in *Artificial Intelligence Review - special issue of AI applications in Biology and Agriculture*, 1997.

A schematic architecture of the hybrid NUFZY system given in Figure 2.2. It resembles a triple-layered feedforward neural network. Layer 1 and layer 2 of the NUFZY system conduct the antecedent part of the fuzzy system and layer 3 the consequent part.

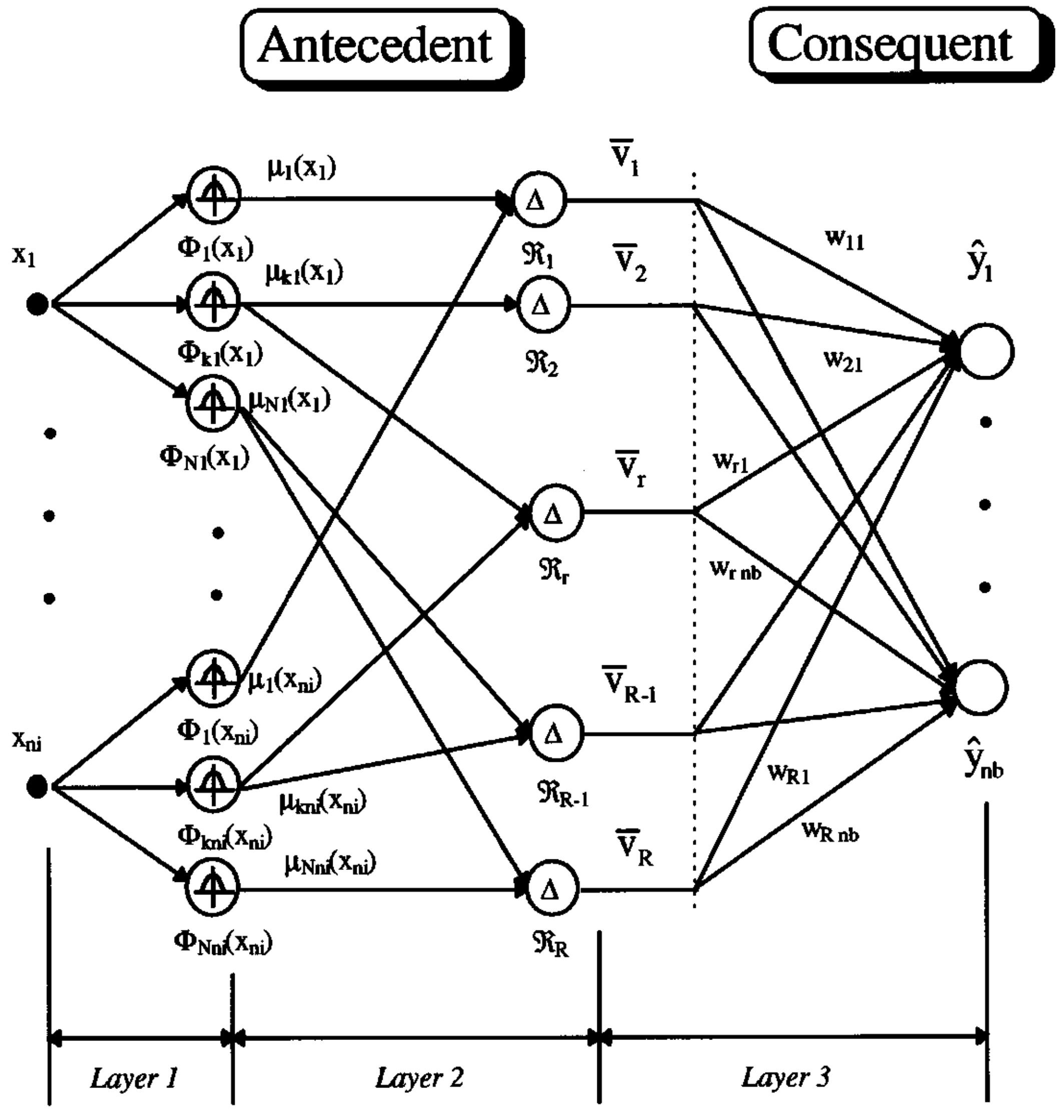

Figure 2.2: The structure of the neural-fuzzy, NUFZY, system.

2.3.1 Architecture of the antecedent part of NUFZY

Layer 1

This layer consists of the *input node* x_i and *membership node* $\Phi_{ki}(x_i)$ and fixed connection weights between input nodes and membership nodes. The input node only distributes the input into the membership nodes with fixed weights of unity, whereas the membership node, defined by a radial basis function, is used to obtain the fuzzified values (degree of membership), $\mu_{ki}(x_i)$. Two types of radial basis functions are considered,

(1) Gaussian function:

$$\mu_{ki}(x_i) = \exp\left(-\frac{1}{2}\frac{(x_i - c_{i,ki})^2}{\sigma_{i,ki}^2}\right) \tag{2.35}$$

(2) inverse multiquadratic function (IMQ):

$$\mu_{ki}(x_i) = \frac{1}{\sqrt{(x_i - c_{i,ki})^2 + \sigma_{i,ki}^2}} \tag{2.36}$$

where $c_{i,ki}$ and $\sigma_{i,ki}$ are the ki^{th} center and bandwidth of $\Phi_{ki}(x_i)$, respectively. The total number of membership nodes $\Phi_{ki}(x_i)$ in layer one, M, is given by

$$M = \sum_{i=1}^{ni} N_i \tag{2.37}$$

Once all N_i's are decided for each x_i, the center and width of each membership function can be determined from the available data set. For instance, given np pairs of input data $x(t)=[x_1(t)\ ..\ x_i(t)\ ..\ x_{ni}(t)]^T$, $t = 1,.., np$, the centers of the membership functions can be taken as equally spaced in the range of each input x_i, i.e.

$$c_{i,ki} = \min(x_i) + [\max(x_i) - \min(x_i)] / [N_i - 1] \qquad \text{for } ki = 1, .., Ni \tag{2.38}$$

The width of the membership functions, $\sigma_{i,ki}$, can be chosen as reasonable values in a range guided by the values of the variance or standard deviation of the available data set in order to ensure the suitable overlap of the membership functions, thus guaranteeing the continuity of the predicted output of the fuzzy system. The output of each neuron in this layer corresponds to a fuzzified value $\mu_{ki}(x_i)$.

Layer 2

This layer consists of the *rule node* $\Re_r$, which represents a fuzzy rule for r = 1, 2,.., R, where R is the total number of all fuzzy rules. When we use a full combination of all inputs in each rule node, the total number of all fuzzy rules is given by

$$R = \prod_{i=1}^{ni} N_i \tag{2.39}$$

Inputs to the rule node are the fuzzified values of $\mu_{ki}(x_i)$ from layer one. Each rule node performs a two-step operation as will be described later. According to Eq.(2.34), each rule node involves only one membership function for each input. Therefore, the existence of a connection between a rule node $\Re_r$ and a membership node $\Phi_{ki}(x_i)$ is represented by a value of either 1 or 0, forming a relationship matrix ***RM*** with dimension R×M. Each row r of ***RM***, represents the status of the antecedent part of a fuzzy rule, i.e., a value of 1 represents a link between the r^{th} rule node and the corresponding membership node, whilst an element with value 0 indicates no connection. Hence, the relationship matrix indeed constructs a prototype fuzzy rule base with all possible combinations of input variables provided each Ni is assigned. The following example shows how the ***RM*** matrix is used to represent the connections between nodes on layers one and two. Suppose we had three inputs, x_1, x_2, x_3, having 2, 3 and 2 membership functions, respectively. The membership values, for input x_1 are $\mu_1(x_1)$ and $\mu_2(x_1)$; for input x_2 are $\mu_1(x_2)$, $\mu_2(x_2)$, and $\mu_3(x_2)$; for input x_3 are $\mu_1(x_3)$ and $\mu_2(x_3)$. They are denoted by μ_{11}, μ_{12}, μ_{21}, μ_{22}, μ_{23}, μ_{31} and μ_{32}, respectively, for short in below. In this case, R = 2 × 3 × 2 = 12 and M = 2 + 3 + 2 = 7. The relationship matrix is denoted as $\boldsymbol{RM}_{(2,3,2)}$ with values as follows:

$$
\begin{array}{l}
 \\
r=1 \\ r=2 \\ r=3 \\ r=4 \\ r=5 \\ r=6 \\ r=7 \\ r=8 \\ r=9 \\ r=10 \\ r=11 \\ r=12=R
\end{array}
\begin{array}{c}
\begin{array}{ccccccc} \mu_{11} & \mu_{12} & \mu_{21} & \mu_{22} & \mu_{23} & \mu_{31} & \mu_{32} \end{array} \\
\left[\begin{array}{cc|ccc|cc}
1 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 \\
\boxed{1} & \boxed{0} & \boxed{0} & \boxed{0} & \boxed{1} & \boxed{1} & \boxed{0} \\
0 & 1 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 0 & 1
\end{array}\right]
\end{array}
$$

From the above example, it can be seen that the third fuzzy rule (r = 3) is composed by μ_{11}, μ_{23}, and μ_{31}, respectively from the first, third, and first membership node of input x_1, x_2, and x_3. Obviously, when the number of input variables increases, this prototype fuzzy rule base will become considerably large. Therefore, methods to reduce the dimension of the model structure (number of fuzzy rules in this case) must be taken into account. In the next chapter we will consider how to remove redundant fuzzy rules.

Now let's come back to the two-step operation of the rule node to generate the node output in this layer. First, the algebraic product T-norm operation is used to realize the linguistic 'AND' conjunction of the antecedent part of Eq.(2.34). Consequently, the transient firing strength of each rule $v_r(\mathbf{x})$, or v_r for short, is obtained as a function of input x_i via $\mu_{ki}(x_i)$ together with the ***RM*** matrix by

$$v_r = \Pi_{i=1}^{ni} \ \boldsymbol{RM}(r, a_i{:}b_i) \cdot \mu_i(\mathbf{x}) \tag{2.40}$$

where $\boldsymbol{RM}(r, a_i{:}b_i)$ represents a subset of the r^{th} row vector of the ***RM*** matrix with partial elements from a_i to b_i, with $b_i = \Sigma N_p$, p = 1,.., i, and $a_i = b_i - N_i + 1$. Vector $\mu_i(\mathbf{x})$ is given by $\mu_i(\mathbf{x}) = [\mu_1(x_i) \ \mu_2(x_i) \ .. \ \mu_{ki}(x_i) \ .. \ \mu_{Ni}(x_i)]^T$. Eq.(2.40) specifies that v_r is obtained by just multiplying the membership functions involved in the r^{th} fuzzy rule, according to the connections shown in Figure 2.2. Second, to normalize v_r the normalized firing strength $\overline{v}_r$ is calculated by

$$\overline{v}_r = \frac{v_r}{\sum_{p=1}^{R} v_p} \tag{2.41}$$

This normalized firing strength $\overline{v}_r$ represents the output of the rule node in this layer.

2.3.2 Architecture of the consequent part of NUFZY

Layer 3

The *output node*, denoted as $\hat{y}_n$, stands for the n^{th} output of the NUFZY system output. The link in this layer represents a weight parameter denoted as w_{rn}, for r = 1, .. , R, and n = 1, .. , nb, that connects node $\hat{y}_n$ and $\Re_r$. These weighs, w_{rn}'s, actually represent the constant parameters in the consequent part of the r^{th} fuzzy rule given in Eq.(2.34). With the centroid of gravity defuzzification method, this node then performs a weighted summation such that the n^{th} model output is given by

$$\hat{y}_n = \frac{\sum_{r=1}^{R} v_r \times w_{rn}}{\sum_{p=1}^{R} v_p} = \sum_{r=1}^{R} w_{rn}\overline{v}_r = \mathbf{w}_n^T\overline{\mathbf{v}} \qquad \text{for } n = 1, .., nb \tag{2.42}$$

where $\mathbf{w}_n$ is the consequent weight parameter vector given by $\mathbf{w}_n = [w_{1n} \,..\, w_{rn} \,..\, w_{Rn}]^T$ and $\bar{\mathbf{v}}$ is a normalized firing strength vector given by $\bar{\mathbf{v}} = [\,\bar{v}_1 \,..\, \bar{v}_r \,..\, \bar{v}_R\,]^T$ with element $\bar{v}_r$ defined by Eq.(2.41). It is interesting to note that the model output is linear in the weight parameters; this means that the unknown weight parameters can be identified by some standard least squares parameter estimation method, for example, the orthogonal least squares (OLS) method. Note further that this linear property in the NUFZY system will be still retained for other choices of T-norm operators for the AND connection in the fuzzy rules as well as for other choices of the membership functions. Moreover, the normalized firing strength $\bar{v}_r$ can be viewed as a fuzzy basis function [80] , so that the NUFZY output forms a fuzzy basis function expansion. Hence, the NUFZY system has the property of a universal function approximator.

2.4 CONCLUDING REMARKS

Several aspects of fuzzy systems have been addressed in this chapter. A fuzzy system can be cast in a network structure to perform input-output mapping, just like artificial neural networks. We distinguish the Mamdani type fuzzy rule and the Takagi-Sugeno type fuzzy rule. The former is intuitively comprehensible due to the use of linguistic terms in both antecedent and consequent of the rule, but its fuzzy reasoning process is more complicated because it is based on fuzzy relations as well as the composition of fuzzy relations. In contrast, the latter, using linear functions of system inputs as consequence, paves the way for easier fuzzy reasoning. However, although the linear function in the consequence can explain the local linear relationship of system input and output, it is less interpretable compared to the Mamdani fuzzy rule. Based on the zeroth-order T-S fuzzy rules, we can establish an integrated neural-fuzzy system, the NUFZY system, which has a transparent network structure and gives a self-explanatory representation of the fuzzy rules. Since there is only one weight parameter in the consequent of each fuzzy rule and due to the use of weighted sum defuzzification, outputs of the NUFZY system are linear-in-the-parameter. Hence, very fast estimation of these consequent weight parameters can be accomplished by least squares estimation. This implies a fast learning method for training the integrated neural-fuzzy network. In the next chapter, we will illustrate how to apply the NUFZY system for modeling of nonlinear systems.

3. BATCH LEARNING OF THE NUFZY SYSTEM

I daily examine myself on three points:-whether, in transacting business for others, I may have been not faithful;-whether, in intercourse with friends, I may have been not sincere;-whether I may have not mastered and practiced the instructions of my teacher.

- Tsang, The Lun Yu, Analects of Confucius

曾子曰：吾日三省吾身—為人謀而不忠乎？與朋友交而不信乎？傳不習乎？

論語學而篇

3.1 INTRODUCTION†

It has been shown that the established neural-network-like fuzzy inference system, NUFZY, can implement fuzzy reasoning through a special type of network with partial connections in the antecedent part of its structure. The fact that the connection is only partial does in no way impair the function approximation ability of the system. Moreover, the structural network property of the NUFZY system allows us to train it in a similar way as neural networks. On the other hand, due to the fuzzy inference, the network structure of the NUFZY system can be interpreted in a linguistic way and becomes more transparent, in contrast to ordinary neural networks. In this chapter we will consider the training of the NUFZY system. We assume the antecedent structure of the NUFZY system could be determined *a priori* by the users. In that case the unknown weight parameters in the consequent of the NUFZY system form a linear-in-the-parameter problem. As such, given a batch of training data, the consequent weights can be identified with a very fast least squares method. This identification problem is usually called batch learning or off-line learning since the parameter identification is carried out off-line.

The performance of NUFZY for function approximation can be improved by increasing the number of fuzzy rules. In terms of linear regression the number of fuzzy rules is equivalent to the number of regressors. Likewise, there exists a problem of redundant fuzzy rules which should be taken care of in the NUFZY modeling. The orthogonal least squares method can detect redundant fuzzy rules and remove them from the NUFZY model. It also identifies the remaining consequent weight parameters of the reduced fuzzy rule based NUFZY model.

We will first outline the principle of orthogonal least squares method in section 3.2, on which the batch learning of the NUFZY model is based. Section 3.3 explains the removal of redundant fuzzy rules, hence establishing the NUFZY model with a reduced fuzzy rule base. Two nonlinear static examples are given in section 3.4 to illustrate the identification performance of the NUFZY model with the orthogonal least squares method. Special attention is given to applications to agricultural problems in section 3.5, in which we will deal with two nonlinear dynamic problems, identification of lettuce growth and greenhouse temperature in the greenhouse production system. Finally, a discussion is given in section 3.6 and section 3.7 concludes this chapter.

† This chapter is adopted from two published papers [69] , titled 'Neural-Fuzzy systems for non-linear system identification - orthogonal least squares training algorithms and fuzzy rule reduction' in *Preprints of the 2nd IFAC/IFIP/EurAgEng Workshop on AI in Agriculture* ,Wageningen, The Netherlands, May 29-31, 1995, pp 249-254, and [72] , titled 'A neuro-fuzzy approach to identify lettuce growth and greenhouse climate' accepted for publication in *Artificial Intelligence Review - special issue of AI applications in Biology and Agriculture*, to appear in 1997.

3.2 ORTHOGONAL LEAST SQUARES LEARNING

The least squares identification method is an effective optimization tool that yields a unique solution for the values of the parameters in a linear regression model. Because it is necessary to offer a set of training data to estimate the parameters, the procedure is sometimes referred to as batch learning or off-line training.

Using the same notations used in the previous chapter, given a set of training data with np events, the ni×nb multi-input-multi-output NUFZY model can be expressed as a linear regression model in a matrix form as follows

$$\mathbf{Y} = \overline{\mathbf{V}}\,\mathbf{W} + \mathbf{E} \tag{3.1}$$

where $\mathbf{Y}$ is the np×nb desired output, $\overline{\mathbf{V}}$ is the np×R normalized firing strength matrix whose elements are obtained from Eq.(2.40) and Eq.(2.41). Matrix $\mathbf{W}$ is the R×nb consequent weight parameter to be identified, and $\mathbf{E}$ is the np×nb matrix of model errors. Hence, the solution of the estimated parameter $\hat{\mathbf{W}}$ can be obtained by ordinary least squares estimation taking the pseudo-inverse of the normal equation of Eq.(3.1),

$$\hat{\mathbf{W}} = (\overline{\mathbf{V}}^{T}\,\overline{\mathbf{V}})^{-1}\,\overline{\mathbf{V}}^{T}\,\mathbf{Y} \tag{3.2}$$

However, this ordinary least squares method suffers from the singular value problem which occurs when the matrix $\overline{\mathbf{V}}^{T}\overline{\mathbf{V}}$ is ill-conditioned or not invertable, in which case the estimated parameter $\hat{\mathbf{W}}$ will seriously be effected by round-off errors accumulated during calculation. In order to avoid the numerical problem, the orthogonal least squares method based on the classical Gram-Schmidt method is a better alternative of ordinary least squares computation [11] . In addition, this method provides information that can be used to restrict the model size.

The main idea of applying the orthogonal least squares method (OLS) to the NUFZY model is to perform fuzzy rule selection such that a set of R_s significant rules ($R_s \le R$), that make the maximum contribution to the variance of the desired output $\mathbf{Y}$ in Eq.(3.1), are selected from the initial R rule base. The orthogonal least squares method decomposes $\overline{\mathbf{V}}$ into $\mathbf{QA}$ such that Eq.(3.1) becomes

$$\mathbf{Y} = (\mathbf{QA})\cdot\mathbf{W} + \mathbf{E} = \mathbf{Q}\cdot\mathbf{G} + \mathbf{E} \qquad \text{with } \mathbf{G} = \mathbf{A}\cdot\mathbf{W} \tag{3.3}$$

where $\mathbf{Q} = [\mathbf{q}_1 \,..\, \mathbf{q}_r \,..\, \mathbf{q}_R]$ is a np×R matrix with orthogonal column vectors, $\mathbf{q}_r$ ($= [q_r(1) \,..\, q_r(t) \,..\, q_r(np)]^T$), i.e., $\mathbf{q}_i^T\mathbf{q}_j = 0$, for $i \neq j$, $1 \le i, j \le R$. Matrix $\mathbf{A}$ is a R×R invertable upper triangular matrix with 1's on the diagonal, i.e.,

$$\mathbf{A}=\begin{bmatrix}1 & \alpha_{12} & \alpha_{13} & \cdots & \alpha_{1R-1} & \alpha_{1R}\\ 0 & 1 & \alpha_{23} & \cdots & & \alpha_{2R}\\ 0 & 0 & 1 & \ddots & \ddots & \alpha_{3R}\\ 0 & 0 & 0 & \ddots & \alpha_{R-1R-1} & \vdots\\ \vdots & \vdots & \vdots & \vdots & 1 & \alpha_{R-1R}\\ 0 & 0 & 0 & 0 & 0 & 1\end{bmatrix} \quad \text{with } \alpha_{ir}=\frac{\mathbf{q}_i^T\bar{\mathbf{v}}_r}{\mathbf{q}_i^T\mathbf{q}_i}\text{; for } 1\le i\le r \tag{3.4}$$

where vector $\bar{\mathbf{v}}_r$ is the r^{th} column of $\bar{\mathbf{V}}$ defined by $[\,\bar{v}_r(1)\ ..\ \bar{v}_r(t)\ ..\ \bar{v}_r(np)\,]^T$ according to Eq.(2.41). Therefore, the least squares solution of Eq.(3.3) is given by

$$\hat{\mathbf{G}}=(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\mathbf{Y} \tag{3.5}$$

where $\hat{\mathbf{G}} = [\,\hat{\mathbf{g}}_1 .. \hat{\mathbf{g}}_n .. \hat{\mathbf{g}}_{nb}\,]$ with column vector $\hat{\mathbf{g}}_n = [g_{1n}\ ..\ g_{rn}\ ..\ g_{Rn}]^T$. The element g_{rn} is calculated as

$$g_{rn}=\frac{\mathbf{q}_r^T\mathbf{y}_n}{\mathbf{q}_r^T\mathbf{q}_r}=\frac{\sum_{t=1}^{np} q_r(t)y_n(t)}{\sum_{t=1}^{np} q_r^2(t)} \qquad 1\le r\le R,\quad 1\le n\le nb \tag{3.6}$$

Once the matrix $\hat{\mathbf{G}}$ is obtained, the orthogonal least squares solution $\hat{\mathbf{W}}$ is then given by

$$\hat{\mathbf{W}}=\mathbf{A}^{-1}\hat{\mathbf{G}} \tag{3.7}$$

Since matrix $\mathbf{A}$ is upper triangular, the inverse of $\mathbf{A}$ is easily achieved by backward substitutions.

The algorithm for orthogonal decomposition of $\bar{\mathbf{V}}$ into $\mathbf{QA}$ based on the classical Gram-Schmidt method (CGS) can be summarized as

$$\mathbf{q}_1 = \bar{\mathbf{v}}_1$$

$$\left.\begin{aligned} &\alpha_{ir}=\frac{\mathbf{q}_i^T\bar{\mathbf{v}}_r}{\mathbf{q}_i^T\mathbf{q}_i}, \quad 1\le i<r\\ &\mathbf{q}_r=\bar{\mathbf{v}}_r-\sum_{i=1}^{r-1}\alpha_{ir}\mathbf{q}_i\end{aligned}\right\} \quad \text{for} \quad r=2,..,R \tag{3.8}$$

3.3 FUZZY RULE REDUCTION

In general, better approximations can be attained by increasing the number of fuzzy rules - similar to increasing the number of hidden nodes in neural networks or the regressors in the linear regression problem -, but at the same time the redundancy in the rules increases, causing the problem of overfitting where the estimated parameters are heavily determined by the noise in the data which has a negative effect on prediction ability. Also, having too many fuzzy rules will make the final fuzzy rule base difficult to interpret. To solve this problem, the classical Gram-Schmidt OLS procedure can be used for rule selection, i.e., $\{ \bar{v}_r \mid 1 \le r \le R_s \le R \}$, such that the R_s significant fuzzy rules are extracted from the initial candidate rule base. In other words, the orthogonal least squares algorithm not only solves the unknown parameters, but also implicitly reveals a procedure for determining the structure of the fuzzy system. The following describes the principle and procedure of fuzzy rule selection, which is based on a criterion called 'error reduction ratio' [11] .

First, we consider the case of a single output, the n^{th} desired output vector, $\mathbf{y}_n$ (= $[y_n(1) .. y_n(t) .. y_n(np)]^T$). This is one column of $\mathbf{Y}$ and can be expressed as

$$\mathbf{y}_n = \mathbf{Q} \cdot \hat{\mathbf{g}}_n + \mathbf{e}_n$$

An estimate of the variance of output $\mathbf{y}_n$, after its mean has been removed, is given by

$$\frac{1}{np}\mathbf{y}_n^T\mathbf{y}_n = \frac{1}{np}\hat{\mathbf{g}}_n^T\mathbf{Q}^T\mathbf{Q}\hat{\mathbf{g}}_n + \frac{1}{np}\mathbf{e}_n^T\mathbf{e}_n = \frac{1}{np}\sum_{r=1}^{R} g_m^2\mathbf{q}_r^T\mathbf{q}_r + \frac{1}{np}\mathbf{e}_n^T\mathbf{e}_n \qquad (3.9)$$

where g_m is the element of $\hat{G}$ as defined in Eq.(3.6). It is seen that the term $(g^2_m\, \mathbf{q}^T_r\, \mathbf{q}_r)/np$ is the increment of the estimate of the variance of the desired output due to introducing an additional regressor, $\mathbf{q}_r$, which according to Eq.(3.8) follows from previous regressors and the fuzzy rule node output $\bar{v}_r$, while $(\mathbf{e}^T_n\mathbf{e}_n)/np$ is the unexplained variance of error. Once a set of regressors has been orthogonalized and added to the model, the contribution of each individual regressor to the desired output variance can be determined by the criterion 'error reduction ratio', [err], which is defined as

$$[\mathrm{err}]_m = \frac{g_m^2\mathbf{q}_r^T\mathbf{q}_r}{\mathbf{y}_n^T\mathbf{y}_n} \quad 1 \le r \le R \qquad (3.10)$$

Subscript m denotes the error reduction ratio of the r^{th} regressor with respect to the n^{th} desired output. It is noted from Eq.(3.9) that because the left hand side is fixed, the variance of the system error decreases whenever a new regressor is added to the model.

During the regressor selection process, at every step of the iterative procedure according to Eq.(3.8), the values of [err] of each candidate regressor will be calculated and only the one with the maximum [err] value is selected and added to the model. If the model contains certain highly correlated regressors, which means that they are almost linearly dependent, or redundant in the NUFZY model, it will result in an ill-conditioned problem. This condition can be detected by simply checking the norm of the newly added orthogonalized regressor to the model: $\|\mathbf{q}_r\|^2$. A value of $\|\mathbf{q}_r\|^2 = \mathbf{q}^T{}_r \cdot \mathbf{q}_r = 0$, simply implies that the newly selected regressor $\mathbf{q}_r$ is a linear combination of formerly selected orthogonalized regressors in the model. It therefore does not add to the information content of the model and should be kicked out. In practical the norm will almost never be equal to zero. Accordingly, a small value of 10^{-7} is specified as a threshold value of the norm $\|\mathbf{q}_r\|^2$ in order to check the almost linear dependence of orthogonalized regressors.

The orthogonalization procedure terminates when all candidate regressors have been processed. Among the results is a sequence of indices ranking the significant rules, and a list of indices of regressors which have been removed. Consequently, the total number of rules at the end, R_s , is less than the theoretical number of total rules belonging to the given input partition, provided almost linear dependent regressors did occur. A new set of weights is then calculated according to the selected linear independent rules that are used as a final rule base and the weights belonging to those almost linear dependent rules are set to be zeros.

In the case of multiple outputs, we can modify the error reduction ratio as the sum of $[err]_m$ and define a new criterion, $[ERR]_r$, for the MIMO system. i.e.,

$$[ERR]_r = \sum_{n=1}^{nb} [err]_m$$

One should be aware that the above criterion of 'error reduction ratio' for fuzzy rule selection only considers the performance of the model, i.e., the variance of residuals, and does not take into account the model complexity. Many possible alternatives of model subset selection criteria can be used that compromise the performance and complexity of the model, such as Akaike's information criterion AIC [8] , or a cross-validation based criterion [49] ‡. In spite of this, we will use this error reduction ratio as a criterion for fuzzy rule selection in this chapter to demonstrate the method.

Before proceeding, it is worthwhile to compare our present study to others. Previous works on application of OLS in neural networks [11] aimed at the selection of potential centers of radial basis functions from a large set of numerical data. The procedure of fuzzy rule selection therefore has a comparable function as the selection of centers in RBF neural networks. The procedures outlined above are inspired by the approach in [80] . In their work, the total number of fuzzy rules is initially equal to the number of input data, leading to a huge rule set. This set is then reduced by application of OLS, where the final number of rules is set arbitrarily on the basis of some subjective judgment. In contrast, our present study uses the

‡ As a matter of fact, when the maximum likelihood estimation is used for the model, the criterion AIC is asymptotically equivalent to the cross-validation criterion, see [55] .

OLS algorithm as a tool to reduce redundant or insignificant fuzzy rules, provided the prototype NUFZY model, i.e., the number of membership functions of the inputs, has been determined *a priori*. In practice, this may be an easier task than specifying in advance the number of required rules.

As shown in the previous chapter, a prototype of the fuzzy rule base can be constructed whenever the number of membership functions of each input variable, N_i, is chosen. N_i as well as the parameters in layer 1 (center, c, and bandwidth, σ) either can be determined by the designer based on experience, or alternatively the parameter values of c and σ can be estimated from the data set. In that case N_i becomes the only parameter that has to be assigned by the user. In the present study, we adopt the latter approach. The centers of membership functions are uniquely chosen by equal spacing in the range of x_i and a suitable value is taken as the bandwidth to ensure moderate overlap of the membership functions for each input x_i. During the identification procedure, parameters of layer 1, c and σ, are kept constant for the specified prototype fuzzy rule base and the OLS method is solely used to estimate the consequent weight parameter, **w**, of layer 3 and to delete redundant fuzzy rules. Hence, the determination of the most effective structure and the estimation of the optimal parameters (in the least squares sense) can be carried out at once by the OLS method. In summary, in addition to estimating the consequent weight parameters, the OLS algorithm mainly acts as a tool to eliminate redundant or insignificant fuzzy rules from the prototype NUZFY system.

3.4 EXAMPLES OF NONLINEAR STATIC SYSTEMS

The following subsections will present examples of the NUFZY identification for nonlinear static systems. Without loss of generality, only the single output case is considered. A notation marked as NUFZY($N_1 \times N_2 \times .. \times N_{ni}$; R_s; Gau/IMQ) represents a NUFZY model with x_1, x_2,.., x_{ni} inputs, where each input has N_1, N_2, .., N_{ni} Gaussian or IMQ membership functions, respectively, and the number of the final identified fuzzy rules is R_s. For example, NUFZY(3×5;10;Gau) means that there are two inputs x_1 and x_2; 3 Gaussian membership functions are assigned to x_1 and 5 to x_2; and the total number of identified fuzzy rules is 10.

3.4.1 Example 1 - synthetic nonlinear system

The NUFZY model is first applied to identify a nonlinear static MISO system given by

$$YD = \exp(-x_1) \times \sin(6x_2)/2 \tag{3.11}$$

A set of training data with 63 points is generated, by taking x_1 and x_2 randomly from the input space within the range of $[-1,1] \times [-1,1]$ and calculating the desired outputs using Eq.(3.11). These data are used to train the NUFZY model in order to get the estimated weights. Another independent data set, named validation data set, with a total of 441 points taken as grid points at 0.1 intervals in the range of $[-1,1] \times [-1,1]$, is used for verifying the trained NUFZY model. In this example, 4 membership functions are assigned to input variable x_1 and 6 to x_2. Both Gaussian and IMQ membership functions are investigated in this example. The centers of these membership functions are taken as equally spaced according to Eq.(2.38), whereas the bandwidth is assigned as the variance value of x_i in the case of Gaussian, and three times the variance of x_i in the case of the IMQ membership function.

Using the training data set, the development of the sum of squared error (sse) during the OLS identification is depicted in Figure 3.1. It is obvious that the value of sse decreases at every addition of a newly selected rule to the fuzzy rule base. At the end of the OLS procedure, using the IMQ membership function, a NUFZY(4×6;20;IMQ) model is found. This means that out of 24 possible rules, there are 4 (almost) linear independent rules that have been removed by OLS. With Gaussian membership functions, however, the result is (4×6;24;Gau), i.e., no rules have been deleted in this case. Using the identified weights, both NUFZY models are verified by applying them to the validation data set. Figure 3.2 and Figure 3.3 show the performance in reconstructing the nonlinear surface expanded by x_1 and x_2 as described in Eq.(3.11). Good interpolation is obtained by the identified NUFZY model. The squared error plot, Figure 3.2.(d) and Figure 3.3.(d) show that NUFZY with IMQ membership functions performs better than that with Gaussian membership functions despite less fuzzy rules.

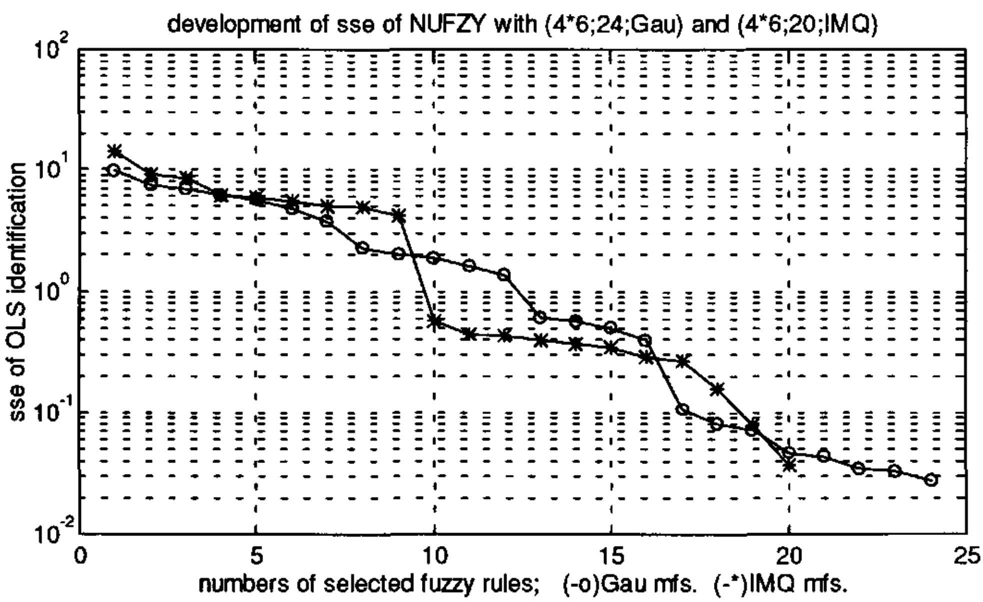

Figure 3.1: The development of SSE of the NUFZY model with NUFZY(4×6;24;Gau) (line-circle) and NUFZY(4×6;20;IMQ) (line-star) during the identification process

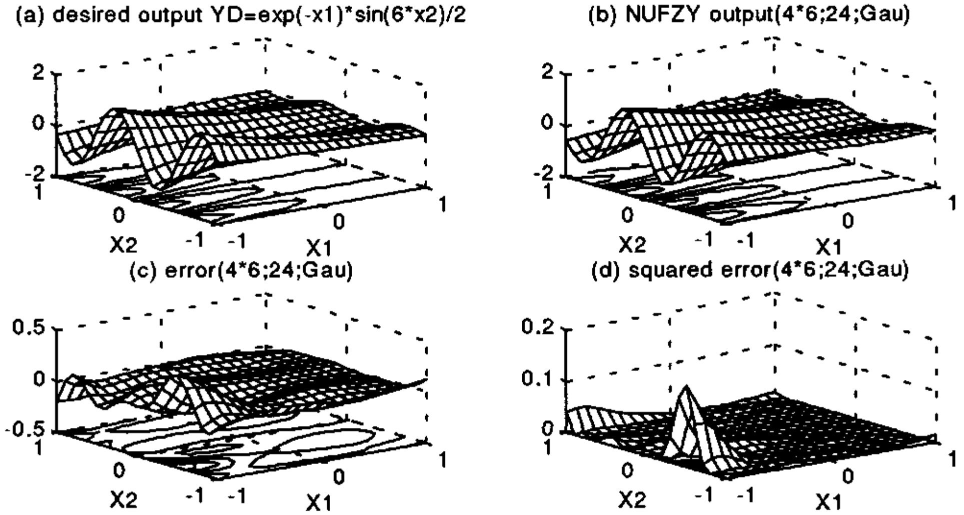

Figure 3.2: Validation of the NUFZY model with NUFZY(4×6;24;Gau). (a) desired output (b) NUFZY model output (c) error surface (d) squared error surface.

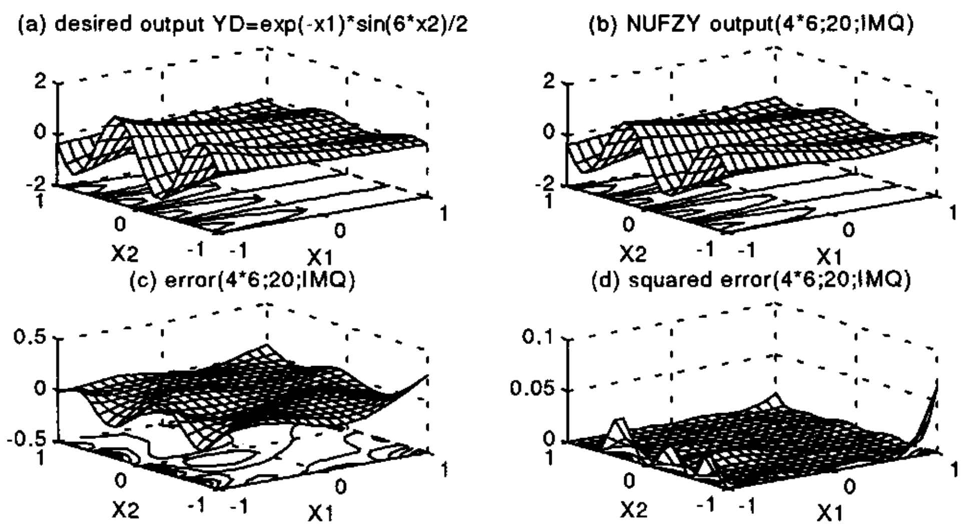

Figure 3.3: Validation of the NUFZY model with NUFZY(4×6;20;IMQ). (a) desired output (b) NUFZY model output (c) error surface (d) squared error surface.

In order to demonstrate the efficient performance of the NUZFY system as a function approximating network, a comparison is made with two layered neural networks which are trained by back-propagation algorithms. The simulations of these feedforward neural networks are performed by the neural network toolbox of Matlab® [15] . Four neural networks, denoted as 2-10-1 NN, 2-20-1 NN, 2-30-1 NN, and 2-40-1 NN , are set up, each with two inputs and one output, and 10, 20, 30, and 40 neurons in the hidden layer, respectively. The nonlinear functions of the neurons used in the hidden layer are tan-sigmoid functions and those in the output layer are linear functions. The back-propagation training is initialized by the Nguyen-Widrow initialization. Moreover, a momentum term and an adaptive learning rate to speed up the procedure are used. During the identification procedure, the training is halted when the sse value of the neural net is less than 0.04 or when the training arrives at 5000 epochs (iterations). The value of 0.04 is estimated according to the maximum sse value obtained by the NUFZY model with the IMQ membership function during the OLS training procedure. Unfortunately, despite long training times, most of these networks are still unable to reach the requirement of a sse being less than 0.04 within 5000 epochs. After the training , the networks are verified by the validation data set. The results are shown in Figure 3.4.

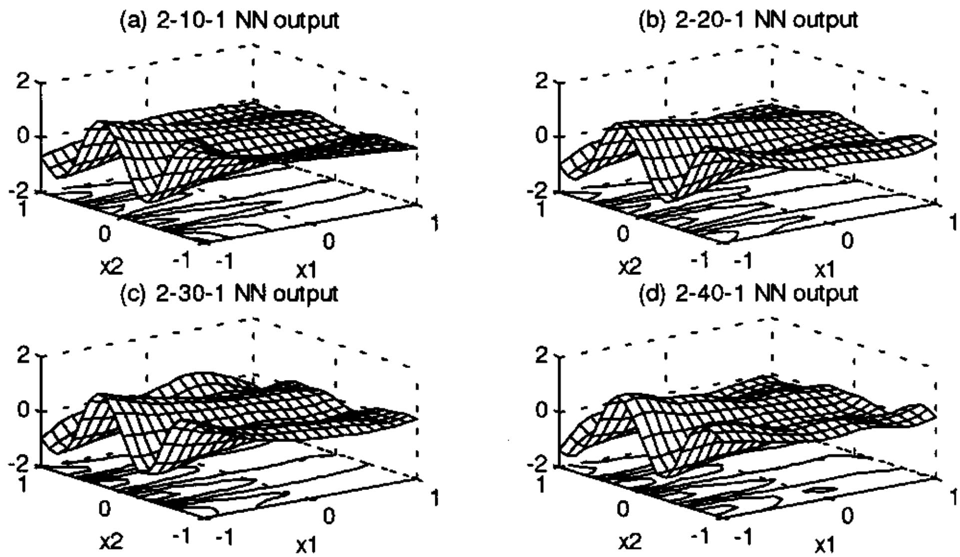

Figure 3.4: The results of validation procedure of neural networks with different neurons in the hidden layer. (a) 10 neurons, (b) 20 neurons, (c) 30 neurons, and (d) 40 neurons.

The numbers of the parameters required for the neural networks with back-propagation are 32, 62, 92, and 122, respectively. In constrast, the identified parameters of the NUFZY model are 24 for the Gaussian and 20 for the IMQ membership function. It can be seen that the predetermined input parameters (centers and widths) together with the OLS algorithm makes it easy to identify a NUFZY network with less free parameters and yet results in better approximation as compared to ordinary neural networks in conjunction with back-propagation training.

3.4.2 Example 2 - tomato production

In this example, the NUFZY model is applied to an agricultural problem which concerns the dry matter production of tomato. Data were obtained from three experiments of tomato in multispan Venlo-type greenhouses in Wageningen [2] . The temperature set-point was maintained at 18 °C day and night and no CO_2 enrichment was used. Environmental factors, such as indoor temperature, relative humidity, CO_2 concentration, and outdoor global radiation were recorded hourly. During the period of 25 January to 23 November, 1988, three experiments were done. In each experiment, six tomato plants were sampled for destructive measurement at about weekly intervals. In this identification example, accumulated radiation (W/m^2) over time t and averaged CO_2 concentration (ppm) from the initial date to time t are taken as input 1 and 2, respectively, and the total dry weight (TDW) of tomatoes (kg/m^2) at time t as the output for the NUFZY model. Data of experiments 1 and 3 are used for training in order to identify the underlying tomato production and those of experiment 2 are used for validation of the NUFZY model. By trial and error, eight IMQ membership functions have been assigned to input 1, and four to input 2 which together construct a prototype fuzzy rule base with a total number of 32 rules. The centers of the membership functions are determined as equally spaced and widths are taken equal to the value of the standard deviation of the inputs. At the end of the OLS procedure it appears that only 10 rules are left.

The results of the TDW measurements and the NUFZY predicted output are depicted in Figure 3.5. As shown in Figure 3.5.(a), (b) and (c), most of the NUFZY predicted outputs are located within the 95% confidence interval of measured TDW. It demonstrates the ability of the NUFZY approach to identify the tomato production process. Figure 3.5.(d) shows a prediction according to NUFZY(8×4;10;IMQ) of how expected TDW of tomato is related to accumulated radiation and period averaged CO_2 concentration. The figure suggests that TDW increases as accumulated radiation increases, and that TDW is higher when the averaged CO_2 over the growth period has been higher.

It should be noted that the model result is restricted to a limited CO_2 range because no CO_2 dosage has been applied. Average CO_2 is used rather than instantaneous CO_2 because the latter is influenced by the instantaneous irradiation due to plant photosynthesis, and therefore is not a truely independent variable. The CO_2 effect may also partly contain a temperature effect, because in the raw data, despite temperature control, there was a slight negative correlation between prevailing CO_2 and temperature. However, the NUFZY model does not contain more than 10 rules to describe the complex process of tomato production. This compares favorably with sophisticated models like TOMSIM and TOMGRO [2] .

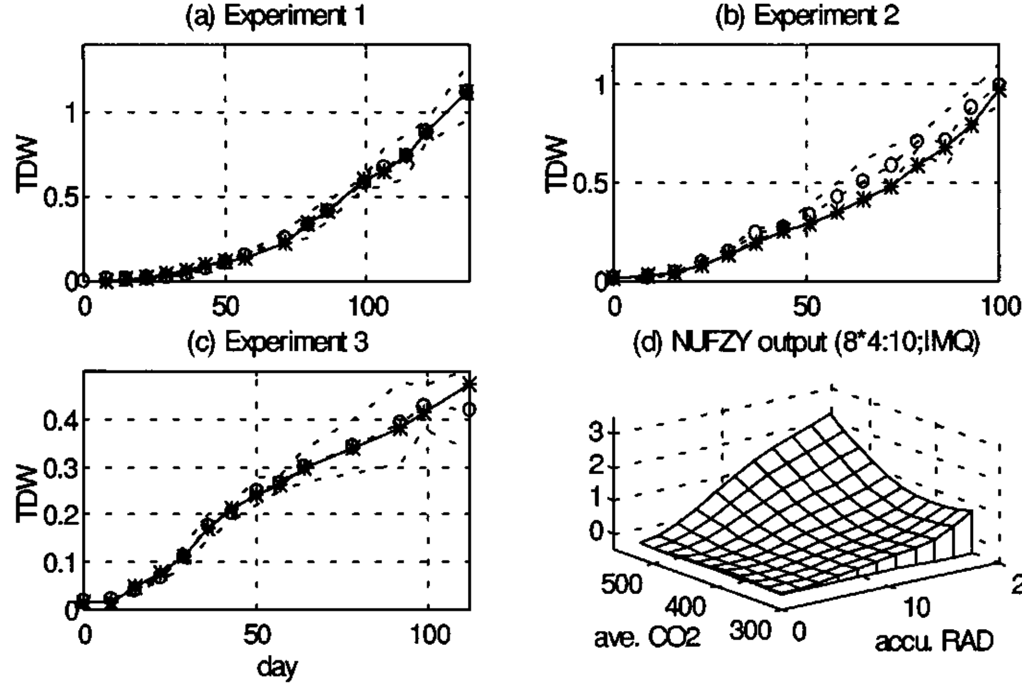

Figure 3.5: The measured TDW of tomato (circle-dotted line) with 95% confidence interval (dotted line) and NUFZY simulated output (star-dotted line) for experiments 1 (a), 3 (c). (identification data) and 2 (b) (validation data). (d) The predicted TDW by NUFZY(8×4;10;IMQ) is related to accumulated radiation (×1000 W/m^2) and the CO_2 concentration (ppm) averaged over the period from the first day to the measurement day.

3.5 EXAMPLES OF NONLINEAR DYNAMIC SYSTEMS - APPLICATIONS TO AGRICULTURAL PROBLEMS

In the optimal control of greenhouse crop production, one has to deal with different time scales: slow crop growth and fast greenhouse physics. In the work of [77] , a two-time scale decomposition using a singular perturbation method was employed to solve the optimal control problem. The same idea is used in examples of this section to separate the crop growth problem and the greenhouse physics problem. For crop growth, the fast dynamics of climatic conditions are regarded as irrelevant and one can utilize the mean climatic values for the crop growing period. On the other hand, during the short term identification of the greenhouse climate, the states of the crop are assumed to be constant.

From the point of view of control applications the models should have predictive ability. Accordingly, in the following two examples, NUFZY models are established to predict the lettuce growth and the greenhouse temperature. In the development of models, a set of data is first gathered from previous experiments and taken as the training set, so that the consequent weight parameters of the NUFZY model can be identified by the above OLS method. With these identified parameters, another independent set of data, named validation set, is used to evaluate the prediction ability of the identified NUFZY model.

3.5.1 Identification of lettuce growth process

3.5.1.1 Problem description

In [77] , a lettuce growth model is described by a single state variable, namely total dry weight X_d. The governing differential equation is

$$\frac{dX_d}{dt} = c_\beta (c_\alpha \phi_{phot} - \phi_{resp}) \tag{3.12}$$

where c_β and c_α are conversion parameters; ϕ_{phot} and ϕ_{resp}, represent gross photosynthesis gain and maintenance respiration loss of lettuce, respectively. They are complex nonlinear functions of the dry weight itself and several input variables, e.g., greenhouse indoor CO_2 concentration (denoted as Z_c), greenhouse indoor air temperature (denoted as Z_t), and outdoor radiation (denotes as V_i). Parameters of c_α and c_β in these relationships were determined empirically. Further details can be found in [77] . Two experiments were done to calibrate the model parameters and to validate the above model. The lettuce used in the experiments is *Lactuca sativa* L., which was grown in an experimental greenhouse in Wageningen from 17/10/1991 - 16/12/1991 (cultivar 'Berlo') and from 21/1/1992 - 17/3/1992 (cultivar 'Norden'). The greenhouse was under computer control according to the rules used in normal Dutch horticultural practice. During the two experiments, destructive measurements of the lettuce (10 and 9 sampling dates, respectively) were performed, whereas the greenhouse climate, the actuators and the outdoor climate conditions were recorded for further analysis.

Eq.(3.12) defines a continuous-time model of the lettuce growth rate, which is a relationship between the biomass and the external input variables. In the set up of the NUFZY model, the same variables are used. First, the time step is taken as one day (24 hours), using the daily averaged measurements of V_i , Z_t and Z_c as approximations of the input signals. Due to the fact that the lettuce dry weight is not available every day for these experiments, linearly interpolated values are taken as estimates of the data. Hence, a modified discrete-time model, based on daily averaged data of V_i, Z_t and Z_c and linear interpolated X_d, is applied to approximate the continuous growth of lettuce. It is represented as

$$\frac{\Delta X_d(k)}{\Delta T(k)} = f_1(X_d(k), Z_c(k), Z_t(k), V_i(k)) = \frac{X_d(k+1) - X_d(k)}{T(k+1) - T(k)} \tag{3.13}$$

where k is the discrete time index, $X_d(k)$ is the dry weight of lettuce at day T(k); $f_1(\cdot)$ represents the nonlinear function to be identified. Therefore, a one-day-ahead prediction of lettuce dry weight according to Eq.(3.13) can be obtained as

$$X_d(k+1) = X_d(k) + f_1(X_d(k), Z_c(k), Z_t(k), V_i(k)) \cdot \Delta T(k) \tag{3.14}$$

An alternative formulation of a one-day-ahead prediction of lettuce dry weight is to incorporate the previous values of inputs in the nonlinear function directly,

$$X_d(k+1) = f_2(X_d(k), Z_c(k), Z_t(k), V_i(k)) \tag{3.15}$$

3.5.1.2 NUFZY model establishment

In [77] , a sensitivity analysis has been done on the lettuce growth model of Eq.(3.12) suggesting that dry matter production of lettuce is mainly dominated by a limited number of inputs. In particular, the CO_2 concentration has a stronger positive effect on dry matter production than greenhouse indoor air temperature. Because the temperature in the greenhouse was controlled in order to keep it within operation bounds, the variation of the temperature is too small to be informative. Besides, it can be seen that the indoor air temperature strongly correlates to the outdoor radiation. Hence, on establishing the NUFZY model, the function variables have been restricted to outdoor radiation V_i, indoor CO_2 concentration Z_c, and the present state of crop X_d. In practice, a derived quantity, temperature-day, is commonly used, which suggests that the summation of temperature values over the growing periods should be taken into account. In terms of outdoor radiation V_i, this temperature-day quantity can be replaced by another quantity, accumulated radiation, $\sum V_i(k)$ (denoted as $ACV_i(k)$ in below), which sums up the daily averaged radiation over time T(k). Therefore, for the NUFZY modeling, the chosen variables are X_d, Z_c and ACV_i. Formally, the goal of the identification of the lettuce growth is to establish the NUZFY models, f_{NUFZY1} and f_{NUFZY2}, such that they can approximate the unknown nonlinear functions f_1 and f_2 in Eq.(3.14) and Eq.(3.15), respectively. Here, we call f_{NUFZY1} and f_{NUFZY2} the first kind and the second kind of NUFZY modeling, respectively. It can be seen that in the first kind of NUFZY modeling the one-step-ahead prediction of lettuce growth is obtained indirectly based on the inferred growth rate, whilst the second kind of NUFZY modeling infers the one-step-ahead prediction of lettuce growth directly. The predicted dry weight of lettuce by the NUFZY model, $\hat{X}_d(k+1)$, can be written as

The first kind of NUFZY modeling

$$\hat{X}_d(k+1) = X_d(k) + f_{NUFZY1}(X_d(k), Z_c(k), ACV_i(k)) \cdot \Delta T(k) \tag{3.16}$$

and the second kind of NUFZY modeling

$$\hat{X}_d(k+1) = f_{NUFZY2}(X_d(k), Z_c(k), ACV_i(k)) \tag{3.17}$$

When using the model in a real application, the biomass is usually not observed. Therefore, the validation was done in such a way that no measurements of X_d were employed except for the initial point. Thus, on validation of the NUFZY model, Eq.(3.16) and Eq.(3.17) become

$$\hat{X}_d(k+1) = \hat{X}_d(k) + f_{NUFZY1}(\hat{X}_d(k), Z_c(k), ACV_i(k)) \cdot \Delta T(k) \tag{3.18}$$

and

$$\hat{X}_d(k+1) = f_{NUFZY2}(\hat{X}_d(k), Z_c(k), ACV_i(k)) \tag{3.19}$$

where the input variable $\hat{X}_d(k)$ is obtained from the prediction of the trained NUFZY model at the previous step k-1. At the next step, the present estimate $\hat{X}_d(k+1)$ is iteratively fed into the model as input. The validation by this method therefore tests the ability of the model to predict n-steps ahead. This procedure, sometimes called the parallel method [48] , is a much more severe test than the serial-parallel method, as in Eq.(3.16) and Eq.(3.17), which just takes the real value of X_d into the NUFZY model. The training and validation process of NUFZY modeling is given as follows.

3.5.1.3 Training process

The training of the NUFZY model is done with data taken from experiment 1 (17/10/1991 - 16/12/1991, in total 61 days). In this set of training data, the values of daily averaged CO_2 concentration Z_c (ppm) and accumulated daily averaged outdoor radiation ACV_i (W/m^2) as well as the linear interpolated dry weight of lettuce X_d (g) are treated as external inputs. Hence, in total, 60 tuples [$X_d(k)$, $Z_c(k)$, $ACV_i(k)$, $\Delta X_d(k)/\Delta T(k)$] or [$X_d(k)$, $Z_c(k)$, $ACV_i(k)$, $X_d(k+1)$] are used to train the first and the second kind of NUFZY modeling, respectively. From the training set, when each N_i is assigned (see below), the centers of the IMQ membership functions (c in Eq.(2.36)) are taken equally spaced in each input range, whilst to the width, σ, of the membership functions of each input three times the value of the standard deviation is assigned. The consequent weight parameter $\mathbf{w}_1$ and $\mathbf{w}_2$ of the NUFZY models, f_{NUFZY1} and f_{NUFZY2}, are estimated by the OLS method such that the sum of square errors $\Sigma_k[\Delta X_d(k)/\Delta T(k) - f_{NUFZY1}(\cdot;\mathbf{w}_1)]^2$ and $\Sigma_k[X_d(k+1) - f_{NUFZY2}(\cdot;\mathbf{w}_2)]^2$ are minimized, where k = 1 to 60. The number of membership functions for each input, N_i, (i = 1, 2, and 3, corresponding to X_d, Z_c, and ACV_i, respectively) varies from 2 to 4. Therefore, the initial number of fuzzy rules ranges from 2×2×2 = 8 to 4×4×4 = 64. By taking the sum of squared errors as a criterion function, the final structure used is the one that has the lowest sum of squared errors.

3.5.1.4 Validation process

After obtaining the consequent weight parameter $\mathbf{w}_1$ and $\mathbf{w}_2$, another set of data taken from experiment 2 (21/1/1992 - 17/3/1992, in total 56 days) is used to validate the first and second kind of NUFZY models. Since the values of $X_d(k)$ is unknown at every moment except the sample dates, the validation process is carried out according to the parallel method. The mean value of the measured X_d on the first sample date, 0.15 g, is used as an initial value $X_d(1)$. Values of the other inputs $Z_c(k)$ and $ACV_i(k)$ are measurable for the whole experiment period. So, after the first step, the estimate $\hat{X}_d(k)$ is obtained from Eq.(3.18) or Eq.(3.19) with available measurements $Z_c(k-1)$ and $ACV_i(k-1)$ and the initial value $X_d(1)$ (using the parameter set $\mathbf{w}_1$ or $\mathbf{w}_2$), whereas in the following sequence, this estimate $\hat{X}_d(k)$ is iteratively fed into Eq.(3.18) or Eq.(3.19) together with $Z_c(k)$ and $ACV_i(k)$, in order to generate the next step prediction of $\hat{X}_d(k+1)$.

3.5.1.5 Results

Figure 3.6 and Figure 3.7 demonstrate the results of both models for the training and the validation process. The results of the training process show that the first kind of NUFZY modeling, denoted as NUFZY1(3×4×3:15;IMQ), can achieve a good approximation, where the notation NUFZY1(3×4×3:15;IMQ) indicates that 3, 4, and 3 IMQ membership functions are assigned to X_d, Z_c, and ACV_i, respectively. As a result of the OLS identification, only 15 fuzzy rules are chosen to be involved in the fuzzy rule base and, consequently, 21 redundant rules have been removed from the initial 36 (= 3×4×3) rules. The best result of the second kind of NUFZY modeling is NUFZY2(2×2×3:10;IMQ), so that X_d, Z_c, and ACV_i have 2, 2 and 3 IMQ membership functions, respectively, and the number of identified fuzzy rules is only 10. For the purpose of comparison, in these figures the mean values of the dry weight measurements with 95% confidence intervals are plotted as vertical bars and the simulated results from [77] , by Eq.(3.12), are presented too. It can be seen that both results obtained from $f_{NUFZY1}(\cdot;\mathbf{w}_1)$ and $f_{NUFZY2}(\cdot;\mathbf{w}_2)$ perform as well as the mechanistic model Eq.(3.12) during the training process and are slightly less accurate compared to the result of the mechanistic model in the validation process. However, the number of estimated parameters (15 and 10, respectively) in the NUFZY models compares favorably to the 19 parameters of the mechanistic model.

Among the identified fuzzy rules, one example (Rule 6) of the generated fuzzy rules of the first kind of NUFZY modeling is listed below, which suggests that when the crop is in the initial stage of development (X_d is small, A_1, and ACV_i is low, C_1), a very high CO_2 concentration (Z_c is B_4) will result in a negative effect on the growth rate. Other identified rules are similar to it and will not be presented here.

Rule 6: **IF** $X_d(k)$ is A_1 **AND** $Z_c(k)$ is B_4**AND** $ACV_i(k)$ is C_1 **THEN** $\Delta X_d(k)/\Delta T(k) = -2.0663$

The centers and widths of linguistic variables [A_1 A_2 A_3] of input $X_d(k)$ (g) are [0.15 3.38 6.61] and [1.9532 1.9532 1.9532], respectively. For $Z_c(k)$ (ppm) with the linguistic variables [B_1 B_2 B_3 B_4], centers and widths are [422.65 498.05 573.46 648.86] and [61.10 61.10 61.10 61.10], respectively; whereas for the linguistic variables [C_1 C_2 C_3] of $ACV_i(k)$ (W/m^2), [21.62 420.01 818.41] and [210.47 210.47 210.47], respectively. Linguistic variables of [A_1 A_2 A_3] represents small, medium, and large, respectively. [B_1 B_2 B_3 B_4] represents low, medium, high, and very high; [C_1 C_2 C_3] represents low, medium, and high.

Some remarks about this example are made below. In this example, we have used two NUFZY models, f_{NUZFY1} and f_{NUFZY2}. The above results show that both of them have comparative prediction ability as that of Eq.(3.12), a mechanistic model describing the lettuce growth. Yet, the prediction behavior of the two forms is different. It is noticed from Figure 3.6.(b) that the increment model f_{NUZFY1} leads to a smoother growth curve than the direct model f_{NUZFY2} in Figure 3.7.(b). However, it should be kept in mind that if the sampling interval becomes large (i.e., less sampling is done), the linear interpolation used for training of both NUFZY models should be circumvented since it leads to a model that tries to mimic a piecewise linear growth curve rather than the true growth curve of lettuce. In this example, however, the sampling interval is around one week, which does not cause severe problems in this respect.

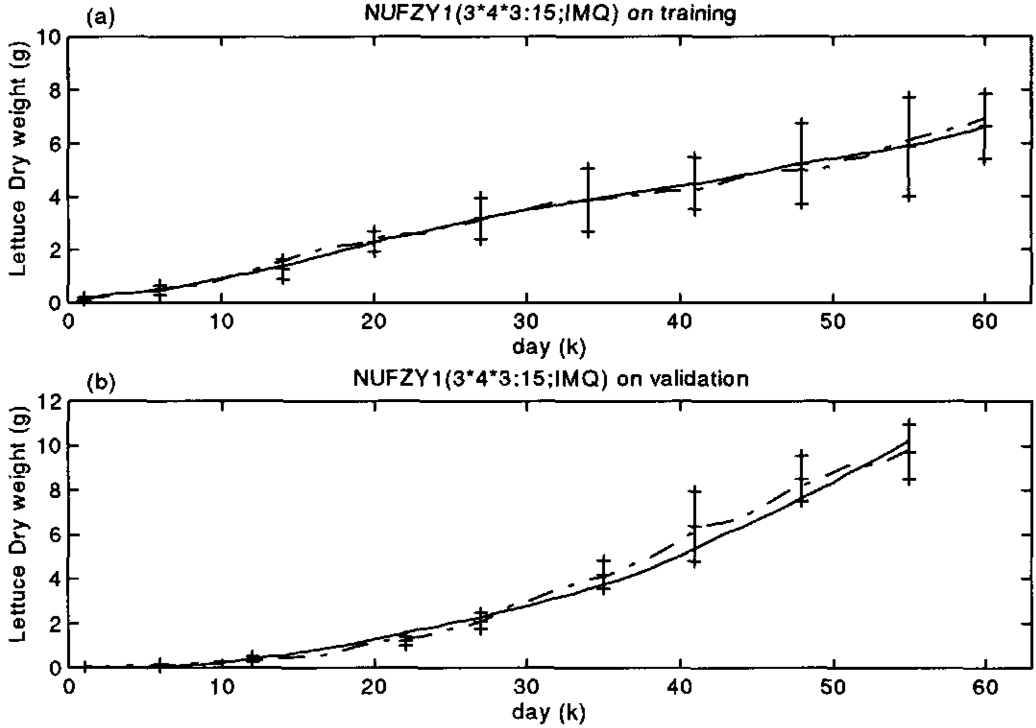

Figure 3.6: Prediction of lettuce dry weight by NUFZY1(3×4×3:15;IMQ) during experiment 1, training (one-step-ahead) (a); and experiment 2, validation (seasonal prediction) (b). Solid line indicates output from NUFZY1 and dash-dot line indicates result from the one state variable model, Eq.(3.12), described in [77] . The vertical bars show a 95% confidence interval around the mean value of the measurements.

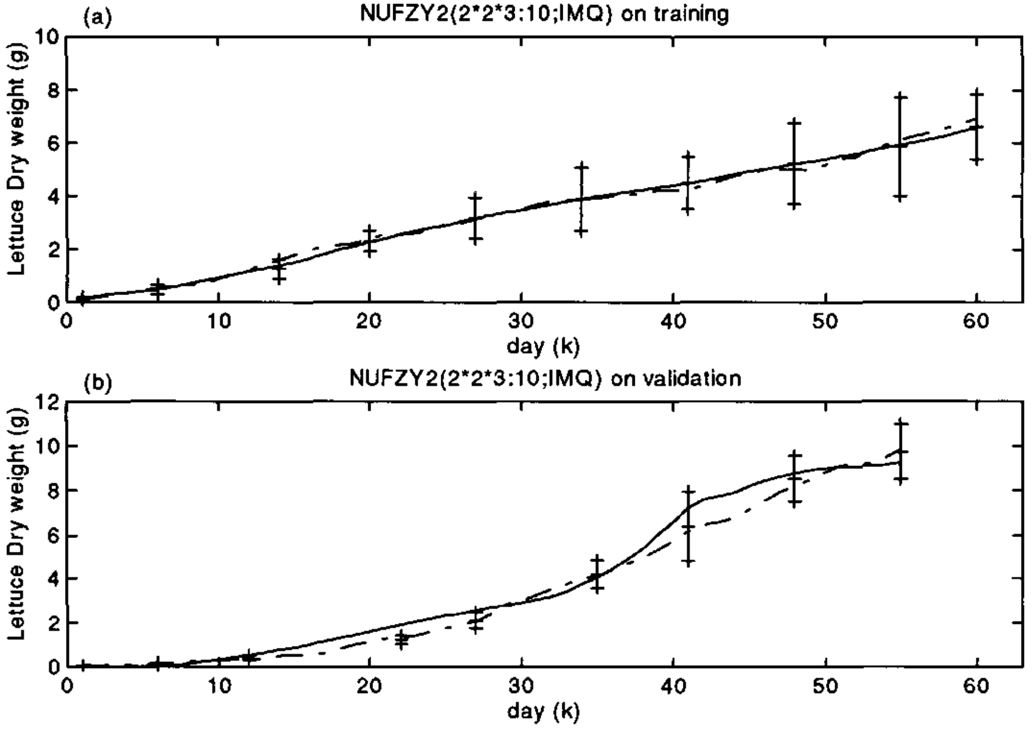

Figure 3.7: Prediction of lettuce dry weight by NUFZY2(2×2×3:10;IMQ) during experiment 1, training (one-step-ahead) (a); and experiment 2, validation (seasonal prediction) (b). Solid line indicates output from NUFZY2 and dash-dot line indicates result from the one state variable model, Eq.(3.12), described in [77] . The vertical bars shows a 95% confidence interval around the mean value of the measurements.

Another factor of interest is that although the models are trained to yield one-day-ahead predictions, the validation is done by running the model for the whole growing season of experiment 2 without updating from the actual measurements (a parallel method). In this approach, there might exist a risk that the prediction becomes worse because of the accumulation of model errors. In this particular example, the results remain within the measurement uncertainty, thus indicating that, from the training data set, the NUFZY model matches the lettuce growth dynamics sufficiently well to achieve a reasonable seasonal prediction, which can be used, e.g., in production scheduling.

3.5.2 Identification of greenhouse temperature dynamics

In contrast to the slow response of crop growth, the greenhouse climate shows very distinct fast dynamics. Based on a time scale in minutes, the effect of crop growth is assumed to be negligible. In the following example, the NUFZY model demonstrates the modeling of the dynamics of the greenhouse temperature and performs a one-step-ahead prediction based on the present indoor states, control inputs, and outdoor disturbances. Extensions to other climatic states is possible but will not be discussed here.

3.5.2.1 Problem description

For control purposes, several dynamic models of greenhouse physics have been developed. Among them, models based on first principles such as [4] , and more recently [13] ; others are transfer function models such as [74] [75] or intermediate [65] . In [64] based on energy and mass balances, the dynamics of the greenhouse temperature, denoted as Z_t, is described by a first order differential equation with heat losses from natural ventilation, through the roof cover and to the soil; and heat gains from heating pipes and solar radiation. The differential equation is written as

$$\frac{dZ_t}{dt} = \frac{1}{c_g}[(c_v + c_r)(V_t - Z_t) + c_s(Z_s - Z_t) + c_p(Z_p - Z_t) + c_i V_i] \tag{3.20}$$

where c_g is the greenhouse heat capacity, and c_v, c_r, c_s, and c_p are the effective heat transfer coefficients of ventilation, roof cover, soil and heating pipe, respectively. V_t, Z_s, and Z_p stand for outdoor air temperature, indoor greenhouse soil temperature and heating pipe temperature, respectively. c_i and V_i are radiation efficiency factor and disturbance from outdoor radiation, respectively. In Eq.(3.20), the control of window opening U_w and two disturbances, outdoor wind speed V_s and wind direction V_d, are implicitly involved in the coefficient c_v, whereas control of the heat supply U_t determines the heat pipe temperature Z_p. Hence, taking these implicit relations into account, Eq.(3.20) can be formally written as

$$Z_t(k+1) = f_3(Z_t(k), Z_s(k), U_w(k), U_t(k), V_t(k), V_s(k), V_d(k), V_i(k)) \tag{3.21}$$

where $f_3(\cdot)$ is a complex nonlinear function describing the greenhouse temperature dynamics as a function of the input variables described above.

3.5.2.2 NUFZY model establishment and results

In this example, greenhouse climate data are taken from the experimental results of [63] . The experiment of tomato production is done where one greenhouse compartment is controlled by a receding horizon optimal control (RHOC) algorithm in order to compare it to another compartment controlled by a current commercial greenhouse climate control computer. The experiment was conducted from 1/8/1995 to 30/10/1995. All the input variables mentioned in

Eq.(3.21) were measured every minute. In the following, the training and validation data used for the NUFZY modeling make use of measurements and records taken from the optimal controlled greenhouse compartment.

From Eq.(3.21), it is seen that the large number of input variables causes a large fuzzy rule base for the NUFZY model. For ease of modeling, the higher order effects induced by the state of soil Z_s are neglected. It is also observed that the windward and lee side windows are almost fully open every day in August because of the high air temperature inside the greenhouse. Hence, in this demonstration, the input variables, U_w, V_s, and V_d, can be further left out by properly choosing those days on which the windows are open. Therefore, a NUFZY model with restricted validity is established to give a one-step-ahead prediction of Z_t with reduced input variables as

$$\hat{Z}_t(k+1) = f_{NUFZY3}(Z_t(k), U_t(k), V_t(k), V_i(k)) \tag{3.22}$$

where the input variables are greenhouse temperature Z_t (°C), heating valve opening U_t (0 - 100 %), outdoor air temperature V_t (°C) and radiation V_i (W/m^2). The effect of the disregarded input variables will appear as unmodeled effects in the model errors.

A set of data originating from 23/8/1995 - 25/8/1995 is taken as training data. During this period, the windows are almost always fully open. In order to reduce the amount of data to be processed, measurements are taken for analysis every five minutes. The training process, similar to the previous example, is carried out by the OLS method. Independent data sets on several dates in three different periods of the experiment, *viz.* in the beginning (21/8 - 22/8, 26/8 - 27/8, and 31/8 - 4/9), halfway (21/9 - 25/9), and toward the end (16/10 - 20/10), are taken to validate the identified NUFZY model. In contrast to the previous example, only the one-step-ahead prediction capability was tested, using the measured temperature as the input for the next prediction (serial-parallel prediction).

Results of the training and validation process are shown in Figure 3.8. The identified model is NUZFY3(3×4×3×2:31;IMQ), showing that, to input variables Z_t, U_t, V_t, and V_i are assigned 3, 4, 3, and 2 IMQ membership functions, respectively, and the final number of identified fuzzy rules is 31. For the training data process, Figure 3.8.(a) shows that a fairly good fit is obtained with this reduced set of fuzzy rules, since 41 rules are removed from the prototype rule base (3×4×3×2 =72). Figure 3.8.(b) illustrates that the relative error of the residuals is less than ±10% (around ± 2°C). The larger discrepancies mainly occur when outdoor radiation disturbance has large fluctuations, i.e., during day time. Other factors, such as wind speed and direction that are not used in the modeling, may have considerable influence on the discrepancy too. The validation results in the beginning period (Figure 3.8 (c) date 21/8 - 22/8, (d) 26/8 - 27/8, and (e) 31/8 - 4/9) possess a fair fit to the measured data. However, larger discrepancies can be found in the other two periods. This result confirms that black-box models are firmly data dependent and need retraining as the process slowly moves to other operating regions.

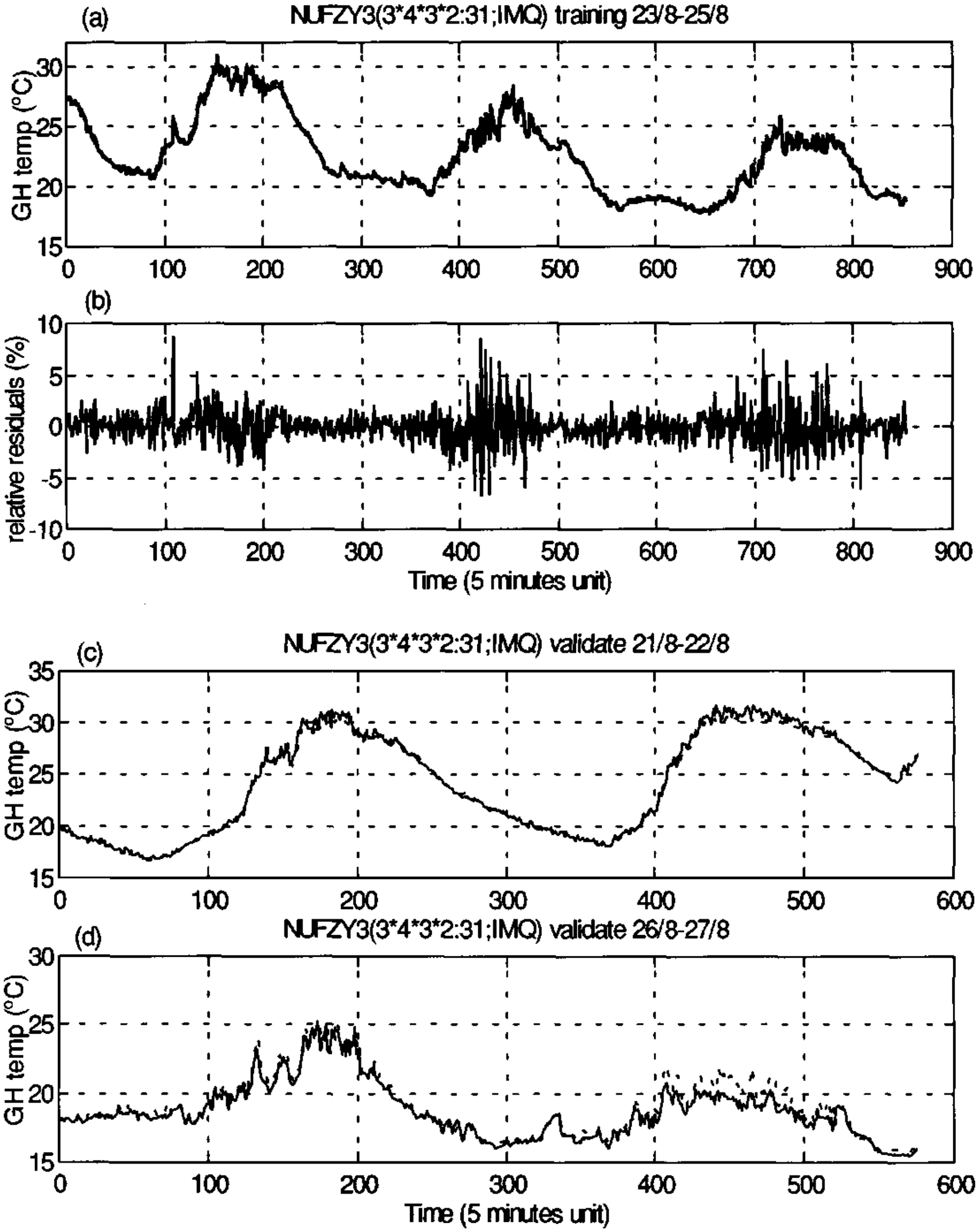

Figure 3.8. (a)- (d) (to be continued)

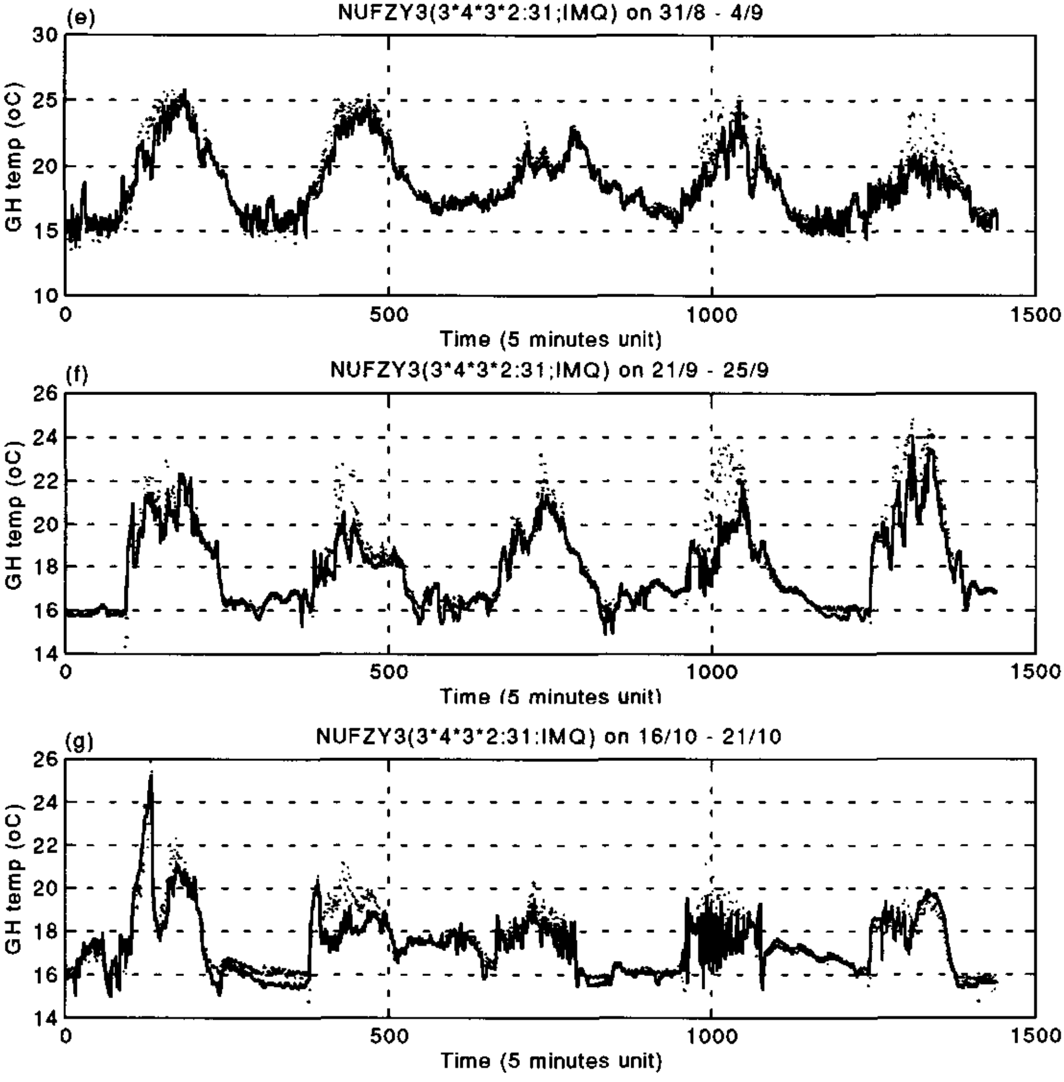

Figure 3.8: Simulated one-step-ahead prediction of greenhouse temperature by NUFZY3(3×4×3×2:31;IMQ) during 23/8 - 25/8 , training process (a) and relative residuals of training (b); validation on date 21/8 - 22/8 (c); 26/8 - 27/8 (d); 31/8 - 4/9 (e); 21/9 - 25/9 (f) and 16/10 - 20/10 (g). Solid line: measured greenhouse temperature; dotted line: NUFZY prediction. In figures (a) and (c), it is hard to spot the difference between the measurement and the NUFZY prediction.

It is worthwhile to make some remarks on this simplified example of identifying the greenhouse temperature. It is the fast fluctuation (within the 5 minutes period) of outdoor radiation during day time that results in larger model errors. Also, by neglecting the effects of window opening and wind speed, the NUFZY model occasionally tends to overestimate the greenhouse temperature. For instance on 27/8 in Figure 3.8.(d) and on 24/9 in Figure 3.8.(f), it can be verified that the wind speed increases from the range 0 - 6 m/s (used in the training process) to 0 - 10 m/s. This creates an extra heat flux due to extra ventilation out of the real greenhouse which, of course, does not occur in the present model. As identification is done by off-line training, this phenomenon confirms, again, that the NUFZY model is data dependent - just like any other black box approach - and that the model can only be expected to perform well if it is used under similar conditions as encompassed in the training process. In order to construct a more general model for long term prediction, further study is needed to involve more climate factors into the NUFZY model. Or, alternatively, for control purposes, a recursive identification can be used which adjusts the model to the most recent data. For example, in [70] a recursive adaptation of the parameters of the NUFZY model has been studied, showing the feasibility for on-line application. This recursive learning scheme will be presented in next chapter.

3.6 DISCUSSION

On the basis of the experience with the previous examples, some aspects of application of the NUFZY model deserve further attention. First, the construction of the antecedent part of NUFZY is based on some *a priori* knowledge such that one is able to assign the number of membership functions likely to be needed for each input variable. The question of how many membership functions can both partition the input space well and establish a moderate size of the fuzzy rule base, concerns a trade-off between accuracy and complexity. As mentioned before, several criteria can be used for compromising between performance and complexity of the model, such as Akaike's information criterion AIC, and the generalized cross-validation criterion. Analogous approaches are known in neural network research. It seems necessary that a similar criterion has to be taken into account for the NUFZY model in order to decide on the size of the fuzzy rule base. Moreover, although the NUFZY model with the OLS algorithm does give a theoretical basis for removing truly linear dependent fuzzy rules, care is needed in selecting the threshold value used to remove insignificant rules.

Second, by employing the numerical data set as guideline, the determination of centers and widths of the membership functions of input variables is very similar to a clustering problem. For simplicity, in this chapter equally spaced centers and equal widths of the membership functions have been chosen. This is significant because it eliminates a considerable number of otherwise free parameters in the antecedent part of the fuzzy rule base and allows for a very fast estimation of the remaining parameters in the consequent part, since the model is linear in these parameters. Yet, for certain applications, the modification of the antecedent parameters may be desirable, which leads to a nonlinear parameter optimization problem. This task can be facilitated by the relationship matrix $\boldsymbol{RM}$, since it offers a coding table that makes it easy to calculate the gradient of the output with respect to each parameter. When the system is expected to change in time, or when the network has to be trained on-line, the training parameters need ongoing justification. To this end, a recursive scheme of parameter adjustment can be employed. In chapter 4, examples of dynamic systems will show that the implementation of a recursive scheme is straight-forward in the NUFZY system.

Third, one of the appealing properties of fuzzy logic lies in the ability to interpret fuzzy rules linguistically. In the present NUFZY model, one hopes to deduce any interpretable fuzzy rules that describe the system's behavior from the particular fuzzy-rule-like network structure. An example of interpreting the fuzzy rule of the NUFZY model illustrated in section 3.5.1.5, shows some difficulty on direct interpretation of the identified fuzzy rule from the NUFZY model as its consequent is expressed by a crisp number, rather than a fuzzy set. In chapter 5, we will investigate the issue of interpretation of the fuzzy rules deduced from the T-S fuzzy model.

Another appealing feature of applying a fuzzy system is that the expert's knowledge can be utilized and incorporated into the framework of the fuzzy system. However, the present approach has not used this feature. The merit gained from the present NUFZY approach is that it only uses available experimental data to construct a specific model to carry out function approximation, like an ordinary feedforward neural network does. Yet, compared to the mechanistic models such as Eq.(3.12) and Eq.(3.20), the NUFZY modeling may save a lot of work on parameter calibration and model development, provided a comparable model accuracy is required.

With respect to the use of expert knowledge in fuzzy modeling, there are several possibilities to incorporate qualitative information into fuzzy models. One is by collecting the expert's knowledge and then directly aggregating it as fuzzy rules which are suitable for fuzzy modeling. In practice, this approach is not easy to implement since usually further refinement of those aggregated fuzzy rules is needed to match the modeling task. More specifically, parameters of membership functions of each fuzzy rule have to be defined by trial and error. Another approach is to incorporate qualitative information as an initialization of parameters in both the antecedent and the consequent part of fuzzy rules [78] . During the training process, the parameters are trained in order to give a good match of the model to the given data. As a result, the final rules may be quite different from original rules proposed by experts. When this happens it reveals contradictions in the consistency of qualitative information used in such an adaptive fuzzy system. If the qualitative information used in the initialization does contain the key behavior of the unknown system, this method will just facilitate the convergence of the fuzzy system; otherwise, the contradiction remains. Another aspect that seems to interfere with the ability to insert qualitative information is the observation that if the antecedent part of the fuzzy rule is fixed, like in the NUFZY model, a good model accuracy can still be achieved by merely tuning the consequent weights of the fuzzy rules [68] . It is, however, conceivable that simultaneous training of the antecedent and the consequent parameters would allow models with fewer rules which may be easier to interpret. In any case, more work has to be done to clarify the issue of utilizing qualitative information. In chapter 6, we will investigate the issue of how to incorporate *a priori* knowledge into a T-S fuzzy model.

3.7 CONCLUSIONS

In this chapter, we have demonstrated the batch learning procedure for the developed NUFZY system. Due to the pre-determination of the antecedent part of the NUFZY system, the consequent weights become the only unknown parameters to be identified. Because the model is linear in these parameters, they can be identified efficiently by an orthogonal least squares algorithm based on the classical Gram-Schmidt decomposition. Moreover, the OLS identification procedure gives a convenient way to remove the linear dependent or almost linear dependent fuzzy rules from the prototype fuzzy rule base, thus solving the redundancy problem. Some simulation results presented in this chapter show that a NUFZY model with the fast OLS training algorithms and a reduced fuzzy rule base can perform fairly well to mimic nonlinear systems.

The capability of the NUFZY model for real systems is demonstrated by a practical example involving tomato plant growth. We also apply the NUFZY model to other agricultural applications. Examples are taken from real experimental data, including a developing system, i.e., lettuce growth, and a system with a stationary operating point, i.e., greenhouse temperature. In the case of lettuce growth, the established NUFZY model offers seasonal prediction as a function of the accumulated solar radiation and actual CO_2 concentration. In contrast, the greenhouse temperature model is evaluated as a one-step-ahead prediction model. Results show that the NUFZY model can give a suitable identification of lettuce growth and greenhouse temperature.

Acknowledgement

The author gratefully acknowledges Dr. E. Heuvelink, Dr. E. J. van Henten and Ir. R. F. Tap offering the experimental data for analysis in the examples of section 3.4.2, 3.5.1, and 3.5.2, respectively.

4. RECURSIVE LEARNING OF THE NUFZY SYSTEM

Honest people use no rhetoric; Rhetoric is not honesty. Enlightened people are not cultured; Culture is not enlightenment. Content people are not rich; Riches are not contentment. So the sage does not serve himself; The more he does for others, the more he is satisfied; The more he gives, the more he receives. Nature flourishes at the expense of no one; So the sage benefits all men and contends with none.

- Lao Tze, TaoDeJing

信言不美，美言不信。善者不辯，辯者不善。知者不博，博者不知。聖人不積，既以爲人己愈有，既以與人己愈多。天之道，利而不害；聖人之道，爲而不爭。

老子道德經

4.1 INTRODUCTION†

In the identification of systems which may contain time-variant properties, on-line tuning is required to follow the varying characteristics of the system. A recursive scheme for adjusting the system parameters is considered as a potential method. The commonly used back-propagation algorithm for neural network training, where the steepest descent gradient serves as the search direction, is not suitable for recursive adaptation, because it encounters problems of slow convergence. In contrast, a recursive prediction error algorithm based on the alternative approximate Gauss-Newton search direction, was reported to have improved learning capabilities [10] . The recursive prediction error (RPE) algorithm was shown to have similar convergence properties as its off-line counter part in the case of linear systems [44] [53] . More specifically, for a chosen criterion, the estimated parameters obtained by the RPE method will converge with probability one either to a stationary point (local minimum) or get stuck at the boundary of the domain as time approaches infinity. The asymptotic convergence property of the estimates makes it attractive to adopt these ideas to multi-layered networks and to extend the application to nonlinear systems [10] . Compared to on-line identification of neural networks with back-propagation learning, the fast convergence of the RPE method is appealing in cases where the system parameters are slowly time-variant, provided the networks were previously trained well enough in an off-line way in order to provide good starting values.

Although neural networks identification has been very successful, the information representation of the internal network structure seems rather obscure, as little information can be extracted about the actual functioning of the system. On the other hand, fuzzy rule based models do have content but seem difficult to train. Like neural networks [27] [82] , fuzzy systems are also universal approximators that can approximate any real continuous function on a compact set to arbitrary accuracy [6] [80] . It has been shown in the previous chapter that the integrated neural-fuzzy system, NUFZY, based on the structural similarity and functional equivalence between fuzzy systems and neural networks, can be used to identify nonlinear systems with fairly satisfactory performance. Due to the resemblance of the NUFZY system and multi-layered neural networks, it is attractive to try to adapt the recursive prediction error algorithm to the NUFZY system. In contrast to the batch learning procedure, our goal in this chapter is to investigate the applicability of the adapted recursive prediction error algorithm for the NUFZY system as a recursive learning scheme. Based on this procedure, parameters on both the antecedent and consequent parts of the NUFZY system can be tuned in an on-line manner to achieve recursive adaptation.

When employing recursive learning we need to know the sensitivity derivatives of the NUFZY system with respect to the tuning parameters. Hence, for completeness, in section 4.2, the

† This chapter is modified from the paper [70] , titled 'Recursive prediction error algorithm for the NUFZY system to identify nonlinear systems' in *Proceedings of the 9th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems IEA/AIE-96*, Fukuoka, Japan, June 4-7, 1996, pp 569-574.

structure of the NUFZY system as developed in chapter 2 will be restated briefly and the sensitivity derivatives are given as well. The gist of the RPE algorithm and its implementation are described in section 4.3. Examples are demonstrated in section 4.4, and conclusions are drawn in section 4.5.

4.2 SENSITIVITY DERIVATIVES OF THE NUFZY SYSTEM

As shown in Figure 2.2, the developed NUFZY system is characterized by a triple-layered feedforward network. The first and second layer of NUFZY deal with the antecedent part of the fuzzy rule base and the third layer concerns the consequent part of the fuzzy rule base. The NUFZY model performs a Takagi-Sugeno (T-S) type of fuzzy inference [62] , i.e., the consequent part is formed as a linear combination of the premise variables. A variant of this T-S type of fuzzy model is that the consequent part just uses crisp real values, which is the method adopted in the NUFZY model. Given a system with ni input variables x_i, i = 1, .. , ni, and nb output variables y_n, n = 1 , .. , nb, where each x_i has its own N_i membership functions. Then the zeroth-order T-S fuzzy rules used in the NUFZY model can be expressed in the form

$$\boldsymbol{R}^r_{(TS)}: \quad \textbf{\textit{IF}}\ x_1 \text{ is } A^r_{k1}(x_1) \text{ AND .. } x_i \text{ is } A^r_{ki}(x_i) \text{ AND .. } x_{ni} \text{ is } A^r_{kni}(x_{ni})$$

$$\textbf{\textit{THEN}}\ y_1 = w_{r1}, .., y_n = w_{rn}, \ , y_{nb} = w_{r\,nb} \tag{4.1}$$

where superscript r denotes the r^{th} fuzzy rule and $A^r_{ki}(x_i)$ represents the ki^{th} linguistic label of x_i with respect to the fuzzy rule $\boldsymbol{R}^r$. It is also noted that the membership function in the consequent part is expressed in the form of a singleton value denoted by w_{rn} in the r^{th} fuzzy rule. As in chapter 2, this chapter only considers two bell shaped membership functions for the input fuzzy sets. They are the Gaussian membership function (denoted as Gau) and inverse multiquadratic membership function (denoted as IMQ). The AND connection in the antecedent part of the fuzzy rule is implemented by the algebraic product, and the centroid of gravity (COG) defuzzification is used to construct the NUFZY reasoning functions. In the following we will briefly review the node definitions of each layer and describe their corresponding sensitive derivatives. Detailed derivations can be found in Appendix B.

4.2.1 Nodes and derivatives in Layer 1 of the NUFZY system

The input node, x_i, only serves to distribute the input into the first layer nodes with fixed weights of unity. The *membership node*, denoted as $\Phi_{ki}(x_i)$, represents a membership function that performs fuzzification of the input variables. Each x_i has its own N_i linguistic labels associated with membership functions $\mu_{ki}(x_i)$. The fuzzified values $\mu_{ki}(x_i)$'s represent node outputs of this layer.

(1) Gaussian (Gau) membership function

$$\mu_{ki}(x_i) = \exp(-\frac{1}{2}\frac{(x_i - c_{i,ki})^2}{\sigma_{i,ki}^2}) \tag{4.2.a}$$

where index ki = 1, .. , N_i, i = 1, .. ,ni; $c_{i,ki}$ and $\sigma_{i,ki}$ are the ki^{th} center and bandwidth of $\Phi_{ki}(x_i)$ for the input x_i, respectively.

In contrast to chapter 2, a slight modification of the Gaussian membership function is made in order to avoid getting a membership value of zero when a new input is located outside the predefined domain of x_i. Hence, the shape of the membership function on the left and the right edges are modified to make them monotonously decreasing and increasing, respectively. These edge functions are defined as follows[‡] :

on the left edge of the domain of x_i

$$\mu_1(x_i) = \frac{1}{1 + \exp((x_i - c_{i,1}) \cdot \sigma_{i,1})} \tag{4.2.b}$$

and on the right edge of the domain of x_i

$$\mu_{Ni}(x_i) = \frac{1}{1 + \exp(-(x_i - c_{i,Ni}) \cdot \sigma_{i,Ni})} \tag{4.2.c}$$

For ease of notation, the $\mu_{ki}(x_i)$ will be reordered sequentially and is denoted by $\alpha_m(x_i)$ where subscript m runs from 1 to M. M is the total number of membership functions of all membership nodes, which is given by Eq.(2.37) as

$$M = \sum_{i=1}^{ni} N_i = N_1 + \ldots + N_{ni} \tag{4.3}$$

The transformation of $\mu_{ki}(x_i)$ to $\alpha_m(x_i)$ can be done with the following expression relating the index of m and ki,

$$m = m(i\ ,\ ki) = \sum_{f=1}^{i-1} N_f + ki \tag{4.4}$$

‡ Since with the same c and σ the IMQ membership function at edges does not go to zero as fast as the Gaussian membership function, this modification is only made on the Gaussian membership function. The disadvantage of these edge functions is that the σ effect is opposite to that of the membership function. When σ is large, the edge functions switches from 0 to 1 near the centers, or *vice versa*. When σ is small, it makes a narrow band of membership function with less overlap of each other.

where $i = 1, .., ni$; $ki = 1, .., N_i$ and when $i = 1$, $N_0 = 0$. As one can see m forms a function of i and ki, m(i, ki), which indicates a sequence index to stack all membership functions. To avoid complicated expression in following formulations, m(i, ki) will be simply denoted by m. It is easy to verify that the final index number of m is equal to M when i = ni and $k_{ni} = N_{ni}$,

$$
\begin{aligned}
m &= \left(\sum_{f=1}^{ni-1} N_f\right) + k_{ni} \\
&= (N_1 + N_2 + \cdots + N_{ni-1}) + N_{ni} \\
&= \sum_{i=1}^{ni} N_i = M
\end{aligned}
$$

Hence, with this new subscript and notation, $\alpha_m(x_i)$ can be expressed as

$$
\alpha_m(x_i) = \exp\left(-\frac{1}{2}\frac{(x_i - c_m)^2}{\sigma_m^2}\right) \tag{4.2.d}
$$

The center c_m and bandwidth σ_m corresponding to $c_{i,ki}$ and $\sigma_{i,ki}$ can be obtained in the same manner as above.

(2) Inverse multiquadratic (IMQ) membership function

$$
\mu_{ki}(x_i) = \frac{1}{\sqrt{(x_i - c_{i,ki})^2 + \sigma_{i,ki}^2}} \tag{4.5.a}
$$

or

$$
\alpha_m(x_i) = \frac{1}{\sqrt{(x_i - c_m)^2 + \sigma_m^2}} \tag{4.5.b}
$$

Hence, for a specific input vector $\boldsymbol{x} = [x_1\ x_2\ ..x_i...x_{ni}]^T$, the corresponding membership values can be denoted in a vector form, $\alpha = [\alpha_1\ \ \alpha_2\ ...\ \alpha_m\ ...\ \alpha_M]^T$, which only stacks the outputs of the membership functions for all inputs. As an example, suppose we had three inputs, x_1, x_2, x_3, having 2, 3 and 2 membership functions, respectively. The membership values, for input x_1 are $\mu_1(x_1)$ and $\mu_2(x_1)$; for input x_2 are $\mu_1(x_2)$, $\mu_2(x_2)$, and $\mu_3(x_2)$; for input x_3 are $\mu_1(x_3)$ and $\mu_2(x_3)$. They are denoted by μ_{11}, μ_{12}, μ_{21}, μ_{22}, μ_{23}, μ_{31} and μ_{32}, respectively, below. In this case, $M = 2 + 3 + 2 = 7$ and $\alpha = [\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \alpha_7]^T = [\mu_{11}\ \mu_{12}\ \mu_{21}\ \mu_{22}\ \mu_{23}\ \mu_{31}\ \mu_{32}]^T$.

The node parameters to be determined in this layer are $c_{i,ki}$ and $\sigma_{i,ki}$ (or, c_m and σ_m). Initially, the centers are chosen as equally spaced on the range of x_i (from a set of training data) and the values of the variance of x_i are taken as bandwidths. By the above definitions, it is easy to deduce the derivatives of the outputs of membership nodes with respect to parameters, i.e. $\partial\mu_{ki}(x_i)/\partial c_{i,ki}$ and $\partial\mu_{ki}(x_i)/\partial\sigma_{i,ki}$ (or $\partial\alpha_m(x_i)/\partial c_m$ and $\partial\alpha_m(x_i)/\partial\sigma_m$); see Appendix B.

4.2.2 Nodes and derivatives in Layer 2 of the NUFZY system

In this layer, the *rule node*, denoted as $\mathfrak{R}_r$, represents a fuzzy rule, where subscript r = 1, .. , R and R is the total number of all fuzzy rules given by Eq.(2.39),

$$R = \prod_{i=1}^{ni} N_i \tag{4.6}$$

The existence of a connection between a *rule node* and a *membership node* is represented by a value of either 1 or 0. They are recorded in a R×M relationship matrix ***RM***, which is defined in section 2.3.1. Each *rule node* performs a two-step operation,

Step one : the transient firing strength ν_r is obtained by

$$\nu_r = \prod_{i=1}^{ni} \boldsymbol{RM}(\mathrm{r}, a_i{:}b_i)\cdot\mu_i \tag{4.7}$$

where notations of $\boldsymbol{RM}(\mathrm{r}, a_i{:}b_i)$ and μ_i are defined in the same way as those in section 2.3.1.

Step two: the normalized firing strength $\bar{\nu}_r$ is calculated as

$$\bar{\nu}_r = \frac{\nu_r}{\sum_{p=1}^{R} \nu_p} \tag{4.8}$$

In this layer, there is no parameter to be determined. Yet, due to the chain rule used for the sensitivity derivatives of the NUFZY system, we need the partial derivative of $\bar{\nu}_r$ with respect to α_m, which forms a R by M Jacobian matrix; see Appendix B.

4.2.3 Nodes and derivatives in Layer 3 of the NUFZY system

The node, $\hat{y}_n$, n = 1, .. , nb, stands for the NUFZY model output. The link in this layer represents a weight parameter w_{rn}, for r = 1, .. , R, n = 1, .. , nb, and connects nodes $\hat{y}_n$ and $\mathfrak{R}_r$. Using the centroid of gravity defuzzification method, the model output is obtained by

$$\hat{y}_n = \sum_{r=1}^{R} w_{rn}\bar{\nu}_r = \mathbf{w}_n^T\bar{\mathbf{v}} \tag{4.9}$$

where $\mathbf{w}_n$ is the consequent weight parameter vector given by $\mathbf{w}_n = [w_{1n} .. w_{rn} .. w_{Rn}]^T$ and $\bar{\mathbf{v}}$ is a normalized firing strength vector given by $\bar{\mathbf{v}} = [\bar{v}_1 .. \bar{v}_r .. \bar{v}_R]^T$ with element $\bar{v}_r$ defined by Eq.(4.8). The derivatives of the NUFZY output with respect to weight parameter and normalized firing strength can be found in Appendix B.

4.2.4 Sensitivity derivatives of the NUFZY system

From the above derivation, we can define the parameter set θ of the NUFZY system that needs to be tuned as either $\theta = \varpi$, or $\theta = [\varpi\ \mathbf{c}]^T$, or $\theta = [\varpi\ \mathbf{c}\ \sigma]^T$, where parameter vector θ just stacks all the tuning parameter vector of $\varpi = [\mathbf{w}_1^T .. \mathbf{w}_n^T .. \mathbf{w}_{nb}^T]^T$, with $\mathbf{w}_n = [w_{1n} .. w_{rn} .. w_{Rn}]^T$, and $\mathbf{c} = [c_1 .. c_m .. c_M]^T$ as well as $\sigma = [\sigma_1 .. \sigma_m .. \sigma_M]^T$. Hence, the dimension of θ can be nb×R, or nb×R+M, or nb×R+2M where M is defined in Eq.(4.3). Hereafter, we will use d to denote the dimension of θ, i.e., d = dim(θ). In this subsection, the results will be outlined. Details of derivation and matrix notations used are given to in Appendix B.

(1) Sensitivity derivative of the NUFZY system output with respect to $\mathbf{w}$

In this case, the parameter set θ is defined as $\theta_\varpi = \varpi = [\mathbf{w}_1^T .. \mathbf{w}_n^T .. \mathbf{w}_{nb}^T]^T$ (a (nb·R)×1 vector). For single output $\hat{y}_n$ (i.e., nb = 1), the partial derivative of $\hat{y}_n$ with respect to θ_ϖ, i.e., the sensitivity derivative $\Psi_{\varpi_n} = \left[\frac{\partial \hat{y}_n}{\partial \theta_\varpi}\right]$, becomes a R×1 vector

$$\Psi_{\varpi_n} = \left[\frac{\partial \hat{y}_n}{\partial \theta_\varpi}\right]$$

$$= \left[\left[\frac{\partial(\mathbf{w}_n^T\bar{\mathbf{v}})}{\partial \mathbf{w}_1}\right]^T \ \cdots \ \left[\frac{\partial(\mathbf{w}_n^T\bar{\mathbf{v}})}{\partial \mathbf{w}_n}\right]^T \ \cdots \ \left[\frac{\partial(\mathbf{w}_n^T\bar{\mathbf{v}})}{\partial \mathbf{w}_{nb}}\right]^T\right]^T$$

$$= [\mathbf{0}^T .. \bar{\mathbf{v}}^T .. \mathbf{0}^T]^T = \begin{bmatrix} \mathbf{0} \\ \vdots \\ \bar{\mathbf{v}} \\ \vdots \\ \mathbf{0} \end{bmatrix}$$

where $\mathbf{0}$ is a R by 1 zero vector.

For the multi-output case, the sensitivity derivative $\Psi_\varpi = \left[\frac{\partial \hat{\mathbf{y}}}{\partial \theta_\varpi}\right]$, becomes

$$\Psi_{\varpi} = \frac{\partial \hat{y}}{\partial \theta_{\varpi}} = \left[\left[\frac{\partial \hat{y}_1}{\partial \theta_{\varpi}} \right] \cdots \left[\frac{\partial \hat{y}_n}{\partial \theta_{\varpi}} \right] \cdots \left[\frac{\partial \hat{y}_{nb}}{\partial \theta_{\varpi}} \right] \right]$$

$$= \begin{bmatrix} \overline{\mathbf{v}} & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \vdots & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \overline{\mathbf{v}} & \cdots & \vdots \\ \vdots & \cdots & \vdots & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \cdots & \overline{\mathbf{v}} \end{bmatrix}_{((nb \cdot R) \times nb)}$$

(2) Sensitivity derivative of the NUFZY system output with respect to **c**

Let $\boldsymbol{\theta}_c = \mathbf{c} = [c_1 \,..\, c_m \,..\, c_M]^T$ (or , $\boldsymbol{\theta}_c = [c_{11} \,..\, c_{i\,ki} \,..\, c_{ni\,Nni}]^T$, M×1 vector). Then, an element of the sensitivity derivative of Ψ_c (= $\left[\frac{\partial \hat{y}}{\partial \theta_c} \right]$, M×nb matrix), $\Psi_c(n, m)$ can be obtained as

$$\Psi_c(n,m) = \frac{\partial \hat{y}_n}{\partial c_m} = \hat{\Sigma}_{r=1}^{R} \frac{\partial \hat{y}_n}{\partial \overline{v}_r} \frac{\partial \overline{v}_r}{\partial v_r} \frac{\partial v_r}{\partial \alpha_m} \frac{\partial \alpha_m}{\partial c_m}$$

$$= \boldsymbol{RM}(:,m)^T * \left(\begin{bmatrix} \frac{\partial \hat{y}_n}{\partial \overline{v}_1} \\ \vdots \\ \frac{\partial \hat{y}_n}{\partial \overline{v}_r} \\ \vdots \\ \frac{\partial \hat{y}_n}{\partial \overline{v}_R} \end{bmatrix} .* \begin{bmatrix} \frac{\partial \overline{v}_1}{\partial v_1} \\ \vdots \\ \frac{\partial \overline{v}_r}{\partial v_r} \\ \vdots \\ \frac{\partial \overline{v}_R}{\partial v_R} \end{bmatrix} .* \begin{bmatrix} \frac{\partial v_1}{\partial \alpha_m} \\ \vdots \\ \frac{\partial v_r}{\partial \alpha_m} \\ \vdots \\ \frac{\partial v_R}{\partial \alpha_m} \end{bmatrix} .* \begin{bmatrix} \frac{\partial \alpha_m}{\partial c_m} \\ \vdots \\ \frac{\partial \alpha_m}{\partial c_m} \\ \vdots \\ \frac{\partial \alpha_m}{\partial c_m} \end{bmatrix} \right)$$

(3) Sensitivity derivative of the NUFZY system output with respect to **σ**

Let $\boldsymbol{\theta}_\sigma = \boldsymbol{\sigma} = [\sigma_1 \,..\, \sigma_m \,..\, \sigma_M]^T$ (or $\boldsymbol{\theta}_\sigma = [\sigma_{11} \,..\, \sigma_{i\,ki} \,..\, \sigma_{ni\,Nni}]^T$, M×1 vector). Then, an element of the sensitivity derivative of Ψ_σ (= $\left[\frac{\partial \hat{y}}{\partial \theta_\sigma} \right]$, M×nb matrix), $\Psi_\sigma(n, m)$ can be obtained as

$$\Psi_\sigma(n,m) = \frac{\partial \hat{y}_n}{\partial \sigma_m} = \hat{\Sigma}_{r=1}^{R} \frac{\partial \hat{y}_n}{\partial \overline{v}_r} \frac{\partial \overline{v}_r}{\partial v_r} \frac{\partial v_r}{\partial \alpha_m} \frac{\partial \alpha_m}{\partial \sigma_m}$$

$$= \boldsymbol{RM}(:,m)^T * \left(\begin{bmatrix} \frac{\partial \hat{y}_n}{\partial \overline{v}_1} \\ \vdots \\ \frac{\partial \hat{y}_n}{\partial \overline{v}_r} \\ \vdots \\ \frac{\partial \hat{y}_n}{\partial \overline{v}_R} \end{bmatrix} .* \begin{bmatrix} \frac{\partial \overline{v}_1}{\partial v_1} \\ \vdots \\ \frac{\partial \overline{v}_r}{\partial v_r} \\ \vdots \\ \frac{\partial \overline{v}_R}{\partial v_R} \end{bmatrix} .* \begin{bmatrix} \frac{\partial v_1}{\partial \alpha_m} \\ \vdots \\ \frac{\partial v_r}{\partial \alpha_m} \\ \vdots \\ \frac{\partial v_R}{\partial \alpha_m} \end{bmatrix} .* \begin{bmatrix} \frac{\partial \alpha_m}{\partial \sigma_m} \\ \vdots \\ \frac{\partial \alpha_m}{\partial \sigma_m} \\ \vdots \\ \frac{\partial \alpha_m}{\partial \sigma_m} \end{bmatrix} \right)$$

4.3 PREDICTION ERROR ALGORITHM

Recursive estimation by the prediction error method has been studied systematically [43] [44] [84] . In this section we give results based on [44] , from two aspects, off-line and on-line identification, which correspond to batch and recursive learning, respectively.

4.3.1 Batch prediction error learning

Parameter estimation methods need some objective criteria to measure the fitness between the real system output y(t) and the model predicted output $\hat{y}(t)$, where dim(y(t)) = nb. For a time-invariant system, given an available set of data $\{x(t)\ y(t)\}_{t=1,\dots,np}$, a good choice of the objective criterion is a quadratic form of the prediction error weighted by its covariance matrix,

$$V_{np}(\theta) = \frac{1}{2}\sum_{t=1}^{np} \varepsilon^T(t,\theta)\Lambda^{-1}\varepsilon(t,\theta) \tag{4.10}$$

where $\varepsilon(t,\theta) = y(t) - \hat{y}(t)$, a nb by 1 column vector used to evaluate the search direction, is the discrepancy between the real system output and model predicted output. Matrix Λ is a d by d covariance matrix of the prediction error, which is evaluated based on the true parameter θ_0 of the system. Minimizing the above criterion, the estimated parameter θ has minimal variance, and asymptotically converges to the true parameter θ_0 when the sample number np goes to infinity. Since a set of sampled data $\{x(t)\ y(t)\}_{t=1,\dots,np}$ is available, θ can be updated iteratively by estimation based on the Robbins-Monro stochastic approximation method [44] shown in the next equation, using all np observed samples until the minimum of Eq.(4.10) is reached.

$$\hat{\theta}^{(i+1)} = \hat{\theta}^{(i)} + \gamma^{(i)}\Xi(\hat{\theta}^{(i)}) \tag{4.11}$$

where the superscript (i) denotes the iteration step in the minimization procedure; γ is a positive gain, which tends to zero and modifies the step size of update as well as influences the convergence of the iteration; $\Xi(\hat{\theta}^{(i)})$ is a search direction based on information about $V_{np}(\theta)$ acquired in the previous iteration. The search direction $\Xi(\hat{\theta}^{(i)})$, determined by the negative gradient of $V_{np}(\theta)$ with respect to θ, can be further expressed together with a search direction modification matrix, $M(\hat{\theta}^{(i)})$

$$\Xi(\hat{\theta}^{(i)}) = M(\hat{\theta}^{(i)})(-V'_{np}(\hat{\theta}^{(i)})) \tag{4.12}$$

where $V'_{np}(\hat{\theta}^{(i)})$, a d×1 column vector, is the gradient of V_{np} with respect to $\boldsymbol{\theta}$. If we define the sensitivity derivative as

$$\Psi(t,\hat{\theta}^{(i)}) = \left.\frac{\partial \hat{y}}{\partial \theta}\right|_{\theta=\hat{\theta}^{(i)}} \quad (= -\left.\frac{\partial \varepsilon}{\partial \theta}\right|_{\theta=\hat{\theta}^{(i)}}) \tag{4.13}$$

then $V'_{np}(\hat{\theta}^{(i)})$ can be written as

$$V'_{np}(\hat{\theta}^{(i)}) = -\sum_{t=1}^{np} \Psi(t,\hat{\theta}^{(i)})\Lambda^{-1}\varepsilon(t,\hat{\theta}^{(i)}) \tag{4.14}$$

Hence, Eq.(4.11) becomes

$$\hat{\theta}^{(i+1)} = \hat{\theta}^{(i)} + \gamma^{(i)}\mathbf{M}(\hat{\theta}^{(i)}) \cdot \left[\sum_{t=1}^{np} \Psi(t,\hat{\theta}^{(i)})\Lambda^{-1}\varepsilon(t,\hat{\theta}^{(i)})\right] \tag{4.15}$$

It is interesting to note that different learning algorithms can be derived by choosing different representations of the matrix $\mathbf{M}(\theta)$ [52] . For instance,

(1) *Gradient* direction, simply take

$$\mathbf{M}(\theta) = \mathbf{I} \tag{4.16}$$

The parameter updating method based on the gradient direction is referred to as the gradient algorithm or steepest-descent algorithm. It is fast in the initial stage but tends to have very slow convergence near the optimum. The commonly used back-propagation learning in neural networks is one example employing this gradient direction.

(2) *Gaussian - Newton* direction, use the inverse of Hessian matrix, $\mathbf{H}(\theta)$, of the system as $\mathbf{M}(\theta)$, i.e.,

$$\mathbf{M}(\theta) = \mathbf{H}^{-1}(\theta) \tag{4.17}$$

where the Hessian matrix $\mathbf{H}(\theta)$ is

$$\mathbf{H}(\theta) = \left[\sum_{t=1}^{np} \Psi(t,\theta)\Lambda^{-1}\Psi^T(t,\theta) + \sum_{t=1}^{np} \frac{\partial^2 \hat{y}}{\partial \theta^2}\Lambda^{-1}\varepsilon(t,\theta)\right] \tag{4.18}$$

It is notice that the second derivative term in above equation is in fact a tensor, which makes the estimation of $\mathbf{H}(\theta)$ in the 'true' Gaussian-Newton direction more complex. This complex second derivative in the estimation of $\mathbf{H}(\theta)$ is, however, often not taken into account. Hence, for simplicity, a modified Gaussian-Newton search direction, using an approximation of the Hessian matrix, also called Fisher information matrix $\mathbf{R}(\theta)$, can be an alternative neglecting the second derivative of Eq.(4.18). This results in the following:

(3) *modified Gaussian - Newton* direction,

$$\mathbf{M}(\theta) = \mathbf{R}^{-1}(\theta) \quad \text{and} \quad \mathbf{R}(\theta) = \left[\sum_{t=1}^{np} \Psi(t,\theta)\Lambda^{-1}\Psi^{T}(t,\theta)\right] \cong \mathbf{H}(\theta) \tag{4.19}$$

The parameter updating method based on this modified Gaussian-Newton direction is also termed Quasi-Newton algorithm, which has better convergence performance than the steepest-descent algorithm at the expense of increased complexity. It is noted that, in addition to the diagonal block elements, matrix $\mathbf{R}(\theta)$ has other elements which exist over the entire matrix. A more simplified version of matrix $\mathbf{R}(\theta)$ can be made by merely making use of the diagonal block elements of $\mathbf{R}(\theta)$ and let the off-diagonal block elements be zero. This is termed parallel prediction error algorithm by [7] [10] . In contrast to the 'full' $\mathbf{R}(\theta)$ matrix, this simplified matrix $\mathbf{R}(\theta)$ increases the computational efficiency and is a good compromise as compared to even further simplifications as, e.g., Eq.(4.16).

(4) *Levenberg - Marquardt* direction,

$$\mathbf{M}(\theta) = \mathbf{H}^{-1}(\theta) \quad \text{and} \quad \mathbf{H}(\theta) = \left[\sum_{t=1}^{np} \Psi(t,\theta)\Lambda^{-1}\Psi^{T}(t,\theta) + \delta \mathbf{I}\right] \tag{4.20}$$

where δ is a small positive value and the identity matrix is with appropriate dimension. The term $\delta\mathbf{I}$ is introduced to avoid singularity of the Hessian matrix.

The iteration procedure starts with initial estimates of the unknown parameters, θ_{ini}, and updates the parameter according to Eq.(4.15) based on Eq.(4.13) and Eq.(4.16) - Eq.(4.20), using all the observed samples until the minimization of V_{np} is reached. Since this iteration is done off-line, it can be regarded as a batch prediction error learning algorithm. We will denote these off-line estimates of the parameter as $\hat{\theta}_{np}$.

4.3.2 Recursive prediction error learning

In case we deal with a time-variant system, or applications of on-line identification, the above process of updating the parameter must be recursive. The consequence is that the recursive methods cannot be expected to determine the off-line estimates $\hat{\theta}_{np}$. Instead, one has to be content with recursive approximations to the $\hat{\theta}_{np}$. In the following derivation, we will focus on the modified Gauss-Newton direction where the search direction $\mathbf{M}(\theta)$ is represented by the inverse of the approximated Hessian, $\mathbf{R}^{-1}(\theta)$, as defined in Eq.(4.19).

Now, consider that t sampled data (k = 1, .. , t) are available, the objective function then becomes

$$V_t(\theta) = \frac{1}{2}\sum_{k=1}^{t} \varepsilon^T(k,\theta)\Lambda^{-1}\varepsilon(k,\theta) \tag{4.21}$$

Let $\hat{\theta}(t-1)$ be the estimate at time t-1. Our goal is to find an estimate $\hat{\theta}(t)$ that can minimize the objective function $V_t(\theta)$ based on the previous estimation $\hat{\theta}(t-1)$. This means that the approximations of $\mathbf{M}(\theta)$ and $V'_t(\theta)$ (the gradient of $V_t(\theta)$) in the next update equation, similar to Eq.(4.15), are also both evaluated based on the estimate $\hat{\theta}(t-1)$ at time t-1.

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \gamma(t)\cdot\left[\overline{V}''_t(\hat{\theta}(t-1))\right]^{-1}\cdot\left[-V'_t(\hat{\theta}(t-1))\right] \tag{4.22}$$

where matrix $\mathbf{M}(\hat{\theta}(t-1))$ is replaced by $\left[\overline{V}''_t(\hat{\theta}(t-1))\right]^{-1}$ and $\overline{V}''_t(\hat{\theta}(t-1))$ means an approximation of the second derivative of $V_t(\theta)$ based on observations up to time t. If we denote the approximation of second derivative $\overline{V}''_t(\hat{\theta}(t-1))$ by $\mathbf{R}(t)$, then the update equation is written as

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \gamma(t)\cdot\mathbf{R}^{-1}(t)\cdot\left[-V'_t(\hat{\theta}(t-1))\right] \tag{4.23}$$

From Eq.(4.21), the derivative of $V_t(\theta)$ with respect to θ, $V'_t(\theta)$, can be obtained.

$$V_t'(\theta) = \sum_{k=1}^{t}\frac{\partial\varepsilon(k,\theta)}{\partial\theta}\Lambda^{-1}\varepsilon(k,\theta) \tag{4.24}$$

Similar to Eq.(4.13), if the sensitivity derivative (a d×nb Jacobian matrix) is defined as

$$\Psi(t,\theta) = \left[\frac{\partial\hat{y}}{\partial\theta}\right] \quad \left(= -\left[\frac{\partial\varepsilon}{\partial\theta}\right]\right) \tag{4.25}$$

then $V'_t(\theta)$ can be represented recursively,

$$
\begin{aligned}
V'_t(\theta) &= \sum_{k=1}^{t} \frac{\partial \varepsilon(k,\theta)}{\partial \theta} \Lambda^{-1} \varepsilon(k,\theta) = -\sum_{k=1}^{t} \Psi(k,\theta)\Lambda^{-1}\varepsilon(k,\theta) \\
&= -\sum_{k=1}^{t-1} \Psi(t,\theta)\Lambda^{-1}\varepsilon(k,\theta) - \Psi(t,\theta)\Lambda^{-1}\varepsilon(t,\theta) \\
&= V'_{t-1}(\theta) - \Psi(t,\theta)\Lambda^{-1}\varepsilon(t,\theta)
\end{aligned}
\tag{4.26}
$$

In order to evaluate Eq.(4.23), we have to introduce several approximations. First we assume that the next estimate $\hat{\theta}(t)$ is to be found in the vicinity of $\hat{\theta}(t-1)$. This assumption is reasonable if t is large. Then we assume that estimate $\hat{\theta}(t-1)$ is indeed the optimal estimate at time t - 1, such that

$$
V'_{t-1}(\hat{\theta}(t-1)) = 0 \tag{4.27}
$$

Hence, according to the above assumptions and Eq.(4.26), the gradient $V'_t(\hat{\theta}(t-1))$ becomes

$$
\begin{aligned}
V'_t(\hat{\theta}(t-1)) &= V'_{t-1}(\hat{\theta}(t-1)) - \Psi(t,\hat{\theta}(t-1))\Lambda^{-1}\varepsilon(t,\hat{\theta}(t-1)) \\
&= -\Psi(t,\hat{\theta}(t-1))\Lambda^{-1}\varepsilon(t,\hat{\theta}(t-1)) \\
&= -\Psi(t)\Lambda^{-1}\varepsilon(t)
\end{aligned}
\tag{4.28}
$$

where $\Psi(t, \hat{\theta}(t-1))$ and $\varepsilon(t, \hat{\theta}(t-1))$ are denoted by $\Psi(t)$ and $\varepsilon(t)$, respectively, for short.

The approximation of the Hessian matrix $\mathbf{R}(t)$ at $\theta = \hat{\theta}(t-1)$ based on t-1 observations can be expressed in a recursive manner together with the gain factor, $\gamma(t)$. I.e.,

$$
\mathbf{R}(t) = \mathbf{R}(t-1) + \gamma(t)[\Psi(t)\Lambda^{-1}\Psi^T(t) - \mathbf{R}(t-1)] \quad \text{with initial } \mathbf{R}(0) = \mathbf{R}_0 \tag{4.29}
$$

As mentioned above, the covariance matrix Λ based on the true parameter θ_0 is the optimal choice for weighting in the objection function $V_t(\theta)$. However, this optimal covariance is typically unknown because the true θ_0 cannot be obtained. Consequently, a reasonable approximation of the covariance, $\hat{\Lambda}(t)$, has to be estimated recursively in a similar fashion as in Eq.(4.29),

$$
\hat{\Lambda}(t) = \hat{\Lambda}(t-1) + \gamma(t)[\varepsilon(t)\varepsilon(t)^T - \hat{\Lambda}(t-1)] \tag{4.30}
$$

In order to avoid the inverse of $\mathbf{R}$ in Eq.(4.23), a more convenient algorithm can be obtained from the recursive form of Eq.(4.29) by applying the matrix inverse lemma, i.e., if we introduce

$$\mathbf{P}(t) = \gamma(t)\,\mathbf{R}^{-1}(t) \tag{4.31}$$

as an approximation of the inverse of the Hessian matrix. Applying the matrix inverse lemma, we obtain the recursive expression of $\mathbf{P}(t)$, together with the 'forgetting factor', $\lambda(t)$, as below.

$$\mathbf{P}(t) = \frac{1}{\lambda(t)}\{\mathbf{P}(t-1) - \mathbf{P}(t-1)\Psi(t)\,[\lambda(t)\hat{\Lambda}(t) + \Psi^T(t)\mathbf{P}(t-1)\Psi(t)]^{-1}\,\Psi^T(t)\mathbf{P}(t-1)\} \tag{4.32}$$

Moreover, using this expression for $\mathbf{P}(t)$, we can further write,

$$\begin{aligned} \mathbf{L}(t) &= \gamma(t)\mathbf{R}^{-1}(t)\Psi(t)\hat{\Lambda}^{-1}(t) \\ &= \mathbf{P}(t)\Psi(t)\hat{\Lambda}^{-1}(t) \\ &= \mathbf{P}(t-1)\Psi(t)[\lambda(t)\hat{\Lambda}(t) + \Psi^T(t)\mathbf{P}(t-1)\Psi(t)]^{-1} \end{aligned} \tag{4.33}$$

Hence, the recursive prediction error algorithm based on the above derivation is summarized below [44] ,

$$\begin{aligned} \varepsilon(t) &= \mathbf{y}(t) - \hat{\mathbf{y}}(t) \\ \hat{\Lambda}(t) &= \hat{\Lambda}(t-1) + \gamma(t)[\varepsilon(t)\varepsilon(t)^T - \hat{\Lambda}(t)] \\ \mathbf{S}(t) &= \lambda(t)\hat{\Lambda}(t) + \Psi^T(t)\mathbf{P}(t-1)\Psi(t) \\ \mathbf{L}(t) &= \mathbf{P}(t-1)\Psi(t)\mathbf{S}^{-1}(t) \\ \hat{\theta}(t) &= \hat{\theta}(t-1) + \mathbf{L}(t)\varepsilon(t) \\ \mathbf{P}(t) &= \frac{1}{\lambda(t)}[\mathbf{P}(t-1) - \mathbf{L}(t)\mathbf{S}(t)^{-1}\mathbf{L}^T(t)] \end{aligned} \tag{4.34}$$

When employing the recursive prediction error method, one must be aware that the purpose of real-time identification is to track time-varying parameters. However, in the presence of noise, it is impossible to accurately follow parameters that change too fast. Obviously, a tradeoff exists between tracking ability and noise sensitivity and only slow time variation of the parameters can be achieved by recursive identification. If it is known beforehand which parameters are time-variant, it has been suggested [44] that the forgetting factor, $\lambda(t)$, in Eq.(4.34) could be generally chosen as a constant smaller than 1, and the gain sequence, $\gamma(t)$, is formed as a suitable function of λ and t. In this case $\gamma(t)$ decreases to zero as t goes to infinite. Usually, however, the variation property of parameters in a system is not exactly known to us. In such a circumstance, we could start to treat it in a time-invariant or slowly varying manner. In the following examples where the time variability is to be studied, we have assumed that the systems have slow varying dynamics, and this is particularly obvious in the example of plant growth in agriculture.

Some initial values of the recursive prediction error learning are set as below. The initial value of the estimated covariance matrix of prediction error, $\hat{\Lambda}(0)$, is set as $0.1\times\mathbf{I}$ (nb by nb identify matrix). Matrix $\mathbf{P}(0)$ is initialized as a d by d diagonal matrix with a diagonal element of 10000; where d = dim (θ), the dimension of θ depends on the choice of the parameter set. It is desirable to set the forgetting factor $\lambda(t) < 1$ at the initial stage in order to achieve rapid adaptation and then let $\lambda(t) \rightarrow 1$ as $t \rightarrow \infty$. Hence the forgetting factor, $\lambda(t)$, and the gain sequence, $\gamma(t)$, are chosen as [44]

$$\lambda(t) = \lambda_0\, \lambda(t\text{-}1) + (1\text{-}\lambda_0) \tag{4.35}$$

$$\gamma(t) = 1 - \frac{\lambda(t)}{\lambda(t) + \gamma(t-1)} \tag{4.36}$$

with initializations of $\lambda_0 = 0.99$ and $\lambda(0) = 0.95$.

4.4 *EXAMPLES*

In this section, two examples are presented to demonstrate the implementation of the recursive prediction error method in the NUFZY system for identification of nonlinear MISO systems. The parameter sets of the NUFZY system that are desired to be tuned can be defined as either $\theta = \boldsymbol{\varpi}$, or $\theta = [\boldsymbol{\varpi}\ \mathbf{c}]^T$, or $\theta = [\boldsymbol{\varpi}\ \mathbf{c}\ \boldsymbol{\sigma}]^T$. Among these options, our previous studies showed that good approximation could be achieved by the mere tuning of consequent weights (i.e. $\theta = \mathbf{w}$, when only one output variable is considered) [68] . Hence, the next two examples presented here are based on using the recursive prediction error algorithm to tune the consequent weight parameters only. In the initialization of $\theta(0)(= \mathbf{w}_{ini})$, we compare two approaches. One uses the consequent weights $\mathbf{w}_{ols}$, that are first identified by orthogonal least squares method with a batch of training data set, as $\mathbf{w}_{ini}$. The other uses zeros as $\mathbf{w}_{ini}$, i.e., all parameter values starting from zero.

4.4.1 Example 1 - synthetic nonlinear system

This example is equivalent to example 4 of [48] . The dynamical system is given as

$$y(\mathrm{k}+1) = \frac{x_1 x_2 x_3 x_5 (x_3 - 1) + x_4}{1 + x_2^2 + x_3^2} \tag{4.37}$$

where $[x_1, x_2, x_3, x_4, x_5] = [y(\mathrm{k}), y(\mathrm{k}\text{-}1), y(\mathrm{k}\text{-}2), u(\mathrm{k}), u(\mathrm{k}\text{-}1)]$. In this example, the NUFZY net is employed to generate a one-step-ahead prediction $\hat{y}(\mathrm{k}+1)$. First, 500 training data points are generated by the plant Eq.(4.37) with a random input signal uniformly distributed in the intervals of [-1,1]. These training data are used to train the NUFZY model with the orthogonal least squares method in order to get a trained output weight, denoted as $\mathbf{w}_{ols}$. These weights $\mathbf{w}_{ols}$ can be regarded as representing the optimal parameter values from this batch of training data. The number of membership functions is assigned as two to each input variable. Hence, the total number of fuzzy rules will be 32 initially. After the orthogonal least squares training, it is found that only 26 rules are significant. These weights are then used as initialization of the RPE method for the subsequent validation step (i.e., $\mathbf{w}_{ini} = \mathbf{w}_{ols}$) where 1000 pairs of data are generated according to Eq.(4.37), based on an input sequence $u(\mathrm{k})$ given by

$$u(\mathrm{k}) = \begin{cases} \sin(\frac{2\pi \mathrm{k}}{250}) & 0 \le \mathrm{k} \le 500 \\ 0.8\sin(\frac{2\pi \mathrm{k}}{250}) + 0.2\sin(\frac{2\pi \mathrm{k}}{25}) & 500 < \mathrm{k} \le 1000 \end{cases} \tag{4.38}$$

Figure 4.1 shows the result of the validation. It can be seen that the NUFZY model gives excellent prediction. Figure 4.2 presents the variation of the consequent weights of the NUFZY model tuned by the RPE method during validation.

On the other hand, if validation is initialized with initial weights set to zero (i.e. $\mathbf{w}_{ini} = \mathbf{0}$, a d by 1 column vector with zeros, rather than $\mathbf{w}_{ols}$), a similar result (not shown) as Figure 4.1 by RPE tuning is obtained but with a little less accuracy than the previous one. However, the variation of the tuned weights is quite different from that shown in Figure 4.3. At the 500^{th} time step, the weights are clearly readjusted as input signals with different frequency come in. This implies that the RPE initialized with **0** was trapped on some local minimum and the tuning proceeds only locally. When the frequency of the input signals changes, the RPE readjusts these weights to another (local) minimum in order to get a good fit. This example demonstrates that it pays to initialize the RPE tuning with parameters obtained off-line from a good excitation signal. However, if not, the NUFZY model still fits the system well on that local excitation signal.

In [48] a parallel model (i.e. the past model predictions are components of the input vector) was identified which requires 100,000 steps of training. In order to compare with the aforementioned simulation, this parallel approach is also adopted for the NUFZY model. As a result, a very good prediction is obtained as well (see [68]) and the NUFZY model accuracy is far superior than that of an artificial neural network trained by back-propagation used in [48] . In addition, the NUFZY model requires less training efforts and has a lower model complexity. For the training phase, 500 samples are used in one step using the OLS identification in contrast to the 100,000 steps of back-propagation adaptation used in [48] . With respect to the model complexity, only 32 weights have to be adjusted in the NUFZY networks while 320 weights were used by them. The key is that good performance can be achieved by just tuning the output weights of the NUFZY model. This makes it very appealing for fast identification since the problem then is linear-in-the-parameters.

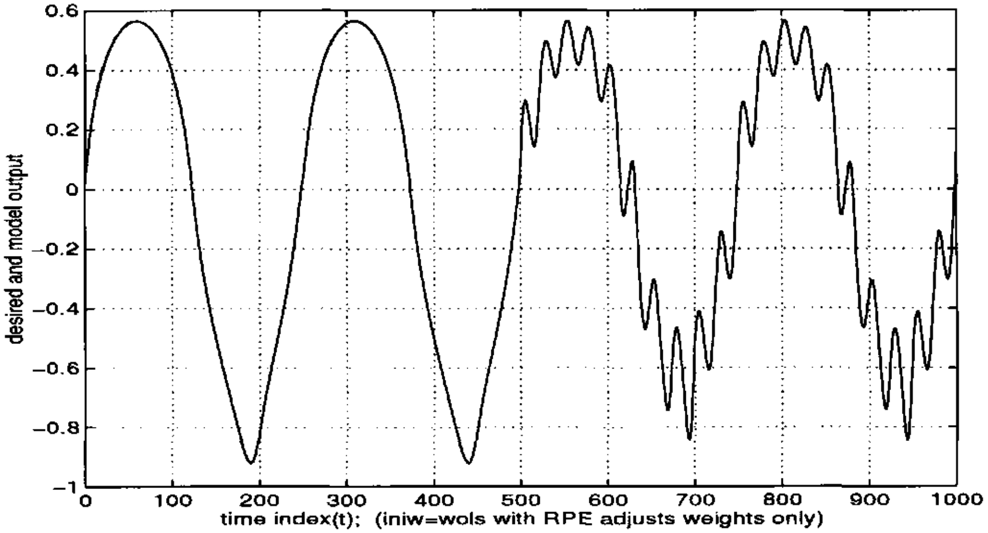

Figure 4.1: The predicted output of the NUFZY model superimposed on the desired output in a time-invariant case. Solid line - the desired output ; dashed line - the predicted output of the NUFZY model. Note, they are hardly distinguishable.

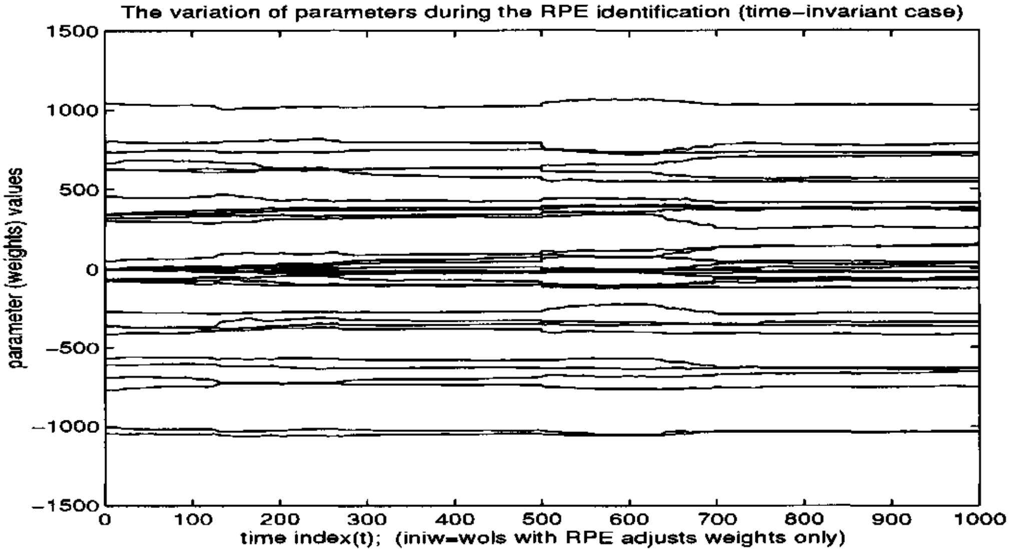

Figure 4.2: The variation of the identified weight of the NUFZY model during the validation in the time-invariant case, when initial weighting values are set to $\mathbf{w}_{ols}$.

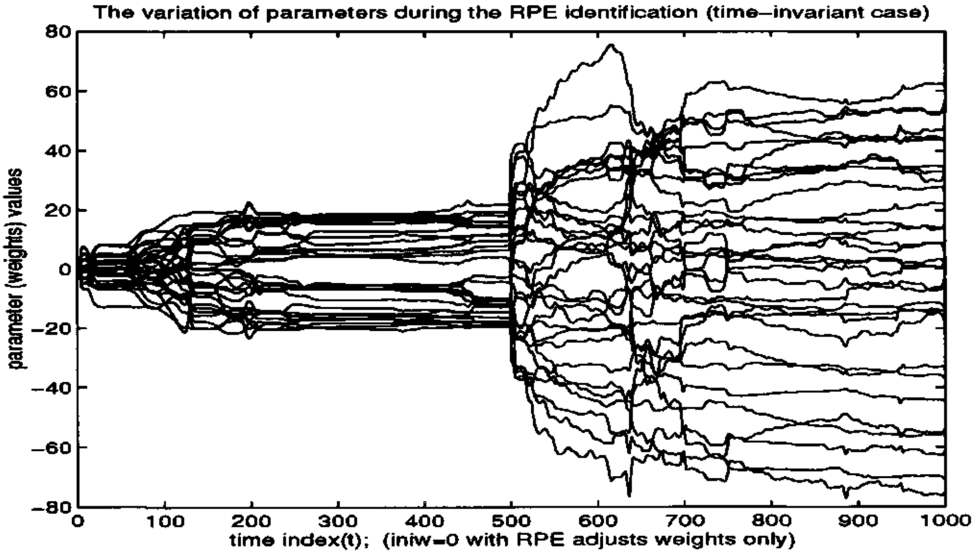

Figure 4.3:The variation of the identified weight of the NUFZY model during the validation in the time-invariant case, when initial weighting values are set to 0.

In order to study time variability, the dynamical system is forced to change its status from Eq.(4.37) to the next status governed by

$$y(k+1)=\frac{x_1x_2x_3x_5(x_3-2)+x_4}{3+x_2^2+x_3^2} \tag{4.39}$$

where, as before, $[x_1, x_2, x_3, x_4, x_5]$ = [y(k), y(k-1), y(k-2) , u(k), u(k-1)]. The following procedure was used to test this time-variant case. The training data are generated as before with a random input signal uniformly distributed in the intervals [-1,1], but the first 250 training points were generated by following Eq.(4.37) and the remaining 250 by following Eq.(4.39), thus simulating a sudden change in parameters. Next, a validation data set is created by first generating 300 points according to Eq.(4.39) and then another 700 according to Eq.(4.37), which means that the system returns to its original status. The data are arranged in this way so that the adaptation ability of RPE method can be examined. The validation result of the RPE on-line tuning initialized with $\mathbf{w}_{ols}$ shows that good prediction is still attained as shown in Figure 4.4. It is also observed that the RPE tuning converges fast to a new working point when the system parameters switch to other values at the 300^{th} time step.

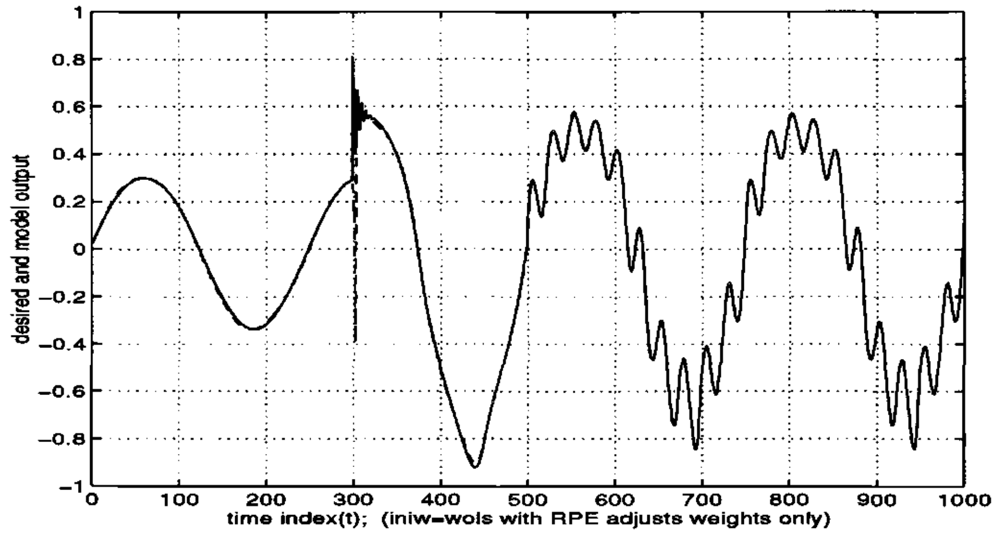

Figure 4.4: The predicted output of the NUFZY model superimposed on the desired output in a time-variant case. Solid line - the desired output ; dashed line - the predicted output of the NUFZY model. Note, they are hardly distinguishable.

4.4.2 Example 2 - prediction of tomato dry weight production

Agricultural plant growth in a conditioned environment inherently represents a system with a nonlinear character and some undetected time-variant parameters. In this example, dry matter production of tomatoes as a function of environmental factors, such as temperature, CO_2 concentration, and radiation, is considered. Data are from the experiments in Wageningen [2] where three experiments have been done on three different growing seasons in 1988. Every 7 - 10 days during these experiment periods, the dry matter amount of the tomato plants is measured by destructive measurements. The total dry weights (TDW) are used for the simulation model.

The goal of NUFZY modeling is to identify the dynamic growing process of tomato and to predict the total dry weights of tomato at the next sampling date. In other words, a NUFZY model was applied to describe the unknown relationships between the environmental factors and plant growth as given below:

$$\hat{y}(k+1) = f(D(k), y(k)) \tag{4.40}$$

where D(k) represents the disturbances to the system. In this case they are the averaged radiation (RAD) and averaged ambient CO_2 concentration of the greenhouse between the sampling intervals from t(k-1) to t(k). y(k) and $\hat{y}(k+1)$ represent the measured and predicted total dry weights (TDW) of tomato at sampling dates t(k) and t(k+1), respectively. The function $f(.)$ represents the unknown dynamics of the plant. The NUFZY model then approximates Eq.(4.40) as

$$\hat{TDW}(k+1) = f_{NUFZY}(\,RAD(k), CO_2(k)\,, TDW(k)\,) \tag{4.41}$$

Data of experiment 1 and experiment 3, in total 31 tuples (RAD,CO_2,TDW), are used for training and to generate initial weights for the validation and on-line prediction with the data of experiment 2 (15 data pairs). Owing to limited data length, the RPE tuning of the NUFZY model is done by feeding these data to the NUFZY model repeatedly up to 5 times to tune the output weights, which are initially set to zero. At the end of the training process, it is found that the best identified results were obtained by NUFZY(3×4×2:12;Gau). This notation means that 3, 4, and 2 Gaussian membership functions are assigned to input variables RAD, CO_2, and TDW, respectively. The results of training and on-line prediction are depicted in Figure 4.5.

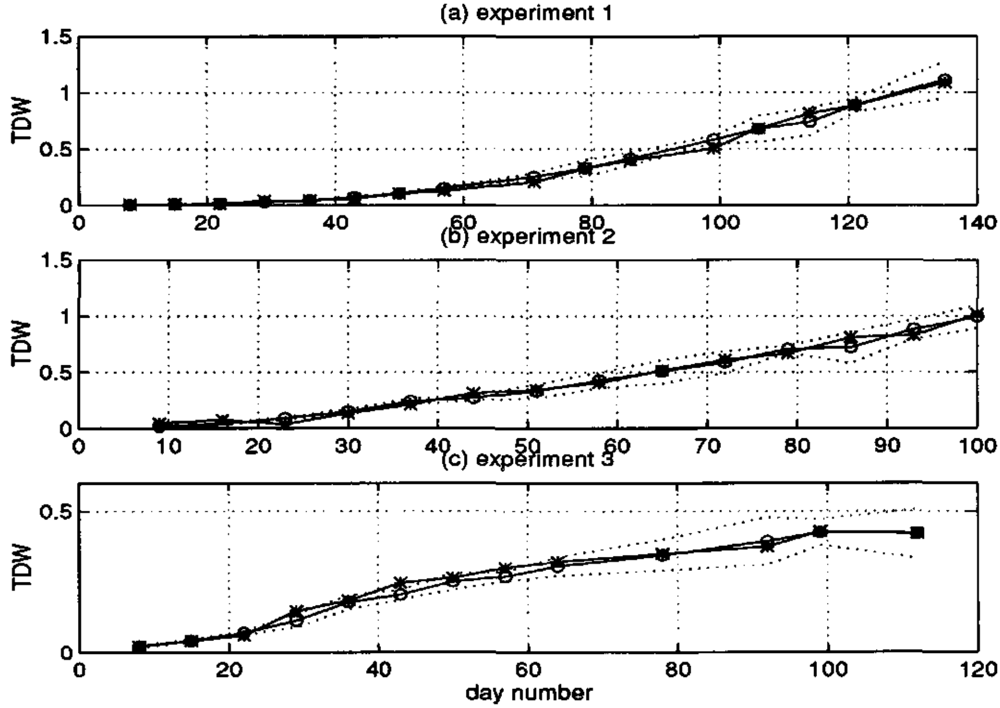

Figure 4.5: The measured TDW of tomatoes (circle-solid line) with 95% confidence interval (dotted line) and NUFZY(3×4×2:24;Gau) predicted TDW (star-solid line)

From Figure 4.5, it can be seen that the predicted TDW by the NUFZY model is located within 95% confidence interval of the measured TDW for both initialization and on-line prediction, showing that a good prediction of TDW has been achieved. This example exhibits the feasibility of RPE tuning for the NUFZY model in a real world application.

4.5 CONCLUSIONS

In this chapter, sensitivity derivatives have been derived for the NUFZY system, which enables the employment of the recursive prediction error method for tuning of the NUFZY model and the identification of the unknown dynamic nonlinear system in an on-line fashion. The recursive prediction error method demonstrates that good model accuracy can be achieved by just tuning the consequent weights of the NUFZY system. Since this problem is linear-in-the-parameters, it is convenient to apply an orthogonal least squares method to the first batch of data, in order to obtain a set of optimal weight parameters as initialization for the recursive prediction error method, thus enabling a faster adjustment for on-line tuning.

Although the results of tuning the parameters of $\mathbf{c}$ and σ in the antecedent part of the NUFZY system are not presented in this chapter, it is worthwhile to mention that simultaneous tuning of all parameters does not give much improvement on prediction, as compared to merely tuning the consequent weights of the NUFZY system, see [68] . This is a significant point to support us in exclusively adjusting the consequent weights, allowing fast tuning of parameters for on-line application while achieving comparative model accuracy. Moreover, it appears that if parameters $\mathbf{c}$ and σ are tuned at the same time, they may get values that are outside the expected ranges, making the results difficult to interpret. To this point, an interior penalty method used in [36] can be an alternative to tune parameters $\mathbf{c}$ and σ bounded in some reasonable domains after tuning. However, this leads to a complicated nonlinear parameter optimization problem that needs more computational efforts to deal with. As such, it is conceivable that one will favor the simpler tuning process, provided the similar model accuracy can be achieved.

5. KNOWLEDGE INTERPRETATION FROM TAKAGI-SUGENO FUZZY MODELS

Sariputra, matter is not different from emptiness, and emptiness is not different from matter. Matter is emptiness and emptiness is matter. So too are sensation, recognition, volition and consciousness.

- Heat of the Prajna - Paramita Sutra

舍利子，色不異空，空不異色；色即是空，空即是色。受、想、行、識，亦復如是。 般若波羅蜜多心經

5.1 INTRODUCTION*

The commonly used Mamdani type of fuzzy rule has advantages both with respect to the incorporation of *a priori* knowledge, as well as with respect to interpretability of the rules, because both the antecedent and consequent of rules are expressed linguistically, so that the Mamdani type of fuzzy rule is more intimate to the human's intuitive knowledge. Nevertheless, its main disadvantage is that information representations differ from one expert to the other. This will cause consistency problems in applications as it is difficult for the system developer to judge which representations should be taken and how to integrate them.

Alternatively, the Takagi-Sugeno (denoted as T-S) type of fuzzy rule provides a means of simple calculation in a fuzzy system as it makes use of a linear combination of the system inputs (or a constant term) as its consequent and then the weighted average output is obtained based on all fuzzy rules. Yet, the interpretation of the T-S fuzzy rule is troublesome since the consequent is expressed by a linear function of system inputs, which does not help much to understand the global behavior of the system, not to mention a meaningful linguistic description of how the system works. Also, for T-S fuzzy rules the incorporation of qualitative information is difficult because most qualitative information is neither represented in a crisp form nor as a linear function. Hence, the fuzzy system using the T-S fuzzy rules is mainly applied for identification and to construct the corresponding fuzzy model from experimental data without utilizing much qualitative information. The very use of qualitative information of the T-S fuzzy model is merely limited to the determination of the number of fuzzy sets in the antecedent part of the fuzzy rule (i.e., the partitioning of input space of a fuzzy rule base) and the initialization of these parameters.

With interpretability in mind, we propose to associate a set of parameters, called consequent significance level (CSL) ρ_{rj}, to the consequent fuzzy sets of a MISO fuzzy system with Mamdani fuzzy rules, in order to overcome the above mentioned inconsistency problem. The CSL parameter describes the degree of confidence of the contribution of the j^{th} output fuzzy set to the consequent of the r^{th} fuzzy rule (where indices r and j will be discussed later). By introducing the concept of the consequent significance level, some interesting results are obtained. First, an extended Mamdani (denoted as EM) type of fuzzy rule can be established. It is found that the ordinary Mamdani fuzzy rule becomes a special case of the EM fuzzy rule when the CSL parameter is taken as either unity or zero. Second, under some conditions, the output of EM fuzzy rules can be related to that of T-S fuzzy rules. This implies that the crisp consequent of the T-S fuzzy rule can be transformed into a Mamdani like fuzzy rule - with an interpretable set of linguistic terms - associated with a CSL parameter. More specifically, the linear function of system inputs (or, a crisp real value) in the consequent of the T-S fuzzy model can be transformed to a linear function of outputs with the CSL parameters as coefficients. Hence, the T-S fuzzy rule becomes linguistically interpretable in a similar way as

* This chapter is an extended version of the paper [71] , titled 'On the interpretation of two types of fuzzy rules' in *Proceedings of the Second International ICSC Symposium on Fuzzy Logic and Applications ISFL-97*, Zürich, Switzerland, February 12-14, 1997, pp 240-246.

the extended Mamdani fuzzy rule. The above transformation can be realized because the fuzzy model has dual representations, i.e., the output can be represented by a linear function of either the system inputs, or, the system outputs. This idea will be purified in this chapter.

Without loss of generality, we will confine ourselves in the following discussion to the multi-input single-output fuzzy system. In section 5.2, a comparison will be made between the T-S type of fuzzy rule and fuzzy model, and the ordinary Mamdani type of fuzzy rule and fuzzy model. In section 5.3, we will introduce the newly defined parameter and the resultant extended Mamdani type of fuzzy rules. The link of the EM fuzzy rules to the T-S fuzzy rules is presented in section 5.4. A simple synthetic example in section 5.5 illustrates how the identified T-S fuzzy rule can be transformed to be interpretable as the EM fuzzy rule. Finally, the conclusion and discussion are addressed in section 5.6.

5.2 COMPARISON OF TWO TYPES OF FUZZY RULES AND THEIR MODELS

In this section, we will first describe the T-S fuzzy rule and model used to carry out fuzzy reasoning. Then a more detailed description of fuzzy inference is given for the Mamdani type of fuzzy rule in order to pave the way for the consequent significance level parameters to construct the extended Mamdani fuzzy rule in the next section.

5.2.1 Takagi-Sugeno type of fuzzy rule and model

Consider a MISO fuzzy system, then a first-order Takagi-Sugeno type of fuzzy rule is expressed as

$$\boldsymbol{R}^{r}_{(TS)}\text{: } \textbf{\textit{IF}} \quad x_1 \text{ is } A^{r}_{k1}(x_1) \textbf{ AND } .. \; x_i \text{ is } A^{r}_{ki}(x_i) \textbf{ AND } .. \; x_{ni} \text{ is } A^{r}_{kni}(x_{ni})$$

$$\textbf{\textit{THEN}} \quad y = a_0^r + \sum_{i=1}^{ni} a_i^r x_i \tag{5.1.a}$$

or in short format

$$\boldsymbol{R}^{r}_{(TS)}\text{: } \textbf{\textit{IF}} \{x_i \text{ is } A^{r}_{ki}(x_i)\} \textbf{\textit{ THEN }} y = a_0^r + \sum_{i=1}^{ni} a_i^r x_i \tag{5.1.b}$$

The consequent of an individual fuzzy rule is formed as a linear function of system inputs together with a set of parameters, a^r_0, a^r_1, .., a^r_{ni}, which need to be identified. In case of the zeroth-order T-S fuzzy rule, which gives (in short)

$$\boldsymbol{R}^{r}_{(TS)}\text{: } \textbf{\textit{IF}} \{x_i \text{ is } A^{r}_{ki}(x_i)\} \textbf{\textit{ THEN }} y = w_r \tag{5.2}$$

the real number w_r in the consequent of the T-S fuzzy rule represents a singleton membership function, which is hardly interpretable linguistically.

Given a set of inputs $\mathbf{x}$ ($\in \mathbb{R}^{ni}$), for all R fuzzy rules, the aggregated output, denoted as $y_{(TS)}$, employing the weighted sum of the consequent part defined in Eq.(5.1), results in

$$y_{(TS)}(\mathbf{x}) = \frac{\sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot (a_0^r + \sum_{i=1}^{ni} a_i^r x_i)}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} = \sum_{r=1}^{R} \bar{\nu}_r(\mathbf{x}) \cdot (a_0^r + \sum_{i=1}^{ni} a_i^r x_i) \tag{5.3.a}$$

or, in the case of a zeroth-order T-S rule (Eq.(5.2))

$$y_{(TS)}(\mathbf{x}) = \frac{\sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot w_r}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} = \sum_{r=1}^{R} \bar{\nu}_r(\mathbf{x}) \cdot w_r \tag{5.3.b}$$

where $\nu_r(\mathbf{x})$ and $\bar{\nu}_r(\mathbf{x})$ represent the firing strength and normalized firing strength of the antecedent part of the r^{th} fuzzy rule, respectively. This firing strength ν_r is obtained as a result of a T-norm operation for the linguistic AND connection in the antecedent that uses the system inputs as arguments of membership functions. It can be defined either by min operation or product operation,

$$\begin{aligned} \nu_r(\mathbf{x}) &= T(\mu_{A_{k1}^r}(x_1), \cdots, \mu_{A_{kni}^r}(x_{ni})) \\ &= \begin{cases} \min(\mu_{A_{k1}^r}(x_1), \cdots, \mu_{A_{kni}^r}(x_{ni})) \\ \prod_{i=1}^{ni} \mu_{A_{ki}^r}(x_i) \end{cases} \end{aligned} \tag{5.4}$$

Once the term $\bar{\nu}_r$ is obtained, Eq.(5.3.b) becomes a linear regression, so that parameters w_r's can be identified by the least squares method. Analogously, the parameter set in the linear function a_i^r's (Eq.(5.3.a)) can be found by the least squares method too. It is also noted that the defuzzified output is obtained by taking the summation of all R fuzzy rules as an aggregation. This method is referred to as *fuzzy-mean* (FM) defuzzification [30] . The advantage of the T-S type of fuzzy rule is that the defuzzified output is linear in the parameters, facilitating mathematical analysis and calculation. Yet, the weakness is that the consequent cannot be interpreted easily. The rest of this chapter will focus only on the case of the zeroth-order T-S fuzzy rules.

5.2.2 Mamdani type of fuzzy rule and model

The Mamdani type of fuzzy rule mainly differs from the T-S fuzzy rule by its consequent part, where the consequent is expressed by fuzzy sets, rather than crisp values or linear functions of system inputs. Assuming that it has the same antecedent as that of the T-S rule, it is typically expressed as

$$\boldsymbol{R}^{r}_{(M)}: \textbf{\textit{IF}} \quad x_1 \text{ is } A^{r}_{k1}(x_1) \textbf{ AND} .. \ x_i \text{ is } A^{r}_{ki}(x_i) \textbf{ AND} .. \textbf{ AND } x_{ni} \text{ is } A^{r}_{kni}(x_{ni})$$

$$\textbf{\textit{THEN}} \ \ y \text{ is } B^{r}_{j}(y) \tag{5.5.a}$$

or in short

$$\boldsymbol{R}^{r}_{(M)}: \textbf{\textit{IF}} \ \{x_i \text{ is } A^{r}_{ki}(x_i)\} \ \textbf{\textit{THEN}} \ y \text{ is } B^{r}_{j}(y) \tag{5.5.b}$$

where output y has N_b fuzzy sets indexed by j, for j = 1, .. , N_b. The notation $B^r_j(y)$ indicates that in rule r the output belongs to the j^{th} linguistic set of y. Also, $B^r(y)$ refers to the linguistic set of rule r (irrespective of j), and $B_j(y)$ refers to the linguistic set j of y (irrespective of r). It is obvious that the consequent of some fuzzy rule $B^r(y)$, $r \in [1, R]$, may share the same fuzzy set $B_j(y)$, $j \in [1, N_b]$, provided $N_b \le R$. The following definitions will be used.

(D.1) cardinal set C_R is defined as a collection of the numbers of fuzzy rules indexed by subscript r. $C_R = \{1, 2, .., r, .., R\}$

(D.2) cardinal set C_B is defined as a collection of the numbers of output fuzzy sets indexed by subscript j. $C_B = \{1, 2, .., j, .., N_b\}$

(D.3) cardinal set C_j, indexed by subscript r_j* and $C_j \subset C_R$, is defined as a collection of those fuzzy rules, whose consequent fuzzy sets, $B^r(y)$'s, are identical,
$C_j = \{ r_j* \mid r_j* \in C_R \text{ such that } B^{r_j*}(y) = B_j(y)\}$; $\forall j \in C_B$

(D.4) complement cardinal set, $\overline{C}_j$, defined as a set which is complementary to C_j, is given by $\overline{C}_j = C_R - C_j$

(D.5) cardinality of set C_j, denoted by N_{cj}, is defined as the total number of its elements and is subject to

$$\sum_{j=1}^{N_b} N_{cj} = N_{c1} + N_{c2} + \cdots + N_{cN_b} = R$$

(D.6) indexing function, I(r), is defined as a function which returns the subindex (subscript j) of the fuzzy output set $B^r_j(y)$ with respect to the r^{th} fuzzy rule, i.e.,
$I(r) = j* \in C_B$ such that $B^r_j(y)$ (or, $B^r(y)$) $= B_{I(r)}(y) = B_{j*}(y) = B_j(y)$

Using definition (D.6), the consequent fuzzy set $B^r_j(y)$ in Eq.(5.5) can be denoted as $B_{j*}(y)$ via the indexing function $I(r)$†. Since the indexing function $I(r)$ maps the causal relation of a consequent fuzzy set in the r^{th} fuzzy rule to its correct subindex in C_B, it is needed to define both the cardinal sets C_R and C_B. Hence, in applying the Mamdani fuzzy model, it is necessary for the designers to define the output fuzzy sets in advance, which is implicitly linked to the fuzzy rules via the indexing function.

The following example explains the use of the above definitions. Suppose we have 6 rules (r = 1, .., 6) and three output fuzzy sets ('small', 'medium', and 'large' denoted by $B_1(y)$, $B_2(y)$ and $B_3(y)$, respectively, j = 1, 2, 3). Rules 1, 3, and 4 have the same consequent 'medium', $B_2(y)$, and rules 2 and 6 correspond to consequent 'large', $B_3(y)$, and rule 5 has consequent 'small', $B_1(y)$. Hence

$$C_1 = \{ r_1^* \mid B^{r_1^*}(y) = B_1(y)\} = \{5\}, \qquad \overline{C}_1 = \{1, 2, 3, 4, 6\}, \quad N_{c1} = 1;$$

$$C_2 = \{ r_2^* \mid B^{r_2^*}(y) = B_2(y)\} = \{1, 3, 4\}, \qquad \overline{C}_2 = \{2, 5, 6\}, \quad N_{c2} = 3;$$

$$C_3 = \{ r_3^* \mid B^{r_3^*}(y) = B_3(y)\} = \{2, 6\}, \qquad \overline{C}_3 = \{1, 3, 4, 5\}, \quad N_{c3} = 2;$$

$$\sum_{j=1}^{3} N_{cj} = 1+3+2 = 6 \text{ ; and}$$

$$I(1) = 2;\ I(2) = 3;\ I(3) = 2;\ I(4) = 2;\ I(5) = 1;\ I(6) = 3.$$

With the indexing function Eq.(5.5.b) can be rewritten as

$$R^r_{(M)}\text{: } \textit{IF } \{x_i \text{ is } A^r_{ki}(x_i)\} \textit{ THEN } y \text{ is } B_{j*}(y) \tag{5.5.c}$$

In order to obtain the defuzzified output of the Mamdani fuzzy model, some additional assumptions of the output fuzzy set $B^r_j(y)$ are made to simplify the calculation of defuzzification.

(A.1): $B^r_j(y)$ is a normal fuzzy set; i.e., $\max\{\mu_{B^r_j}(y)\} = 1$ and $\mu_{B^r_j}(y) \in [0,1]$.

(A.2): $B^r_j(y)$ has compact support (CS) in the domain of Y, the universe of discourse of output y; i.e., $CS(B^r_j(y)) = \{ y \in Y_j \subset Y \mid \mu_{B^r_j}(y) > 0\}$. For a compact support $B^r_j(y)$, the argument y is only defined in a subdomain Y_j, a finite subset of Y. Hence, the area of $B^r_j(y)$, $\int_{Y_j} \mu_{B^r_j}(y)dy$, becomes finite integratable in the subdomain Y_j.

---

† C_R and C_B can be linked as followed,

$$C_R \text{ (with element r) } \underset{C_j = \{ r_j^* \in C_R\}}{\overset{I(r) = j^* \in C_B}{\rightleftharpoons}} C_B \text{ (with element j)}$$

(A.3): The areas of each compact support fuzzy set $B^r_j(y)$ are identical. For example, if area($B^r(y)$) is defined as $a^r = \int_Y \mu_{B^r}(y)dy$ (with respect to the r^{th} fuzzy rule), then the following equality exists: $a^1 = a^2 = .. = a^r = .. = a^R$. Analogously, the area($B_j(y)$), defined as $a_j = \int_Y \mu_{B_j}(y)dy$ (with respect to the j^{th} output fuzzy sets), has the similar equality: $a_1 = a_2 = .. = a_j = .. = a_{Nb}$.

It is noted that if $B^r_j(y)$ is symmetrical, then the location of the centroid of a compact support fuzzy set $B^r_j(y)$ projected onto Y, equals to the point where the membership function of $\mu_{B^r_j}(y)$ reaches its maximum value. This point, will be denoted as z^r_j, (or, z^r or z_j, in the following) is a numerical representation of $B^r_j(y)$ and is defined by $z^r_j = \{ y \in Y_j \mid \max(\mu_{B^r_j}(y)) = 1\}$. When there are several maximum values (such as trapezoidal membership function), the location of the point which takes the mean of these maximal points is taken as z^r_j. Due to property of compact support in $B^r_j(y)$, one merely has to take the mean value of Y_j as its z^r_j, provided $B^r_j(y)$ is symmetrical. Hence, according to (A.2) - (A.3), the first moment of $B^r_j(y)$, $\int_Y \mu_{B^r_j}(y)ydy$, can be just represented by $a^r_j \cdot z^r_j$.(or, $a^r \cdot z^r$ or $a_j \cdot z_j$ in the following).

When implementing Mamdani fuzzy rules, a resultant consequent fuzzy set B'(y) shall be calculated in order to carry out the fuzzy inference. B'(y) can be obtained in two ways. One is that B'(y) is aggregated based on all R fuzzy rules (using fuzzy sets denoted as $B^r(y)$); the other is based on all the N_b output fuzzy sets (using fuzzy sets denoted as $B_j(y)$). In the following, we will discuss these two aggregation methods.

First, when B'(y) is aggregated based on all R fuzzy rules, using R fuzzy sets $B^r(y)$, it can be expressed in terms of a membership function $\mu_{B'}(y)$ as

$$\begin{aligned} \mu_{B'}(y) &= \bigcup_r I_{CC}(\nu_r(\mathbf{x}), \mu_{B^r}(y)) \\ &= \mathop{S_a}_{r=1}^{R}(\nu_r(\mathbf{x}) \circ \mu_{B^r}(y)) \end{aligned} \tag{5.6}$$

where I_{CC} indicates a fuzzy implication based on classical conjunction performed by '$\circ$', a T-norm operation; see Chapter 2; whereas S_a is a S-norm operator for aggregation and $\mu_B{}^r(y)$ is the membership function of the consequent fuzzy set $B^r(y)$ in the r^{th} fuzzy rule. (Note that $\mu_B'(y)$ usually has a complicated shape.)

When the *sum-product* inference method is used to replace S_a and $\circ$, the membership function $\mu_B'(y)$ of B'(y) in Eq.(5.6) becomes (Note, in terms of R fuzzy sets $B^r(y)$)

$$\mu_{B'}(y) = \sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot \mu_{B^r}(y) \tag{5.7}$$

Next, we define the following properties of $B^r(y)$.

(D.7) active area of $B^r(y)$, denoted as a^r

$$a^r = \int_Y \mu_{B^r}(y)dy$$

(D.8) first moment of $B^r(y)$, denoted as m^r

$$m^r = \int_Y \mu_{B^r}(y)ydy = a^r \cdot z^r$$

where z^r is the location of centroid of $B^r(y)$ projected on Y under the assumption (A.3). If assumption (A.3) is dropped, (D.8) usually defines z^r. From definitions (D.7), (D.8), and Eq.(5.7), one can derive the resultant active area, a', and the resultant first moment, m', of B'(y) as

$$\begin{aligned} a' &= \int_Y \mu_{B'}(y)dy = \int_Y (\sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot \mu_{B^r}(y))\,dy \\ &= \sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot (\int_Y \mu_{B^r}(y)dy) = \sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot a^r \end{aligned} \tag{5.8}$$

$$\begin{aligned} m' &= \int_Y \mu_{B'}(y)ydy = \int_Y (\sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot \mu_{B^r}(y))\,y\,dy \\ &= \sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot (\int_Y \mu_{B^r}(y)\,y\,dy) = \sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot m^r = \sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot a^r \cdot z^r \end{aligned} \tag{5.9}$$

Eq.(5.8) and Eq.(5.9) represent a weighted sum of active area and first moment of B'(y), respectively. Hence, based on the above assumptions (A.1) - (A.3), the defuzzified output of such Mamdani fuzzy rules, denoted as $y_{(M)}$, is then obtained by

$$y_{(M)}(\mathbf{x}) = \frac{m'}{a'} = \frac{\sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot a^r \cdot z^r}{\sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot a^r} = \sum_{r=1}^{R} \bar{\nu}_r(\mathbf{x}) z^r \tag{5.10}$$

where $\bar{\nu}_r$ is defined as in Eq.(2.41). From Eq.(5.10), it is interesting to note that the system output of the ordinary Mamdani fuzzy model forms a linear function of z^r, a numerical representation of the output fuzzy set $B^r(y)$ under above assumptions. This linear function is subject to the constraint $\sum_{r=1}^{R} \overline{v}_r(x) = 1$, as similar to the zeroth-order T-S fuzzy model.

We will next derive the resultant output fuzzy set B'(y) from N_b output fuzzy sets $B_j(y)$. This will contrast with the derivation of Eq.(5.8) and Eq.(5.9), which are based on R fuzzy sets $B^r(y)$. The membership function $\mu_{B'}(y)$ of B'(y) is modified from Eq.(5.6) based on the N_b fuzzy sets $B_j(y)$ together with definitions (D.3) and (D.5),

$$\begin{aligned} \mu_{B'}(y) &= \mathop{S_a}_{r=1}^{R} (v_r(x) \circ \mu_{B^r}(y)) \\ &= \mathop{S_{ao}}_{j=1}^{N_b} [(\mathop{S_{ai}}_{r_j^*}^{N_{Cj}} v_{r_j^*}(x)) \circ \mu_{B_j}(y)] \end{aligned} \tag{5.11}$$

It can be seen that Eq.(5.11) performs two steps of aggregation through the outer and the inner S-norm aggregations, S_{ao} and S_{ai}. Since there exist predefined ownership relations that map R firing strength sets into N_b classes of output fuzzy sets (see definitions (D.3)-(D.6)), first those firing strengths v_r's (in total N_{cj}) which have the same consequent part $B_j(y)$ can be aggregated by the operation S_{ai}, together with $\mu_{Bj}(y)$ to perform the fuzzy implication by a T-norm operator '$\circ$'; whereas S_{ao} finally aggregates results from all N_b output fuzzy sets. When S_{ao} and S_{ai} are chosen as summation and the T-norm operator, $\circ$, as algebraic product, Eq.(5.11) becomes (in terms of N_b fuzzy sets $B_j(y)$)

$$\mu_{B'}(y) = \sum_{j=1}^{N_b} [(\sum_{r_j^*=1}^{N_{Cj}} v_{r_j^*}(x)) \cdot \mu_{B_j}(y)] \tag{5.12.a}$$

where the term $\sum_{r_j^*=1}^{N_{Cj}} v_{r_j^*}(x)$ indicates the sum of all v_r's which have the same output fuzzy set $B_j(y)$. It can also be denoted by β_j, with the following definition

(D.9) implicated DOF (degree of fulfillment), β_j

$$\beta_j(x) = \sum_{r_j^*=1}^{N_{Cj}} v_{r_j^*}(x) \quad \text{with property} \quad \sum_{j=1}^{N_b} \beta_j(x) = \sum_{j=1}^{N_b} (\sum_{r_j^*=1}^{N_{Cj}} v_{r_j^*}(x)) = \sum_{r=1}^{R} v_r(x)$$

Hence, Eq.(5.12.a) can be rewritten as

$$\mu_{B'}(y) = \sum_{j=1}^{N_b} \beta_j(x) \cdot \mu_{B_j}(y) \tag{5.12.b}$$

Similarly, we can also define the active area and the first moment of $B_j(y)$, respectively,

(D.10) active area of $B_j(y)$, denoted as a_j

$$a_j = \int_Y \mu_{B_j}(y)dy = \int_{Y_j} \mu_{B_j}(y)dy + \int_{Y-Y_j} 0 \cdot dy$$
$$= \int_{Y_j} \mu_{B_j}(y)dy$$

(D.11) active first moment of $B_j(y)$, denoted as m_j

$$m_j = \int_Y \mu_{B_j}(y)\, y\, dy = \int_{Y_j} \mu_{B_j}(y)\, y\, dy$$
$$= a_j \cdot z_j$$

Note, in definition (D.10), since $B_j(y)$ has compact support in Y_j and $\mu_{Bj}(y) \in (0,1]$ when $y \in Y_j$, therefore, $\mu_{Bj}(y) = 0$ in the complementary domain of Y-Y_j. In (D.11), z_j is defined as the point where the centroid of $B_j(y)$ projected on Y according to (D.8). Using definitions (D.9) - (D.11) and Eq.(5.12.b), the resultant active area, a', and the resultant first moment, m', of B'(y) are derived as below,

$$a' = \int_Y \mu_{B'}(y)dy = \int_Y (\sum_{j=1}^{N_b} \beta_j(x) \cdot \mu_{B_j}(y))dy = \sum_{j=1}^{N_b} \beta_j(x) \cdot (\int_Y \mu_{B_j}(y)dy)$$
$$= \sum_{j=1}^{N_b} \beta_j(x) \cdot (\int_{Y_j} \mu_{B_j}(y)dy) = \sum_{j=1}^{N_b} \beta_j(x) \cdot a_j \tag{5.13}$$

and

$$m' = \int_Y \mu_{B'}(y)ydy = \sum_{j=1}^{N_b} \beta_j(x) \cdot a_j \cdot z_j \tag{5.14}$$

Hence, the defuzzified output of such Mamdani fuzzy rules, based on the N_b output fuzzy sets and the centroid of gravity method, is obtained by

$$y_{(M)}(x) = \frac{m'}{a'} = \frac{\sum_{j=1}^{N_b} \beta_j(x) \cdot a_j \cdot z_j}{\sum_{j=1}^{N_b} \beta_j(x) \cdot a_j} \tag{5.15}$$

According to assumption (A.3), together with the property of β_j from (D.9), the defuzzified output can be simplified further to

$$y_{(M)}(\mathbf{x}) = \frac{\sum_{j=1}^{N_b} \beta_j(\mathbf{x}) \cdot z_j}{\sum_{j=1}^{N_b} \beta_j(\mathbf{x})} = \frac{\sum_{j=1}^{N_b} \beta_j(\mathbf{x}) \cdot z_j}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} = \sum_{j=1}^{N_b} \left(\frac{\beta_j(\mathbf{x})}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} \right) \cdot z_j \qquad (5.16.a)$$

if we define the term $\overline{\beta}_j$ as

(D.12) normalized implicated DOF, $\overline{\beta}_j$

$$\overline{\beta}_j(\mathbf{x}) = \frac{\beta_j(\mathbf{x})}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} = \frac{\sum_{r_j^*=1}^{N_{Cj}} \nu_{r_j^*}(\mathbf{x})}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} = \sum_{r_j^*=1}^{N_{Cj}} \overline{\nu}_{r_j^*}(\mathbf{x})$$

with property $$\sum_{j=1}^{N_b} \overline{\beta}_j(\mathbf{x}) = \sum_{j=1}^{N_b} \left(\frac{\beta_j(\mathbf{x})}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} \right) = \frac{\sum_{j=1}^{N_b} \beta_j(\mathbf{x})}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} = \frac{\sum_{r=1}^{R} \nu_r(\mathbf{x})}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} = 1$$

Therefore, Eq.(5.16.a) can be rewritten as

$$y_{(M)}(\mathbf{x}) = \sum_{j=1}^{N_b} \overline{\beta}_j(\mathbf{x}) \cdot z_j \qquad (5.16.b)$$

From Eq.(5.16.b), it is again noted that the system output forms a linear function of z_j, a numerical representation of output fuzzy set $B_j(y)$ under the above assumptions.

This linear function is subject to the constraint $\sum_{j=1}^{N_b} \overline{\beta}_j = 1$, defined in (D.12).

We have the following remarks for the above definitions and assumptions.

(R.1) With respect to the assumption (A.1), most applications of fuzzy systems usually let the fuzzy set $B^r_j(y)$ be normal since it typically reflects the human intuition. Other non-normal fuzzy sets, like subnormal ($\max\{\mu_{B^r_j}(y)\} < 1$) and supernormal (max{

$\mu_{B^r_j}(y)\}>1\}$), are also possible but are rarely applied in practice since they do not agree to the fuzzy theory.

(R.2) The finite subset Y_j of Y is defined as a compact set and enables the calculation of the area of $B^r_j(y)$ in practice. In addition to the symmetry property, the assumption that each $B_j(y)$ has identical areas will simplify the calculation process as well. As a result, only the parameter z^r_j is needed, which can be easily determined based on *a priori* knowledge of the designers. Although the shape factors, like the bandwidth of $\mu_{Bj}(y)$, are important in some sense, it becomes insignificant to define these factors 'precisely' when one is to incorporate qualitative information because the qualitative knowledge from human itself is essentially vague. The merit that one can gain is a less complex fuzzy system to be processed. In any case, a numerical representation of output fuzzy set, z^r_j, suggests that the shape factors of the output membership functions are insignificant under our present assumptions. As long as z^r_j is determined, the fuzzy reasoning of the Mamdani fuzzy model can still be realized under the above assumptions. It is noticed that the commonly used Gaussian type of membership function of fuzzy set $B_j(y)$ is not compact support in a finite interval since its membership value is greater than zero for the whole universe of Y. Instead, other types of modified membership functions can be applied to agree with the properties of compact support, symmetry, and identical area. For instance, the triangular shape of membership that is generated by B-splines functions, or, by the modified compact Gaussian function [5] as defined below.

$$\mu_{B_j}(y)=\begin{cases}\exp^{-1}(-1)*\exp\left(-\dfrac{\frac{(z_{j,2}-z_{j,1})^2}{4}}{(z_{j,2}-y)(y-z_{j,1})}\right) & ;\text{ if } y\in Y_j=(z_{j,1},z_{j,2})\\ 0 & ;\text{ otherwise}\end{cases} \tag{5.17}$$

(R.3) In most situations, the output universe of discourse Y is defined in a "bounded" way to agree with reality. However, at the edges of this bounded domain, a Z- and an S-shaped membership functions may be chosen to be the most left and the right side ($\mu_{B_1}(y)$ and $\mu_{B_{Nb}}(y)$, respectively) membership function within the bounded domain of Y. By moderately choosing the minimal and the maximal values of Y, [Y_{min}, Y_{max}], the assumption (A.2) can be satisfied. Alternatively, as mentioned above the B-splines function and modified compact Gaussian function can also be chosen as output membership functions since they are defined in finite intervals.

(R.4) Comparing Eq.(5.10) and Eq.(5.16.b) to Eq.(5.3.a), it is found that the fuzzy system has a property of dual representations; i.e., the defuzzified output of the fuzzy system can be represented as a linear function either of system inputs (e.g., a T-S fuzzy model output, see Eq.(5.3.a)), or, of system outputs (e.g., a Mamdani fuzzy model output, which is subject to assumptions (A.1) - (A.3), see Eq.(5.10) and Eq.(5.16.b)). This property, as will be shown later, offers a clue to enable us to link the T-S fuzzy model and the Mamdani fuzzy model. In general, the number of output fuzzy sets is smaller than the number of total fuzzy rules, i.e., $N_b \le R$. When $N_b = R$, Mamdani type of fuzzy rules become Takagi-Sugeno type of fuzzy rules, implying that there are R linguistic descriptive levels of y, characterized by singleton membership functions, existing in the output domain. In such a special case, the consequent weight w_r (in Eq.(5.2)) of the NUFZY system can be regarded as a singleton membership function in the output domain Y.

(R.5) Although different aggregation methods have resulted in different representations of the defuzzified outputs of the Mamdani fuzzy model (Eq.(5.10) and Eq.(5.16)), they are equal to each other. The different representation of the z^r (in Eq.(5.10)) and z_j (in Eq.(5.16)) can be made equivalent through the indexing function. For instance, z^r can be transformed to z_j using the indexing function I(r), so that z^r can be denoted as $z^r{}_j$, or equivalent to z_j (because of $z_{I(r)}$), when I(r) = j. Moreover, comparing Eq.(5.10) to Eq.(5.16), it can be found that the calculation of defuzzified output based on the aggregation of R fuzzy sets $B^r(y)$ is much easier and straightforward than that based on the N_b output fuzzy sets $B_j(y)$. It can be seen that the DOF ν_r in Eq.(5.10) can be obtained directly by manipulating the inputs, whereas the implicated DOF β_j in Eq.(5.16) needs further processing with the predefined ownership relations between ν_r and β_j.

From the previous analysis, it has been shown that the output of a Mamdani fuzzy model forms a linear function of $z^r{}_j$ under the above assumptions, similar to the T-S fuzzy model. Therefore, to interpret the zeroth-order T-S fuzzy model when directly relating Eq.(5.3.b) to Eq.(5.10), an inconsistency might arise that makes the direct interpretation of consequent weight w_r, in terms of $z^r{}_j$, become difficult. For instance, in the Mamdani fuzzy model, although r_1 and r_2 indicate two different fuzzy rules, they refer to the identical consequent $B^{r^*}{}_j(y)$ (with center $z^{r^*}{}_j$), for $r^* = r_1$ or r_2. From the T-S fuzzy rule, one may have two distinct values of the consequent weight, w_{r_1} ($\equiv z^{r_1}{}_j$) $\neq$ w_{r_2} ($\equiv z^{r_2}{}_j$), implying $z^{r_1}{}_j \neq z^{r_2}{}_j$. But, from the Mamdani fuzzy rule one has the equality $z^{r_1}{}_j = z^{r_2}{}_j$. The contradiction means that if $N_b < R$ it is not possible to interpret the T-S fuzzy rule directly by an ordinary Mamdani fuzzy rule. Next, we will find out how to resolve it through modification of the Mamdani fuzzy rules by the introduction of the CSL parameter.

5.3 EXTENDED MAMDANI FUZZY RULE AND MODEL

In the ordinary Mamdani type of fuzzy rule, the output fuzzy sets in the consequent are defined by the designers. Though each rule accompanied by a specific fuzzy set reflects the designer's own experience concerning the problem, it seems that the Mamdani type of fuzzy rules without modification cannot fit well to the same problem in a different situation.

Usually, these predefined fuzzy sets are fixed without further adjustment in the application of ordinary Mamdani fuzzy rules. From the functional approximation point of view, it has been shown in [30] that the Mamdani fuzzy model containing only one fuzzy set in the consequent of each rule has restricted ability to reproduce certain functions, even if the antecedent structure is correct. Also, it has been observed that by introducing extra parameters to each fuzzy rule, one can increase the flexibility of the Mamdani fuzzy models by relaxing its dependency on the definitions of the output sets [5] . Yet, if one is allowed to adjust the output fuzzy set of ordinary Mamdani fuzzy rules via some optimization methods, a conflict might occur where the optimized parameters of the output fuzzy set become controversial to the original knowledge. As an example, the center of the fuzzy set 'BIG' $B_{BIG}(y)$ may become smaller than that of 'SMALL' $B_{SMALL}(y)$ after optimization. Typically, it is assumed that the essence of the expert's rule shall be kept consistent for the same problem and, in practical application only the level of significant contribution of these fuzzy sets might differ from one situation to another.

Hence, in order to increase the modeling flexibility and overcome the above conflict, in this section, a new parameter, the consequent significance level ρ_{rj}, is introduced. It describes the degree of confidence of the contribution of the j^{th} output fuzzy set to the consequent of the r^{th} fuzzy rule. The idea is that, for each Mamdani fuzzy rule, the parameters of consequent significance level ρ_{rj} are assigned to all output fuzzy sets $B^r_j(y)$'s, $j = 1, .. , N_b$, thus forming multiple output fuzzy sets existing in the consequent of each fuzzy rule, in contrast to a single output fuzzy set in the consequent of the ordinary Mamdani fuzzy rule. Therefore, we have an 'extended Mamdani fuzzy rule' (denoted as EM) whose consequent is characterized by multiple fuzzy sets $B^r_j(y)$'s. From now on, the output fuzzy set $B^r_j(y)$ will be denoted as $B_j(y)$ as they are the same for all rules and because of the introduction of ρ_{rj}.

As in the ordinary Mamdani fuzzy model, all output fuzzy sets $B_j(y)$'s in the extended Mamdani fuzzy model, as determined by the designers in advance, are assumed to remain invariant; i.e., the shape factor and center location of $B_j(y)$ will not be adjusted after they are assigned. Hence, the consistence of *a priori* knowledge applied in the fuzzy system can be retained, and so the occurrence of the previously mentioned conflict can be avoided after tuning. Accordingly, the unknown consequent significance level ρ_{rj} becomes the only parameter that needs to be tuned. Another possible way to avoid such a conflict is to apply the interior penalty method to seek optimal parameter values with predefined constraints to the shape and location parameters of membership functions, see [36] . However, this requires a more complex constrained optimization process. Instead of this in the present approach we will merely focus on tuning the consequent significance level as explained in the next section.

When associating the consequent significance level to the ordinary Mamdani fuzzy rule, an extended Mamdani type of fuzzy rule can be constructed as below.

$$\begin{aligned} R^r_{(EM)}: \textit{IF} \quad & x_1 \text{ is } A^r_{k1}(x_1) \text{ AND} \; .. \quad x_i \text{ is } A^r_{ki}(x_i) \text{ AND} \; .. \quad x_{ni} \text{ is } A^r_{kni}(x_{ni}) \\ \textit{THEN} \; & y \text{ is } B_1(y) \text{ with } \rho_{r1}, \text{ and} \ldots, \; y \text{ is } B_j(y) \text{ with } \rho_{rj}, \text{ and} \ldots, \; y \text{ is } B_{Nb}(y) \text{ with } \rho_{rNb} \end{aligned} \tag{5.18.a}$$

or in short

$R^r_{(EM)}$: ***IF*** $\{x_i \text{ is } A^r_{ki}(x_i)\}$ ***THEN*** $\{y \text{ is } B_j(y) \text{ with } \rho_{rj}\}$ (5.18.b)

The parameters ρ_{rj}'s of the above extended Mamdani type of fuzzy rule are subject to the following constraints,

(C.1) $\rho_{rj} \in [0,1]$

(C.2) $\sum_{j=1}^{N_b} \rho_{rj} = 1; \forall r \in C_R = \{1, 2, .., R\}$

(C.3) $0 < \sum_{r=1}^{R} \rho_{rj} < R; \forall j \in C_B = \{1, 2, .., N_b\}$

One can consider that all the output fuzzy sets $B_j(y)$'s, $j \in [1, N_b]$, form a class of N_b fuzzy sets, and the consequent significance level ρ_{rj} indicates the membership value of the r^{th} fuzzy rule associated with the j^{th} fuzzy set $B_j(y)$. A larger value of ρ_{rj} means that the r^{th} fuzzy rule is more certain of being associated with the j^{th} output fuzzy set $B_j(y)$. The second constraint (C.2) shows the sum of all membership values of any single fuzzy rule (r = 1, .. , R) has to be unity. The third constraint (C.3) indicates that there can be no empty classes and there can be no classes which contain all R fuzzy rules, complying to the general knowledge used by human beings. It should be noted that all rules in the rule base formed like Eq.(5.18) are assumed to have contribution. In case any redundant rule ($\boldsymbol{R}^k$) occurs in the rule base, it shall be removed from the rule base and its corresponding ρ values ($\rho_{k1}\ \rho_{k2}\ ..\ \rho_{kN_b}$) are not taken into account in the following analysis.

<u>Remark:</u>

(R.6) it is obvious that when ρ_{rj} is taken as a crisp value of either 0 or 1, i.e. $\rho_{rj} \in \{0,1\}$, then the extended Mamdani type of fuzzy rule (see Eq.(5.18)) is equivalent to an ordinary Mamdani fuzzy rule (see Eq.(5.5)). This will be confirmed later with more details. For example, suppose there are 3 output fuzzy sets $B_1(y)$, $B_2(y)$, and $B_3(y)$, in the consequent. For the ordinary Mamdani type of fuzzy rule, the r^{th} fuzzy rule is merely assigned as 'y is $B_2(y)$' via the index function I(r) = 2. Then, in the extended Mamdani fuzzy rule context, this is the same as saying that the consequent of the r^{th} EM fuzzy rule is 'y is $B_1(y)$ with ρ_{r1} (= 0), and y is $B_2(y)$ with ρ_{r2} (= 1), and y is $B_3(y)$ with ρ_{r3} (= 0)'.

Since the defuzzified output of Mamdani fuzzy rules is calculated based on the first moment and aggregated active area of B'(y), we modify the previous definitions to associate them with the CSL parameters.

(D.13) significant active area of $B_j(y)$, defines the active area of the r^{th} rule associated with the j^{th} output fuzzy set and is denoted as a^r_j, given by

$$a^r_j = \rho_{rj} \cdot a_j$$

Using the definition (D.10), one can further derive the following relations,

$$a^r = \sum_{j=1}^{N_b} a^r_j = \sum_{j=1}^{N_b} \rho_{rj} \cdot a_j = \sum_{j=1}^{N_b} \rho_{rj} \cdot \left(\int_{Y_j} \mu_{B_j}(y)dy\right)$$
$$= \sum_{j=1}^{N_b} \rho_{rj} \cdot \left(\int_{Y} \mu_{B_j}(y)dy\right) = \int_{Y} \left(\sum_{j=1}^{N_b} \rho_{rj} \mu_{B_j}(y)\right)dy$$

Hence, according to definition (D.7), the joint membership function of $B^r(y)$ of the r^{th} EM fuzzy rule can be expressed in terms of $\mu_{Bj}(y)$ associated with ρ_{rj}

$$\mu_{B^r}(y) = \sum_{j=1}^{N_b} \rho_{rj} \cdot \mu_{B_j}(y) \tag{5.19}$$

<u>Remark:</u>

(R.7) As mentioned above, when ρ_{rj} is chosen as either unity or zero, i.e. $\rho_{rj} \in \{0,1\}$, through the indexing function I(r), there exists a $j^* \in C_B$ such that

$$\rho_{rj} = \begin{cases} 1 \ ; \ j = I(r) = j^* \in C_B \\ 0 \ ; \ j \in C_B \setminus j^* \end{cases}$$

therefore, $\mu_B{}^r(y)$ in Eq.(5.19) becomes

$$\mu_{B^r}(y) = \sum_{j=1}^{N_b} \rho_{rj} \cdot \mu_{B_j}(y) = \rho_{rj^*} \cdot \mu_{B_{j^*}}(y) = \mu_{B_j{}^*}(y)$$

This means that the fuzzy set $B^r(y)$ with respect to the r^{th} fuzzy rule of the extended Mamdani fuzzy rule is identical to the j^{*th} output fuzzy set $B_{j^*}(y)$ of the ordinary Mamdani fuzzy rule according to the definition (D.6). This also shows that the Mamdani fuzzy rule (as in Eq.(5.5)) can be regarded as a special case of the extended Mamdani fuzzy rule (as in Eq.(5.18)), provided that the rule significance level ρ_{rj} is chosen as a crisp value (either unity or zero) rather than a fuzzy membership value in [0,1] and condition (C.2) holds.

Again, as in the case of Eq.(5.6), when the *sum-product* inference method is used, the resultant membership function $\mu_{B'}(y)$ of $B'(y)$ is derived by associating it with the rule significance level ρ_{rj} as

$$
\begin{aligned}
\mu_{B'}(y) &= \mathop{S_a}_{r=1}^{R}(\nu_r(\mathbf{x})\circ\mu_{B^r}(y)) = \sum_{r=1}^{R}\nu_r(\mathbf{x})\cdot\mu_{B^r}(y) \\
&= \sum_{r=1}^{R}[\nu_r(\mathbf{x})\cdot(\sum_{j=1}^{N_b}\rho_{rj}\cdot\mu_{B_j}(y))]
\end{aligned}
\tag{5.20.a}
$$

or, by interchanging the first and the second summation notations, we have

$$
\mu_{B'}(y) = \sum_{j=1}^{N_b}[(\sum_{r=1}^{R}\rho_{rj}\cdot\nu_r(\mathbf{x}))\cdot\mu_{B_j}(y))]
\tag{5.20.b}
$$

Hence, analogous to definition (D.9), we define

(D.14) extended implication DOF, η_j, as

$$
\eta_j(\mathbf{x}) = \sum_{r=1}^{R}\rho_{rj}\cdot\nu_r(\mathbf{x})
$$

with property $\displaystyle\sum_{j=1}^{N_b}\eta_j(\mathbf{x}) = \sum_{j=1}^{N_b}(\sum_{r=1}^{R}\rho_{rj}\cdot\nu_r(\mathbf{x})) = \sum_{r=1}^{R}\nu_r(\mathbf{x})\cdot(\sum_{j=1}^{N_b}\rho_{rj}) = \sum_{r=1}^{R}\nu_r(\mathbf{x})$

Note, the term, $\sum_{j=1}^{N_b}\rho_{rj}=1$, is from the constraint (C.2). Hence, Eq.(5.20.b) can be rewritten as

$$
\mu_{B'}(y) = \sum_{j=1}^{N_b}\eta_j(\mathbf{x})\cdot\mu_{B_j}(y)
\tag{5.21}
$$

The normalization of η_j is defined next.

(D.15) normalized extended implication DOF, $\overline{\eta}_j$, is given by

$$
\overline{\eta}_j(\mathbf{x}) = \frac{\eta_j(\mathbf{x})}{\sum_{r=1}^{R}\nu_r(\mathbf{x})} = \frac{\sum_{r=1}^{R}\rho_{rj}\cdot\nu_r(\mathbf{x})}{\sum_{r=1}^{R}\nu_r(\mathbf{x})} = \sum_{r=1}^{R}\rho_{rj}\cdot(\frac{\nu_r(\mathbf{x})}{\sum_{r=1}^{R}\nu_r(\mathbf{x})}) = \sum_{r=1}^{R}\rho_{rj}\cdot\overline{\nu}_r(\mathbf{x})
$$

with property $\displaystyle\sum_{j=1}^{N_b}\overline{\eta}_j(\mathbf{x}) = \sum_{j=1}^{N_b}(\sum_{r=1}^{R}\rho_{rj}\cdot\overline{\nu}_r(\mathbf{x})) = \sum_{r=1}^{R}\overline{\nu}_r(\mathbf{x})\cdot(\sum_{r=1}^{R}\rho_{rj}) = \sum_{r=1}^{R}\overline{\nu}_r(\mathbf{x}) = 1$

Remark:

(R.8) Again, the extended Mamdani fuzzy rule can become equal to the ordinary Mamdani fuzzy rule when ρ_{rj} is chosen from $\{0,1\}$. I.e., if there exists a $r_j^* \in C_j$ such that

$$\rho_{rj} = \begin{cases} 1 \ ; \ r = r_j^* \in C_j \\ 0 \ ; \ r \in \overline{C}_j \end{cases}$$

then η_j is equal to β_j, since

$$\eta_j(x) = \sum_{r=1}^{R} \rho_{rj} \cdot \nu_r(x) = \sum_{r_j^*=1}^{N_{cj}} \rho_{r_j^* j} \cdot \nu_{r_j^*}(x) = \sum_{r_j^*=1}^{N_{cj}} \nu_{r_j^*}(x) = \beta_j(x)$$

Therefore, $\overline{\eta}_j$ becomes $\overline{\beta}_j$

$$\overline{\eta}_j(x) = \frac{\eta_j(x)}{\sum_{r=1}^{R} \nu_r(x)} = \frac{\beta_j(x)}{\sum_{r=1}^{R} \nu_r(x)} = \overline{\beta}_j(x)$$

It is obvious that definitions (D.14) and (D.15) of the extended Mamdani fuzzy rule are analogous to definitions (D.9) and (D.12) of the ordinary Mamdani fuzzy rule.

With Eq.(5.21) and definitions (D.10) and (D.14), the resultant active area, a', is derived as

$$\begin{aligned} a' &= \int_Y \mu_{B'}(y)dy = \int_Y (\sum_{j=1}^{N_b} \eta_j(x) \cdot \mu_{B_j}(y))dy = \sum_{j=1}^{N_b} \eta_j(x) \cdot (\int_Y \mu_{B_j}(y)dy) \\ &= \sum_{j=1}^{N_b} \eta_j(x) \cdot (\int_{Y_j} \mu_{B_j}(y)dy) = \sum_{j=1}^{N_b} \eta_j(x) \cdot a_j \\ &= \sum_{j=1}^{N_b} [(\sum_{r=1}^{R} \rho_{rj} \cdot \nu_r(x)) \cdot a_j] \end{aligned} \tag{5.22.a}$$

The first moment of B'(y), m', is then

$$
\begin{aligned}
m' &= \sum_{j=1}^{N_b} \eta_j(\mathbf{x}) \cdot a_j \cdot z_j \\
&= \sum_{j=1}^{N_b} [(\sum_{r=1}^{R} \rho_{rj} \cdot \nu_r(\mathbf{x})) \cdot a_j \cdot z_j]
\end{aligned}
\tag{5.23.a}
$$

Alternatively, according to Eq.(5.20.a) and definitions (D.10), a' and m' can be written as

$$
\begin{aligned}
a' &= \int_Y \mu_{B'}(y) dy = \int_Y (\sum_{r=1}^{R} [\nu_r(\mathbf{x}) \cdot (\sum_{j=1}^{N_b} \rho_{rj} \cdot \mu_{B_j}(y))]) dy \\
&= \sum_{r=1}^{R} [\nu_r(\mathbf{x}) \cdot (\sum_{j=1}^{N_b} \rho_{rj} \cdot (\int_{Y_j} \mu_{B_j}(y) dy))] \\
&= \sum_{r=1}^{R} [\nu_r(\mathbf{x}) \cdot (\sum_{j=1}^{N_b} \rho_{rj} \cdot a_j)]
\end{aligned}
\tag{5.22.b}
$$

and

$$
m' = \sum_{r=1}^{R} [\nu_r(\mathbf{x}) \cdot (\sum_{j=1}^{N_b} \rho_{rj} \cdot a_j \cdot z_j)]
\tag{5.23.b}
$$

Hence, the defuzzified output of the extended Mamdani fuzzy rules, denoted as $y_{(EM)}$, can be obtained from Eq.(5.22) and Eq.(5.23), based on N_b output fuzzy sets,

$$
y_{(EM)}(\mathbf{x}) = \frac{m'}{a'} = \frac{\sum_{j=1}^{N_b} \eta_j(\mathbf{x}) \cdot a_j \cdot z_j}{\sum_{j=1}^{N_b} \eta_j(\mathbf{x}) \cdot a_j}
\tag{5.24.a}
$$

or, based on R fuzzy rules,

$$
y_{(EM)}(\mathbf{x}) = \frac{m'}{a'} = \frac{\sum_{r=1}^{R} [\nu_r(\mathbf{x}) \cdot (\sum_{j=1}^{N_b} \rho_{rj} \cdot a_j \cdot z_j)]}{\sum_{r=1}^{R} [\nu_r(\mathbf{x}) \cdot (\sum_{j=1}^{N_b} \rho_{rj} \cdot a_j)]}
\tag{5.24.b}
$$

Again, further simplification can be done according to assumption (A.3); i.e., the area of each output fuzzy set is identical, $a_1 = .. = a_j = .. = a_{Nb}$. Using the property of (D.14), $\sum_{j=1}^{N_b} \eta_j = \sum_{r=1}^{R} \nu_r$, and (D.15), Eq.(5.24.a) then becomes

$$y_{(EM)}(\mathbf{x}) = \frac{\sum_{j=1}^{N_b} \eta_j(\mathbf{x}) \cdot z_j}{\sum_{j=1}^{N_b} \eta_j(\mathbf{x})} = \frac{\sum_{j=1}^{N_b} \eta_j(\mathbf{x}) \cdot z_j}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} = \sum_{j=1}^{N_b} \left(\frac{\eta_j(\mathbf{x})}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} \right) \cdot z_j$$
$$= \sum_{j=1}^{N_b} \overline{\eta}_j(\mathbf{x}) \cdot z_j \qquad \left(\text{or} = \sum_{j=1}^{N_b} \left[\left(\sum_{r=1}^{R} \rho_{rj} \cdot \overline{\nu}_r(\mathbf{x}) \right) \cdot z_j \right] \right) \tag{5.25.a}$$

Remark:

(R.9) When the rule significance level ρ_{rj} is chosen as $\{0,1\}$, $\overline{\eta}_j = \overline{\beta}_j$, from the previous remark (R.8) means that the defuzzified output of the extended Mamdani fuzzy rules (Eq.(5.25.a)) is identical to that of the ordinary Mamdani fuzzy rules (Eq.(5.16.b)), $y_{(EM)} = y_{(M)}$. This shows that, under the assumption (A.3), $a_1 = .. = a_j =.. = a_{Nb}$, the ordinary Mamdani fuzzy rule is, again, a special case of the extended Mamdani fuzzy rule.

Similarly, with the same assumption of $a_1 = .. = a_j =.. = a_{Nb}$, and constraint (C.2), $\sum_{j=1}^{N_b} \rho_{rj} = 1$, Eq.(5.24.b) becomes

$$y_{(EM)}(\mathbf{x}) = \frac{\sum_{r=1}^{R} \left[\nu_r(\mathbf{x}) \cdot \left(\sum_{j=1}^{N_b} \rho_{rj} \cdot z_j \right) \right]}{\sum_{r=1}^{R} \left[\nu_r(\mathbf{x}) \cdot \left(\sum_{j=1}^{N_b} \rho_{rj} \right) \right]} = \frac{\sum_{r=1}^{R} \left[\nu_r(\mathbf{x}) \cdot \left(\sum_{j=1}^{N_b} \rho_{rj} \cdot z_j \right) \right]}{\sum_{r=1}^{R} \nu_r(\mathbf{x})}$$
$$= \sum_{r=1}^{R} \left[\frac{\nu_r(\mathbf{x})}{\sum_{r=1}^{R} \nu_r(\mathbf{x})} \cdot \left(\sum_{j=1}^{N_b} \rho_{rj} \cdot z_j \right) \right] \tag{5.25.b}$$
$$= \sum_{r=1}^{R} \left[\overline{\nu}_r(\mathbf{x}) \cdot \left(\sum_{j=1}^{N_b} \rho_{rj} \cdot z_j \right) \right] \qquad \left(\text{or} = \sum_{j=1}^{N_b} \left[\left(\sum_{r=1}^{R} \rho_{rj} \cdot \overline{\nu}_r(\mathbf{x}) \right) \cdot z_j \right] \right)$$

5.4 INTERPRETATION OF THE IDENTIFIED T-S FUZZY RULES BY EM FUZZY RULES

From the previous section, one can see that the defuzzified output of the extended Mamdani fuzzy model can be represented either by Eq.(5.25.a), or by Eq.(5.25.b), where the two summation terms indexed by r and j are interchangeable without affecting the results. This allows us to make an easy comparison to that of the T-S fuzzy model, since the defuzzified output of both models are based on R fuzzy rules. In this section, we will consider how to link the T-S fuzzy rule and the extended Mamdani fuzzy rule, and how to solve for the consequent significance level parameters.

5.4.1 To link the T-S fuzzy rule and the extended Mamdani fuzzy rule

As mentioned in remark (R.4), the property of dual representation of the fuzzy system may lead to a linkage of the T-S fuzzy rule to the extended Mamdani fuzzy rules. I.e., the resultant output of the Mamdani fuzzy model $y_{(EM)}$, Eq.(5.25.b), is comparative to the output of the zeroth-order T-S fuzzy model $y_{(TS)}$, Eq.(5.3.b), if the following equality relation holds.

$$w_r \cong \sum_{j=1}^{N_b} \rho_{rj} \cdot z_j \quad ; \forall\, r = 1, \ldots, R \tag{5.26}$$

Eq.(5.26) reveals that the consequent of the zeroth-order T-S fuzzy rule, w_r, can be transformed into a combination of some fuzzy sets $B_j(y)$'s of the extended Mamdani fuzzy rule, that are centered at z_j's and are associated with ρ_{rj}'s. If this transformation is achievable, then the consequent part of the zeroth-order T-S fuzzy rule, which is represented by real numbers, can be interpreted linguistically in the same way as the extended Mamdani fuzzy model does with the extended Mamdani type of fuzzy rule, Eq.(5.18). Hence, knowledge interpretation of the zeroth-order T-S fuzzy rule in a linguistic way becomes feasible.

5.4.2 To solve the parameters w and ρ

It can be verified from Eq.(5.26) that the value of w_r should be bounded on some reasonable domain of Y that may be characterized by the minimal (z_{min}) and maximal (z_{max}) values in order to obtain a meaningful transformation. Practically, these bounded domains can be determined from the predefined z_j's values either by the users or from the observation data available. Hence, to solve the parameter values of w_r and ρ_{rj}, a method based on constrained optimal searching is proposed and described below.

First, the users have to define the possible output domain of Y bounded on $[z_{min}, z_{max}]$ and the number of output fuzzy sets $B_j(y)$'s, N_b, as well as their corresponding centers z_j's. In practice, this is not a difficult task, since they can be estimated either from experimental data, or from the expert's experience. Second, the optimal values of w_r's, denoted by a vector $\mathbf{w}^* = [w_1^*\ w_2^*\ ..\ w_r^*\ ..\ w_R^*]^T$, are searched to minimize the squared error between the real system output and the model output,

$$\mathbf{w}^* = \arg \min_{\forall w_r \in \Omega_w} \left\{ \sum_{t=1}^{np} \left[y_d(t) - \sum_{r=1}^{N_r} \bar{v}_r(\mathbf{x}(t)) \cdot w_r \right]^2 \right\} \tag{5.27}$$

where, $\Omega_w = [z_{min}, z_{max}]$, the bounded output domain; t = 1, .., np, the number of available data set; and y_d, the desired value of the system. So, this means that in contrast to the ordinary T-S identification, its parameters **w** should be constrained in order to make the two models comparable.

Once all w_r's are obtained from Eq.(5.27), then the optimal values of ρ_{rj}'s, $\rho_r = [\rho_{r1}\ \rho_{r2}\ ..\ \rho_{rj}\ ..\ \rho_{rNb}]^T$, for r = 1, .. , R, can be found by minimizing an objective function deduced from Eq.(5.26),

$$\rho_r = \arg \min \left[w_r^* - \sum_{j=1}^{Nb} \rho_{rj} \cdot z_j \right]^2 \qquad \text{for } r = 1, .., R \tag{5.28}$$

with all solutions are subject to constraints (C.1) - (C.3). Hence, when we have solved the constrained consequent weight $\mathbf{w}^*$ of a zeroth-order T-S fuzzy model using Eq.(5.27), we can further search the values of the consequent significance level that satisfy constraints (C.1) - (C.3) by Eq.(5.28). Making use of these consequent significance level parameters, the zeroth-order T-S fuzzy model is thus transferred to the extended Mamdani fuzzy model and can be interpreted accordingly.

5.5 EXAMPLE

This section presents a synthetic example to illustrate how to interpret the identified T-S fuzzy model in the sense of the extended Mamdani fuzzy model. The example is a nonlinear single-input-single-output system, described by

$$y = 1 - x^2 \tag{5.29}$$

First we generate 41 sets of data pairs $[x(t)\ y_d(t)]$ for training the fuzzy model, where t runs from 1 to 41 and input x is randomly picked from the interval [-1, 1]. The desired value y_d

generated according to Eq.(5.29) is contaminated by some noise. Five triangular membership functions with equal center distance are set for the input x and denoted by linguistic terms 'NM', 'NS', 'ZO', 'PS', and 'PM', that stand for negative medium, negative small, zero, positive small, and positive medium, respectively (see Figure 5.1.(a)). Accordingly these five membership functions construct an initial fuzzy rule base with five rules.

Three fuzzy sets $[B_{small}(y)\ B_{medium}(y)\ B_{big}(y)]$ are defined based on the training data. These correspond to linguistic terms 'S', 'M', and 'B' and centers at $\mathbf{z}$ (= $[z_1\ z_2\ z_3]^T = [0.0278\ \ 0.5217\ \ 1.0156]^T$), respectively. Next, a set of bounded consequent weight $\mathbf{w}^*$ of the zeroth-order T-S fuzzy model is obtained by Eq.(5.27). They are identified as $\mathbf{w}^* = [0.0278\ \ 0.7852\ \ 1.0156\ \ 0.8216\ \ 0.0454]^T$. All weights are located in the observed output domain from the training data $\{\min(y_d), \max(y_d)\}$. So, the five fuzzy rules of the identified zeroth-order T-S fuzzy model are read as

$R^1_{(TS)}$:	***IF*** x is NM	***THEN*** $y = w_1^*$ (= 0.0278)
$R^2_{(TS)}$:	***IF*** x is NS	***THEN*** $y = w_2^*$ (= 0.7852)
$R^3_{(TS)}$:	***IF*** x is ZO	***THEN*** $y = w_3^*$ (= 1.0156)
$R^4_{(TS)}$:	***IF*** x is PS	***THEN*** $y = w_4^*$ (= 0.8216)
$R^5_{(TS)}$:	***IF*** x is PM	***THEN*** $y = w_5^*$ (= 0.0454)

Second, the values of ρ_{rj}'s are obtained according to Eq.(5.28), subject to the constraints (C.1) - (C.3), giving the following values.

	$z_{1(0.0278)}$	$z_{2(0.5217)}$	$z_{3(1.0156)}$	
(rule 1)	1.0000	0.0000	0.0000	($w_1^* = 0.0278$)
(rule 2)	0.0028	0.4609	0.5363	($w_2^* = 0.7852$)
(rule 3)	0.0000	0.0000	1.0000	($w_3^* = 1.0156$)
(rule 4)	0.0983	0.1960	0.7057	($w_4^* = 0.8216$)
(rule 5)	0.9646	0.0353	0.0001	($w_5^* = 0.0454$)

Hence, as an example, the second zeroth-order T-S fuzzy rule can be transformed to an extended Mamdani fuzzy rule to read as

IF x is NS ***THEN*** {y is S with $\rho_{21} = 0.0028$, and
y is M with $\rho_{22} = 0.4609$, and
y is B with $\rho_{23} = 0.5363$}

The above results show that it is possible to transform the zeroth-order T-S fuzzy model (Eq.(5.3.b)) to the extended Mamdani fuzzy model (Eq.(5.18)). Figure5.1.(b) shows the result of the extended Mamdani fuzzy model, $y_{(EM)}$.

An important remark should be made with respect to the transformation. The consequent weight parameters of the T-S fuzzy model are optimal, i.e., they minimize the error between data and model in the least squares sense. This means that the EM fuzzy model is always somewhat less accurate. In addition, if the weights **w** of the unmodified T-S fuzzy model are not contained in the interval $[z_{min}, z_{max}]$, direct comparison is possible only by constraining the T-S parameters according to Eq.(5.27). This again reduces the mapping accuracy. So, one could say that the gain of transparency by transforming the T-S fuzzy model into the extended Mamdani form goes at the expense of some loss in model accuracy.

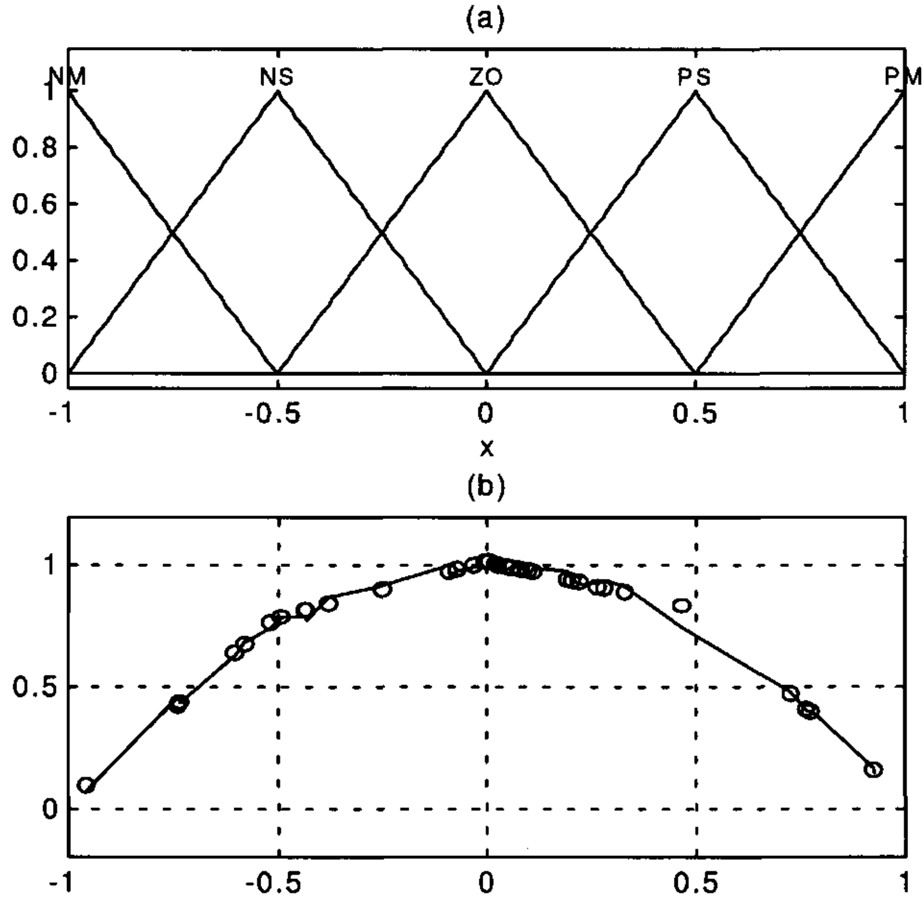

Figure 5.1: (a) membership function of input x. (b) The desired output y_d (circle) and extended Mamdani fuzzy model output $y_{(EM)}$ (solid line).

5.6 CONCLUSION AND DISCUSSION

In this chapter we have analyzed two different types of fuzzy rules, namely the Takagi-Sugeno type of fuzzy rule and the Mamdani type of fuzzy rule. From the interpretation point of view, the T-S fuzzy model with T-S fuzzy rules are less interpretable than the fuzzy model with ordinary Mamdani fuzzy rules. Although it is easy to understand the fuzzy rules of the Mamdani fuzzy model, the modeling flexibility is limited due to the single output fuzzy set in the rule consequent. In order to relax the dependency on the definition of the output fuzzy set of the Mamdani fuzzy model, we introduce a new parameter, namely the consequent significance level, that is associated to the consequent part of each Mamdani type of fuzzy rule. As a result, an extended Mamdani fuzzy rule base can be established, which is characterized by multiple output fuzzy sets in its consequent.

It is shown that the ordinary Mamdani fuzzy rule is a special class of the extended Mamdani fuzzy rule with a consequent significance level parameter equals to either unity or zero. Furthermore, we have analyzed the possibility of deducing linguistic interpretation from an identified T-S fuzzy model. The result shows that the zeroth-order T-S fuzzy rule base can be transformed to be equivalent to an extended Mamdani fuzzy rule base under some certain not too restrictive conditions. Therefore, a zeroth-order T-S fuzzy rule can be interpreted in the same way as an extended Mamdani fuzzy rule via the consequent significance level parameters. As consequent significant level parameters of the extended Mamdani fuzzy model are subject to some constraints, it takes more effort to identify these parameters than the consequent weight parameters of the zeroth-order T-S fuzzy model. However, what one can gain is that the linguistic interpretation of the zeroth-order T-S fuzzy rules becomes possible, yet, perhaps at the expense of some loss in model accuracy. It should be noted, however, that it is not necessary try to get linguistic interpretation from the T-S fuzzy model in case the model is merely used for function approximation.

It is also worthwhile to address a possible problem which may arise in the above transformation. Since the transformation of w to the linear combination of ρ and z is a one to many mapping, the uniqueness of the inverse relation is not always guaranteed by just complying to constraints (C.1) - (C.3). Apart from the uniqueness problem, how one can have a reasonable interpretation of these ρ values is another concern. Consider the following two situations, when

(1) all elements of the obtained ρ_r vector, which is formed by $[\rho_{r1} \,..\, \rho_{rj} \,..\, \rho_{rNb}]^T$ with respect to the r^{th} fuzzy rule, have the same mean values, i.e., $1/N_b$;

or

(2) any two non-consecutive elements of vector ρ_r have identical values of ρ_{rj_1} and ρ_{rj_2}, where j_1 and j_2 indicate two indices of the corresponding fuzzy sets not consecutive to each other;

then the interpretation of these ρ values becomes equivocal. Hence, further constraints shall be taken into account to avoid the above problematic interpretation. One of the possible solutions is that an extra constraint shall be satisfied when the optimization is carried out.

(C.4) elements of ρ_r (= $[\rho_{r1} .. \rho_{rj} .. \rho_{rNb}]^T$) are convex in the sequence. This means that if the relation holds for $j_1 < j_2 < j_3$ and $j_1, j_2, j_3 \in \{1, 2,.., N_b\}$, then $\rho_{rj2} > \min(\rho_{rj1}, \rho_{rj3})$.

Complying to this constraint implies that the values of sequential elements in the row vector ρ_r should be either monotonously increasing or decreasing, or formed as a bell-shaped distribution. Another way to get reasonable values of ρ can be found in [5] , where B-spline functions with different orders are suggested to represent the membership functions of output variables, because B-spline functions naturally fulfill the requirements of assumptions (A.1) - (A.3) and constraints (C.1) - (C.3). (Therefore, this will result in a faster and easier estimation of the corresponding ρ_{rj} values, provided that the identified w_r is bounded in the predefined domain of these B-spline functions.) In constrast, the present analysis shows that any possible general function can be used as the membership functions of the outputs; as long as they satisfy the requirements of assumptions (A.1) - (A.3) and constraints (C.1) - (C.4). The B-spline functions can therefore be regarded as a special of those general functions.

6. INCORPORATING *A PRIORI* KNOWLEDGE INTO T-S FUZZY MODELING

Love is patient, love is kind. It does not envy, it does not boast, it is not proud. It is not rude, it is not self-seeking, it is not easily angered, it keeps no record of wrongs. Love does not delight in evil but rejoices with the truth. It always protects, always trusts, always hopes, always perseveres. Love never fails.

- 1 Corinthians 13, Bible

愛是恆久忍耐，又有恩慈，愛是不嫉妒。愛是不自誇，不張狂，不作害羞的事。不求自己的益處，不輕易發怒。不計算別人的惡，不喜歡不義，只喜歡真理。凡事包容，凡事相信，凡事盼望，凡事忍耐。愛是永不止息。

聖經哥林多前書第十三章

6.1 INTRODUCTION

In this chapter, we will consider how to incorporate available *a priori* knowledge from different sources into the fuzzy modeling problem.

Following the terminology from computer science we can distinguish two approaches, which are 'Case-Based Reasoning' (CBR) and 'Rule-Based Reasoning' (RBR). These approaches use different knowledge sources to construct a 'Case-Based Model' (CBM) or a 'Rule-Based Model' (RBM), respectively. It turns out that 'Case' refers to the identification from the observation of the input/output relation of a system, i.e., a single piece of knowledge based on direct observation. On the other hand, 'Rule' implies the collection of the above mentioned observations to form a set of rules that describe the fundamental behavior of a system in the observed domain. These rules can be obtained by some generalization processes by human beings or by machine learning. The so-called *a priori* knowledge refers to a collection of rules codified by experts over the years and converted as experience. The very use of *a priori* knowledge in modeling results in a hybrid model in the sense that it combines both case-based and rule-based information. Generally, the correct use of *a priori* knowledge in modeling will lead to a better model that can stand up against a deficient or incomplete data set. Hence, in addition to the improvement of model performance, our primary motivation of merging *a priori* knowledge and data-driven modeling is to see whether the extrapolation ability can be improved in such a hybrid approach. Thus, the goal of this chapter is to combine the CBM and RBM into a uniform representation to improve efficiency and accuracy of the hybrid model [19] , as well as to achieve extrapolation ability.

Since *a priori* knowledge is a compact representation of accumulated experience it can often be presented in the following form : '*IF* some conditions are met *THEN* the corresponding reactions reply'. The '*IF* .. *THEN* .. ' statement, that is characterized more by qualitative than by quantitative information, is the most suitable candidate in fuzzy modeling, as qualitative knowledge with the same foundation as the fuzzy rule base can be incorporated. Although many successful applications of fuzzy logic control have been reported in the past decade, there are rare studies on fuzzy modeling which address how to utilize the qualitative information from the literature in a systematic way. The difficulty is due to the fact that the use of qualitative information in fuzzy modeling is very problem-dependent and usually *ad hoc*. Hence, our present attempt focuses on how to incorporate *a priori* qualitative information into data-driven fuzzy modeling; in particular, the model that is based on the zeroth-order Takagi-Sugeno fuzzy model.

Identification of a T-S fuzzy model is classified in [61] as structure identification and parameter identification. In the issue of structure identification, one has to find the input variables, i.e., how to select the input candidates and to determine the suitable input variables. One also has to construct the input-output relation, i.e., determine the fuzzy rules and partition of input space. The structure identification problem is deemed as the most difficult since it takes much more effort than parameter identification. Once the structure is identified, the identification of parameters can easily be attained. In what follows, we will only pay attention to the zeroth-order T-S fuzzy model. It is assumed that the input candidates and the suitable input variables are determined based on some *a priori* knowledge. Furthermore, the construction of the fuzzy rules is determined from some heuristics. Beyond the piece of *a priori* knowledge used for the formulation of the structure of the T-S fuzzy model, other types of qualitative knowledge can be incorporated into the modeling. It turns out that the identification problem then becomes a parameter estimation problem. Before we move to the next section, it is essential to assume that, in case conflicts arise between the observation data and the *a priori* knowledge in the region of interpolation, precedence will always be given to the observation data. Moreover, one has to rely on the *a priori* knowledge used in the extrapolation region where the observation data are lacking.

The chapter is organized as follows. Section 6.2 will introduce the optimization approach on which the incorporation of *a priori* knowledge into a T-S fuzzy model is based. Moreover, the formulation of the performance criterion for optimization and the estimation procedure of the parameters will be addressed. Section 6.3 offers a simple example to demonstrate the proposed optimization approach. Fairly convincing results are given in section 6.4. Finally, conclusions are made in section 6.5.

6.2 OPTIMIZATION BASED APPROACH

We have seen that the zeroth-order T-S model is similar to a neural network performing nonlinear mapping of input space to output space, like the NUFZY model. It is noticed that in the T-S fuzzy model less *a priori* knowledge can be incorporated than in Mamdani fuzzy model. If any available *a priori* knowledge is imposed to the T-S fuzzy modeling, the resultant model will be regulated to comply with the required properties described by the imposed *a priori* knowledge. Hence, the idea of incorporating qualitative information into the T-S fuzzy model is to regard the different pieces of *a priori* qualitative information as soft constraints or penalty terms that are imposed to a performance criterion, which is to be minimized. This idea was inspired by [73] who discussed how to use different sources of knowledge to construct the model and optimization problem, followed by an unifying framework proposed by [34] as an optimization formulation of the modeling problem.

In this section, we will refer to [34] . The author uses an optimization approach to incorporate qualitative information into the T-S fuzzy model. For simplicity, in this work we assume the structure of a zeroth-order T-S fuzzy model is determined and fuzzy rules of this T-S fuzzy model will not change. For identification purposes, a batch of np input-output observations for training and nv observations for validation are available. Furthermore, we also have available a default model that contains possibly imprecise qualitative information. In particular, we are interested in a default model described by a Mamdani fuzzy model with a set of Mamdani type of fuzzy rules. An optimization problem can then be formed based on the empirical data and the *a priori* qualitative information, allowing us to seek the optimal consequent weight parameters of the T-S fuzzy model in the parameter space. The performance criterion for optimization will be penalized by the following conditions,

(1) Mismatch between the model prediction of the T-S fuzzy model and the observation data.
(2) Non-smoothness of the T-S fuzzy model*.
(3) Violation of the soft constraints.
(4) Mismatch between T-S fuzzy model and the default model.

For the purpose of prediction, it is noticed that, in general, a performance criterion merely based on condition (1) is not sufficient to achieve model performance beyond the empirical data. A modified performance criterion based on condition (1) and (2) has been studied either as an effect of regularization† by the neural networks community [17] [18] [52] [53] [54] , or as a ridge regression problem by statisticians [14] [47] . It has been shown that the regularization method allows a smooth interpretation of the model. In this study the zeroth-order regularization is adopted because of its simplicity.

If a Mamdani fuzzy model is available, one can easily get some qualitative description from its fuzzy rule base to describe the system behavior. Some of these qualitative descriptions can be regarded as soft constraints imposed on the T-S fuzzy model for identifying the system. They might include steady-state description of a dynamic system, or specifically known input-output relations around the operation points gathered from the accumulated experience of the users. Hence, condition (3) accounts for the penalty by violating these soft constraints. Condition (4) can be seen as a measure of discrepancy between the T-S fuzzy model and an existing default model. This default model may contain partially imprecise information based on some *a priori* knowledge. For instance, a Mamdani type of fuzzy model provides a basic model that can be used in regions where no observation data are available. When the default model is taken into account for modeling, the operation range of such a model is expected to be as large as possible to cover all operating conditions of the identified system. Next, we will define the performance criterion used for this optimization approach, which may include all the above conditions as well as some inequality constraints (hard constraints) that regulate the range of the consequent weight parameter values of the zeroth-order T-S model.

* The real system under study is, in fact, assumed to be smooth. The smoothness can be defined as the existence and continuity of some sufficiently high-order derivatives of the system [34] .

† Regularization is originally used to avoid the occurrence of an ill-condition in an information matrix of a model. The method used here, referred to as zeroth-order regularization or equivalently, ridge regression, is to add a small positive quantity to the diagonal of the ill-conditioned matrix, so that the matrix remains positive definite, and the determinant of the matrix increases or the elements of the inverse decrease. In other words, the added quantity introduces a bias term to its mean-squared-error evaluation, which favors solutions involving small absolute parameter values. As a result the output of a function becomes less sensitive to the variation of parameters, or is smooth. Other methods that are perhaps more effective in achieving the smoothness property, such as second-order regulariztion based on curvature, are more complicated.

6.2.1 Formulation of the performance criterion

As stated earlier, we assume that we have established the structure of a MISO zeroth-order T-S fuzzy model containing R fuzzy rules. Following from previous chapters, the predicted output of this specifically defined T-S fuzzy model is given by the linear equation

$$y_{TS}(\mathbf{x}(t)) = \overline{\mathbf{V}}(\mathbf{x}(t)) \cdot \mathbf{w} \tag{6.1}$$

where input $\mathbf{x}(t)$(or, $\mathbf{x} \in \mathbb{R}^{ni}$) and normalized firing strength $\overline{\mathbf{V}}(\mathbf{x}(t))$ (or, $\overline{\mathbf{V}}(\mathbf{x}) \in \mathbb{R}^{np \times R}$) are indexed by the sample instance, t, for t = 1, .. , np. The consequent weight vector of this T-S fuzzy model is denoted as $\mathbf{w}$ ($\in \mathbb{R}^{R}$).

In addition, assume that there is a default model related to the system under study described by a Mamdani fuzzy model. Given na sets of input $\mathbf{x}_a(k)$(or, $\mathbf{x}_a \in \mathbb{R}^{ni}$) chosen from the possible operation range, one can obtain the corresponding output $y_a(k)$ from this default Mamdani fuzzy model, where index k = 1, .. , na. Similarly, with this chosen input $\mathbf{x}_a$ on hand, we can also calculate the corresponding normalized firing strength matrix $\overline{\mathbf{V}}_a(\mathbf{x}_a)$ (or, $\overline{\mathbf{V}}_a \in \mathbb{R}^{na \times R}$) and its prediction output, $\overline{\mathbf{V}}_a \mathbf{w}$, by the T-S fuzzy model, so that the discrepancy between the T-S fuzzy model and the default Mamdani fuzzy model can be evaluated. Note, the chosen input $\mathbf{x}_a$ is different from the real observed input $\mathbf{x}$. Since $\mathbf{x}_a$ is artificially chosen and generated for the default model, it can be regarded as excitation signals that cover the most possible operation range of the system; especially, in regions where the observation data are not available.

Some soft constraints describing specific characteristics of the system may be available, which come from the real additional information, like steady state points; time averages; long-term off-sets and trends, etc. Hence, a set of ns rules is likely to be found.

$R^q_{(s)}$: *IF* $\mathbf{x}$ is $\mathbf{A}^q(\mathbf{x}_s)$ *THEN* y is $\mathbf{B}^q(y_s)$

where rule index q = 1, .. , ns. Fuzzy set $\mathbf{A}^q(\mathbf{x}_s)$ characterizes the specific input state $\mathbf{x}_s$, and $\mathbf{B}^q(y_s)$ represents the consequent output fuzzy set characterizing the specific output y_s. Both fuzzy sets correspond to the qth specific fuzzy rule. Specifically, we can read the above special rules as following,

$R^q_{(s)}$: *IF* $\mathbf{x}$ is close to state '$\mathbf{x}_s$' *THEN* y is close to system response 'y_s' (6.2)

where $\mathbf{x}_s$ represents some known operating points or steady states of a dynamic system provided it exists, and y_s is its corresponding output or system response to the specific input $\mathbf{x}_s$. Associating these ns specific fuzzy rules to constrain the T-S fuzzy model, one can expect that it shall possess the expected behaviors complying to the imposed *a priori* knowledge. With this limited number of input $\mathbf{x}_s$($\in \mathbb{R}^{ni}$) one can calculate the corresponding normalized firing strength matrix $\overline{\mathbf{V}}_s(\mathbf{x}_s)$ (or, $\overline{\mathbf{V}}_s \in \mathbb{R}^{ns \times R}$) by the T-S fuzzy model and its prediction output, $\overline{\mathbf{V}}_s \mathbf{w}$. Again, the discrepancy, representing the violation of soft constraints, between the T-S fuzzy model and the specific system response, can be evaluated.

There are different forms in which knowledge from various sources can be incorporated into a unified optimization problem. For demonstration, we will integrate all the above qualitative information together with the requirement of smoothness in a performance criterion, J, defined by

$$J(\mathbf{w};\lambda,\beta,\alpha)=\frac{1}{np}\sum_{t=1}^{np}[y_d(t)-\overline{\mathbf{v}}_t(\mathbf{x}(t))\mathbf{w}]^2+\frac{1}{np}\lambda\sum_{r=1}^{R}w_r^2 \\ +\beta\left[\frac{1}{ns}\sum_{q=1}^{ns}[y_s(q)-\overline{\mathbf{v}}_q(\mathbf{x}_s(q))\mathbf{w}]^2\right]+\alpha\left[\frac{1}{na}\sum_{k=1}^{na}[y_a(k)-\overline{\mathbf{v}}_k(\mathbf{x}_a(k))\mathbf{w}]^2\right] \tag{6.3.a}$$

where row vectors $\overline{\mathbf{v}}_t$, $\overline{\mathbf{v}}_q$, and $\overline{\mathbf{v}}_k$ (all $\in \mathbb{R}^R$) represent the t^{th}, q^{th} and k^{th} row of normalized firing strength matrices $\overline{\mathbf{V}}(\mathbf{x})$, $\overline{\mathbf{V}}_s(\mathbf{x}_s)$, and $\overline{\mathbf{V}}_a(\mathbf{x}_a)$, respectively. $y_d(t)$, the desired output of the t^{th} training set, $y_s(q)$, the specific output of the q^{th} constrained fuzzy rule, and $y_a(k)$, the k^{th} output of the default Mamdani fuzzy model, correspond to the inputs of $\mathbf{x}(t)$, $\mathbf{x}_s(q)$, and $\mathbf{x}_a(k)$, respectively. w_r is the consequent weight of the r^{th} T-S fuzzy rule among the R fuzzy rules. np, ns and na are the corresponding number of input patterns. Penalty weighting parameters λ, β, and α account for the above penalty conditions (2), (3) and (4), respectively. In a compact vector-matrix format, can be expressed as

$$J(\mathbf{w};\lambda,\beta,\alpha) = [\mathbf{y}_d - \overline{\mathbf{V}}(\mathbf{x})\mathbf{w}]^T[\mathbf{y}_d - \overline{\mathbf{V}}(\mathbf{x})\mathbf{w}]/np + \lambda\, \mathbf{w}^T\mathbf{w}/np \\ + \beta[\mathbf{y}_s - \overline{\mathbf{V}}_s(\mathbf{x}_s)\mathbf{w}]^T[\mathbf{y}_s - \overline{\mathbf{V}}_s(\mathbf{x}_s)\mathbf{w}]/ns + \alpha[\mathbf{y}_a - \overline{\mathbf{V}}_a(\mathbf{x}_a)\mathbf{w}]^T[\mathbf{y}_a - \overline{\mathbf{V}}_a(\mathbf{x}_a)\mathbf{w})/na \tag{6.3.b}$$

where $\mathbf{y}_d$, $\mathbf{y}_s$, and $\mathbf{y}_a$ are $\mathbb{R}^{np}$, $\mathbb{R}^{ns}$, $\mathbb{R}^{na}$, respectively; information matrices of the T-S fuzzy model $\overline{\mathbf{V}}\in\mathbb{R}^{np\times R}$, $\overline{\mathbf{V}}_s\in\mathbb{R}^{ns\times R}$, and $\overline{\mathbf{V}}_a\in\mathbb{R}^{na\times R}$ are normalized firing strength matrices using input $\mathbf{x}$, $\mathbf{x}_s$ and $\mathbf{x}_a$ as augments. Eq.(6.3.b) can be further expanded by a quadratic form as below*

$$J(\mathbf{w};\lambda,\beta,\alpha) = 1/2\cdot\mathbf{w}^T\,[2\overline{\mathbf{V}}^T\overline{\mathbf{V}}/np + 2\lambda\,\mathbf{I}_R/np + 2\beta\overline{\mathbf{V}}_s^T\overline{\mathbf{V}}_s/ns + 2\alpha\overline{\mathbf{V}}_a^T\overline{\mathbf{V}}_a/na]\,\mathbf{w} \\ - 2\,[\overline{\mathbf{V}}^T\mathbf{y}_d/np + \beta\overline{\mathbf{V}}_s^T\mathbf{y}_s/ns + \alpha\overline{\mathbf{V}}_a^T\mathbf{y}_a/na]^T\cdot\mathbf{w} \\ + [\mathbf{y}_d^T\mathbf{y}_d/np + \beta\,\mathbf{y}_s^T\mathbf{y}_s/ns + \alpha\,\mathbf{y}_d^T\mathbf{y}_d/na] \tag{6.3.c}$$

If we define ratios of np to ns and np to na , we have

$$\zeta = np/ns \tag{6.4.a}$$

$$\eta = np/na \tag{6.4.b}$$

* The purpose of this quadratic formulation is to be in line with the quadratic programming (**qp**) in Matlab®, which can be used to find the minimum of the performance criterion, J, provided the values of λ, β, and α are given.

Then, for any given values of penalty weighting parameters, λ, β, and α, the optimal consequent weight parameter of **w**, $\mathbf{w}^*_{LS}$, can be easily solved by the least squares method,

$$\mathbf{w}^*_{LS} = \mathbf{A}^{-1} \cdot (\overline{\mathbf{V}}^T \mathbf{y}_d + \zeta\, \beta\, \overline{\mathbf{V}}_s{}^T \mathbf{y}_s + \eta\, \alpha \overline{\mathbf{V}}_a{}^T \mathbf{y}_a) \tag{6.5}$$

where the covariance matrix $\mathbf{A} \in \mathbb{R}^{R \times R}$ containing penalty weighting parameters λ, β, and α, is defined by

$$\mathbf{A} = \overline{\mathbf{V}}^T \overline{\mathbf{V}} + \lambda\, \mathbf{I}_R + \zeta\, \beta\, \overline{\mathbf{V}}_s{}^T \overline{\mathbf{V}}_s + \eta\, \alpha \overline{\mathbf{V}}_a{}^T \overline{\mathbf{V}}_a \tag{6.6}$$

Note that the obtained $\mathbf{w}^*_{LS}$ by the least squares estimation is not subject to any constraint of **w**. In cases when these **w** values are expected to be located in some specific interval $[z_{min}, z_{max}]$ (for instance in chapter 5, if we want to have a meaningful interpretation of these **w** values in the output domain; also see [71]), then we can use the quadratic programming (for example, the subroutine of **qp** from Matlab®) to find the optimal **w** values. These values are obtained by minimizing the following quadratic criterion and are subject to inequality constraints

$$\mathbf{w}^*_{QP} = \{\mathbf{w} \mid \text{minimize } J(\mathbf{w}) = 1/2\, \mathbf{w}^T \mathbf{Q} \mathbf{w} + \mathbf{c}^T \mathbf{w} + \mathbf{Y}_0 \text{ and } \mathbf{B}\mathbf{w} \le \mathbf{b}\} \tag{6.7}$$

where

$$\mathbf{Q} = 2\, \overline{\mathbf{V}}^T \overline{\mathbf{V}} / np + 2\, \lambda\, \mathbf{I}_R / np + 2\, \beta\, \overline{\mathbf{V}}_s{}^T \overline{\mathbf{V}}_s / ns + 2\, \alpha \overline{\mathbf{V}}_a{}^T \overline{\mathbf{V}}_a / na \qquad (\in \mathbb{R}^{R \times R})$$

$$\mathbf{c} = -2\, (\overline{\mathbf{V}}^T \mathbf{y}_d / np + \beta\, \overline{\mathbf{V}}_s{}^T \mathbf{y}_s / ns + \alpha \overline{\mathbf{V}}_a{}^T \mathbf{y}_a / na) \qquad (\in \mathbb{R}^{R})$$

$$\mathbf{Y}_0 = \mathbf{y}_d{}^T \mathbf{y}_d / np + \beta\, \mathbf{y}_s{}^T \mathbf{y}_s / ns + \alpha\, \mathbf{y}_a{}^T \mathbf{y}_a / na \qquad (\in \mathbb{R})$$

$$\mathbf{B} = [\, -\mathbf{I}_R \,;\, \mathbf{I}_R \,] \qquad (\in \mathbb{R}^{2R \times R})$$

$$\mathbf{b} = [\mathbf{z}_{min} \,;\, \mathbf{z}_{max}] \qquad (\in \mathbb{R}^{2R})$$

Matrix **B** and vector **b** are used to restrict the searching of optimal **w**, in order to ensure that the searched optimal values of **w** belong to the desired interval $[z_{min}, z_{max}]$.

6.2.2 Estimation of the penalty weighting parameters

It is noted that the **w** values obtained from either Eq.(6.5) or Eq.(6.7) above, are optimal in the least square sense, presuming the penalty weighting parameters $\mathbf{p} = [\lambda\ \beta\ \alpha]^T$ are given. However, these penalty weighting parameters are in general unknown and need to be estimated. It is noted that the model complexity, which is partially influenced by penalty weighting parameters **p**, will depend on the set of available empirical data (training data) on which base the T-S fuzzy model is identified. As such, one may try to make use of the available empirical data to estimate penalty weighting parameters in order to get the best performance of the model. It has also been studied and suggested by [34] that the model performance indeed depends on penalty weighting parameters **p**. For instance, too small values of λ, β, and α will likely yield over-fitting, and lead to poor performance when extrapolating. In contrast, too large values of λ, β, and α give too little emphasis on the empirical data; as a result, the model may become too biased under the operation conditions where the *a priori* knowledge is incorrect or incomplete. In this subsection, we attempt to find the optimal penalty weighting parameters **p** to minimize the performance criterion, J (in Eq.(6.3)). In the ordinary least squares problem (i.e., there are no penalty weighting terms in J, or, $\mathbf{p} = [\lambda\ \beta\ \alpha]^T = [0\ 0\ 0]^T$), the optimal value of **w** is directly estimated from the available training data. Yet, due to the extra introduction of penalty terms in the present situation and with no further information available (no extra training data) to the identification problem, it is natural to take the best of the current available training data in order to estimate the optimal values of **w** associated to the presence of penalty terms. This problem implicitly relates to the selection of model structure. Various model structure selection heuristics from statistics can be used, such as the coefficient of determination, R^2*, or, residual mean square, s^2†, or several cross-validation based criteria (e.g., see [47]), etc. In the following development, we refer to [49] , who uses a re-estimation procedure to attain the optimal values of the penalty weighting parameter λ which is based on one of the cross-validation criteria‡, generalized cross-validation (GCV, denoted as σ^2_{GCV} below) [20] [24] [25] . In order to explicitly express the GCV criterion as a function of penalty weighting parameters **p** and model parameters **w**, the σ^2_{GCV} is formulated as

$$\sigma^2_{GCV} = \frac{np^2}{(tr(\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}^T))^2}\cdot(\frac{\mathbf{e}^T\mathbf{e}}{np}) \tag{6.8}$$

where matrix **A**, as defined in Eq.(6.6), is a covariance matrix which involves the given parameters of **p** ($=[\lambda\ \beta\ \alpha]^T$). These given parameters of **p** are used to estimate the optimal $\mathbf{w}^*$ according to Eq.(6.5). As soon as the optimal $\mathbf{w}^*$ is obtained based on given **p**, the prediction error, $\mathbf{e}\in \mathbb{R}^{np}$, can thus be calculated by

$$\begin{aligned}\mathbf{e} &= \mathbf{y}_d - \overline{\mathbf{V}}\cdot\mathbf{w}^*_{(\lambda,\beta,\alpha)} \\ &= (\mathbf{I}_{np} - \overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}^T)\cdot\mathbf{y}_d - \zeta\beta\ \overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_s{}^T\mathbf{y}_s - \eta\alpha\ \overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_a{}^T\mathbf{y}_a\end{aligned} \tag{6.9}$$

Hence, taking the derivative of σ^2_{GCV} with respect to penalty weighting parameters **p**, we have

$$\left[\frac{\partial\sigma^2_{GCV}}{\partial\varphi}\right] = \frac{-2\,np^2}{(tr(\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}^T))^3}\cdot tr(\overline{\mathbf{V}}\left[\frac{\partial\mathbf{A}^{-1}}{\partial\varphi}\right]\overline{\mathbf{V}}^T)\cdot(\frac{\mathbf{e}^T\mathbf{e}}{np}) + \frac{np}{(tr(\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}^T))^2}\cdot\left[\frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial\varphi}\right];$$

$$\varphi = \lambda\,,\ \beta\,,\ \text{or}\ \alpha\ \text{and}\ \mathbf{p} = [\ \lambda\ \beta\ \alpha\ ]^T \tag{6.10}$$

* R^2 is defined as the ratio of regression sum of square to the total sum of square, i.e. SS_{reg}/SS_{total}.

† s^2 is also called estimate of error variance, defined as the residual sum of square divided by the residual degree of freedom, i.e., SS_{res}/df(residual).

‡ Other relevant cross-validation based criteria are leave-one-out (LOO) cross-validation (or called PRESS residuals in statistics), unbiased estimate of variance (UEV), final prediction error (FPE) criterion (based on Akaike's information criterion AIC, as similar to Mallow's Cp in statistics), and Schwarz's Bayesian information criterion (BIC). We will use the generalized cross-validation (GCV) criterion to select the model structure because it is the most convenient method, see [49] .

The unknown part in Eq.(6.10) consists of the derivatives of $\mathbf{A}^{-1}$ and $\mathbf{e}^T\mathbf{e}$ with respect to $\mathbf{p}$. They are derived in Appendix C. Here we merely list the results:

$$\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]=\left[\frac{\partial \mathbf{A}^{-1}}{\partial \lambda} \vdots \frac{\partial \mathbf{A}^{-1}}{\partial \beta} \vdots \frac{\partial \mathbf{A}^{-1}}{\partial \alpha}\right]^T$$
$$=\begin{bmatrix} -\mathbf{A}_0^{-1}\mathbf{A}^{-1}+\mathbf{A}_0^{-2}-\mathbf{A}^{-1}\mathbf{A}_0^{-1}-\mathbf{A}_0^{-1}(\mathbf{A}_0-\mathbf{A}_0\mathbf{A}^{-1}\mathbf{A}_0)^2\mathbf{A}_0^{-3} \\ -\zeta\mathbf{A}_0^{-1}(\mathbf{A}_0-\mathbf{A}_0\mathbf{A}^{-1}\mathbf{A}_0)^2\mathbf{F}^{-1}\overline{\mathbf{V}}_s^T\overline{\mathbf{V}}_s\mathbf{F}^{-1}\mathbf{A}_0^{-1} \\ -\eta\mathbf{A}_0^{-1}(\mathbf{A}_0-\mathbf{A}_0\mathbf{A}^{-1}\mathbf{A}_0)^2\mathbf{F}^{-1}\overline{\mathbf{V}}_a^T\overline{\mathbf{V}}_a\mathbf{F}^{-1}\mathbf{A}_0^{-1} \end{bmatrix} \tag{6.11}$$

where

$$\mathbf{A}_0 = \overline{\mathbf{V}}^T\overline{\mathbf{V}} + \lambda\mathbf{I}_R \tag{6.12.a}$$
$$\mathbf{F} = \zeta\beta\overline{\mathbf{V}}_s^T\overline{\mathbf{V}}_s + \eta\alpha\overline{\mathbf{V}}_a^T\overline{\mathbf{V}}_a \tag{6.12.b}$$
$$\mathbf{A} = \mathbf{A}_0 + \mathbf{F} \tag{6.12.c}$$

Note that in Eq.(6.11) we need to calculate the inverse of matrices $\mathbf{A}$, $\mathbf{A}_0$, and $\mathbf{F}$. In general, matrices $\mathbf{A}$ and $\mathbf{A}_0$ are not singular, indicating that it is possible to take their inverses. However, in some situations, $\mathbf{F}$ can be singular such that its inverse becomes problematic. To avoid this singular problem, we modify Eq.(6.12) as follows, where we decompose the λ term into two parts, $(\lambda - \lambda_F)$ and λ_F, and distribute them into Eq.(6.12.a) and Eq.(6.12.b), respectively. This does not change the property of $\mathbf{A}$ in Eq.(6.12.c), but guarantees matrix $\mathbf{F}$ being non-singular due to the small constant λ_F added in its diagonals. In this study we set the small constant value, λ_F, as 10^{-6}. As a result, the modified Eq.(6.12.a) and Eq.(6.12.b) become

$$\mathbf{A}_0 = \overline{\mathbf{V}}^T\overline{\mathbf{V}} + (\lambda-\lambda_F)\mathbf{I}_R \tag{6.12.d}$$
$$\mathbf{F} = \zeta\beta\overline{\mathbf{V}}_s^T\overline{\mathbf{V}}_s + \eta\alpha\overline{\mathbf{V}}_a^T\overline{\mathbf{V}}_a + \lambda_F\mathbf{I}_R \tag{6.12.e}$$

For the derivative of $\mathbf{e}^T\mathbf{e}$ with respect to $\mathbf{p}$, we have

$$\left[\frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial \mathbf{p}}\right]=\left[\frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial \lambda} \vdots \frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial \beta} \vdots \frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial \alpha}\right]^T$$
$$\begin{aligned} &=\mathbf{y}_d^T\left\{(-\overline{\mathbf{V}}\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]\overline{\mathbf{V}}^T)^T\mathbf{K}+\mathbf{K}^T(-\overline{\mathbf{V}}\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]\overline{\mathbf{V}}^T)\right\}\mathbf{y}_d \\ &-2\mathbf{y}_d^T\left\{(-\overline{\mathbf{V}}\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]\overline{\mathbf{V}}^T)\cdot(\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a)+\mathbf{K}^T(\left[\frac{\partial \mathbf{K}_b}{\partial \mathbf{p}}\right])\mathbf{y}_s+\mathbf{K}^T(\left[\frac{\partial \mathbf{K}_a}{\partial \mathbf{p}}\right])\mathbf{y}_a\right\} \\ &+2\left\{\mathbf{y}_s^T\left[\frac{\partial \mathbf{K}_b}{\partial \mathbf{p}}\right]^T\mathbf{K}_b\mathbf{y}_s+\mathbf{y}_s^T(\left[\frac{\partial \mathbf{K}_b}{\partial \mathbf{p}}\right]^T\mathbf{K}_a+\mathbf{K}_b^T\left[\frac{\partial \mathbf{K}_a}{\partial \mathbf{p}}\right])\mathbf{y}_a+\mathbf{y}_a^T\left[\frac{\partial \mathbf{K}_a}{\partial \mathbf{p}}\right]^T\mathbf{P}_a\mathbf{y}_a\right\} \end{aligned} \tag{6.13}$$

where

$$\mathbf{K} = \mathbf{I}_{np} - \overline{\mathbf{V}}\,\mathbf{A}^{-1}\,\overline{\mathbf{V}}^{T} \tag{6.14.a}$$

$$\mathbf{K}_b = \zeta\beta\,\overline{\mathbf{V}}\,\mathbf{A}^{-1}\,\overline{\mathbf{V}}_s^{T} \tag{6.14.b}$$

$$\mathbf{K}_a = \eta\alpha\,\overline{\mathbf{V}}\,\mathbf{A}^{-1}\,\overline{\mathbf{V}}_a^{T} \tag{6.14.c}$$

$$\left[\frac{\partial \mathbf{K}_b}{\partial \mathbf{p}}\right] = \begin{bmatrix} \dfrac{\partial \mathbf{K}_b}{\partial \lambda} \\ \dfrac{\partial \mathbf{K}_b}{\partial \beta} \\ \dfrac{\partial \mathbf{K}_b}{\partial \alpha} \end{bmatrix} = \zeta\beta\overline{\mathbf{V}}\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]\overline{\mathbf{V}}_s^{T} + \begin{bmatrix} \mathbf{0} \\ \zeta\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_s^{T} \\ \mathbf{0} \end{bmatrix} \tag{6.14.d}$$

$$\left[\frac{\partial \mathbf{K}_a}{\partial \mathbf{p}}\right] = \begin{bmatrix} \dfrac{\partial \mathbf{K}_a}{\partial \lambda} \\ \dfrac{\partial \mathbf{K}_a}{\partial \beta} \\ \dfrac{\partial \mathbf{K}_a}{\partial \alpha} \end{bmatrix} = \eta\alpha\overline{\mathbf{V}}\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]\overline{\mathbf{V}}_a^{T} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \eta\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_a^{T} \end{bmatrix} \tag{6.14.e}$$

In [49] the re-estimation procedure is achieved by letting the derivative of σ^2_{GCV} with respect to $\mathbf{p}$ (Eq.(6.10)) be zero, so that the estimate of $\mathbf{p}$ is expressed as a function of these parameters themselves explicitly and implicitly through the covariance matrix $\mathbf{A}$. Given the initial value of penalty weighting parameter, $\mathbf{p}_0$, the generalized cross-validation criterion, Eq.(6.8), can be calculated and leads to a new estimate of $\mathbf{p}$. This new estimate is re-fed to calculate the generalized cross-validation criterion which results in another new estimate of $\mathbf{p}$. This procedure is repeated until the estimated values of $\mathbf{p}$ converge and finally an optimal estimate of $\mathbf{p}$ is obtained. Yet, due to the extra parts in ζ and η terms, it is not easy to present the above mentioned explicit functions in our case. As an alternative, we use the routine **constr*** of Matlab® for linear searching, which makes use of Eq.(6.8) and its derivatives $[\partial\sigma^2_{GCV}/\partial\mathbf{p}]$ to search the optimal $\mathbf{p}$. The resultant parameters λ, $\boldsymbol{\beta}$, and α, will be greater or equal to zeros†. During the search procedure, we recalculate the optimal $\mathbf{w}^*$ based on the current searched penalty weighting parameters, and then re-estimate the current prediction error Eq.(6.9), which appears in Eq.(6.8). The re-estimation procedure is depicted in Figure 6.1.

---

* The search procedure of optimal penalty weighting parameters suggested by [34] is **fminu**. Since **fminu** searches the optimal parameter that minimizes the cost function without any constraints to these parameters, the results will sometimes appear to be negative, which violate the definition of penalty weighting. Hence, we use **constr** to replace **fminu** by constraining all the parameters to be greater or equal to zero, but without an upper limit.

† Also note from the modified Eq.(6.12.d), as λ_F is set as 10^{-6} we'll further set λ to be greater than 10^{-5} to ensure a non-singular matrix $\mathbf{A}_0$.

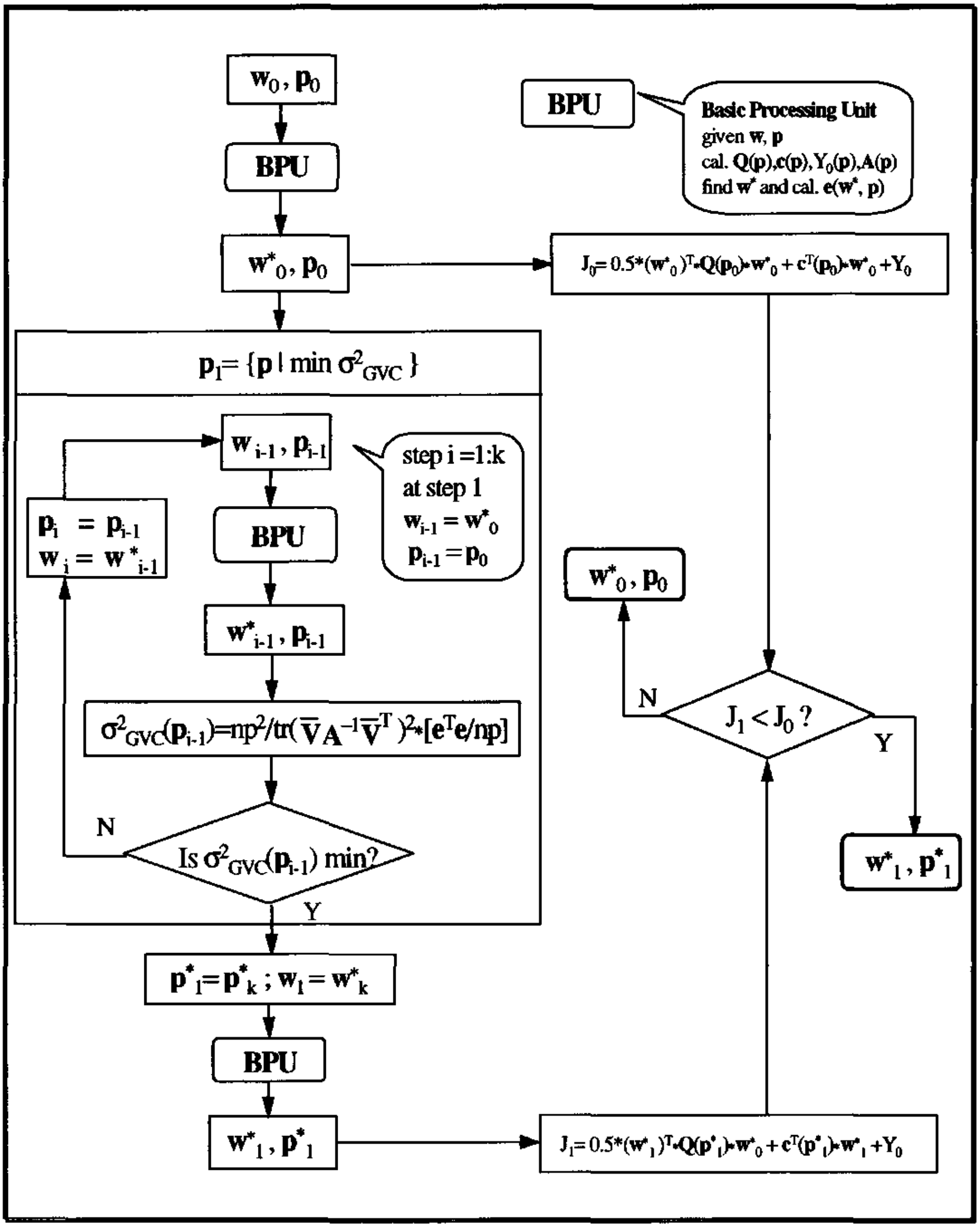

Figure 6.1: The re-estimation procedure to estimate the optimal values of λ, β, α, and w.

6.3 EXAMPLE

We test this optimization approach to a synthetic nonlinear single-input-single-output example,

$$y = 1 - 4x^2 + x^4 \tag{6.15}$$

First we define a possible operation range of the system for input x as [-2.5 2.5], a training range of x as [-0.9 1.1], and a validation range of x as [-2 2]. We assume some *a priori* qualitative information is available within the operation range [-2.5 2.5]. Furthermore, quantitative data are available over the training range. In order to test the extrapolation properties of the hybrid optimization approach, we let the validation range of input x be much broader than the training range.

A T-S fuzzy model is initialized by seven Gaussian membership functions to its input variable x. The widths of these Gaussian membership functions are set to be 0.5 and their centers are equally spaced in the operation range [-2.5 2.5] in order to be in line with the default Mamdani model (to be introduced later). The seven linguistic terms 'NB', 'NM', 'NS', 'ZO', 'PS', 'PM', and 'PB', stand for negative big, negative medium, negative small, zero, positive small, positive medium, and positive big, respectively; see Figure 6.2.(a). Accordingly, seven fuzzy rules are constructed for the T-S fuzzy model. The training data contain 38 pairs of input-output observations (np = 38), where input x is randomly selected from the above training range [-0.9 1.1] and the corresponding real system output is generated from Eq.(6.15) and to which some noise with mean -0.0002 and variance 0.0053 is added. The validation of the identified T-S fuzzy model will be tested by 81 inputs of x independent from the training data (nv = 81), with intervals of 0.05 in the validation range [-2 2]. The real system output is depicted in Figure 6.2.(d), where a solid line represents the system output defined on the validation range and the curve marked by circles the training range.

The qualitative information is represented by a Mamdani fuzzy model and is defined on the operation range [-2.5 2.5]. The default Mamdani fuzzy model has seven and four membership functions for its input and output, respectively (see Figure 6.2.(b) and (c)). The symbols 'NB', 'NM', 'NS', 'ZO', 'PS', 'PM', and 'PB' of input x denote the same linguistic terms as above, but are characterized by seven different membership functions. The output membership function is termed by four linguistic symbols, 'NS' - negative small, 'NM' - negative medium, 'NB' - negative big, and 'P' - positive. The seven fuzzy rules of the default Mamdani fuzzy model are listed below,

Rule_a 1: *IF* x is ZO *THEN* y is P (6.16.a)

Rule_a 2: *IF* x is PS *THEN* y is NS (6.16.b)

Rule_a 3: *IF* x is NS *THEN* y is NS (6.16.c)

Rule_a 4: *IF* x is PM *THEN* y is NM (6.16.d)

Rule_a 5: *IF* x is NM *THEN* y is NM (6.16.e)

Rule_a 6: *IF* x is PB *THEN* y is P (6.16.f)

Rule_a 7: *IF* x is NB *THEN* y is P (6.16.g)

Note that there are only 3 output membership functions are included in the default model currently due to the operation range used. The min-max fuzzy inference and centroid of gravity defuzzification method are used to obtain the prediction output of the default Mamdani fuzzy model. Choosing 167 input signals (na = 167) from the interval [-2.5 2.5] as default input $\mathbf{x}_a$ to the default Mamdani fuzzy model, results in 167 pairs of input-output records in the operation range. The default Mamdani fuzzy model has the approximate nonlinearity as that of the desired system, Eq.(6.15), but is quite imprecise in the validation range, see Figure 6.2.(d). This is allowed as the default Mamdani fuzzy model is not expected to be very accurate. It serves as a reference model to assist the estimation of the consequent weight parameters of the T-S fuzzy model under the present optimization approach. Furthermore, the following special situations of the system are regarded as soft constraints to the optimization criterion*,

Rule_s 1: *IF* x is close to zero *THEN* y is close to 1 (6.17.a)

Rule_s 2: *IF* x is close to positive 1 *THEN* y is close to -2 (6.17.b)

Rule_s 3: *IF* x is close to negative 1 *THEN* y is close to -2 (6.17.c)

Rule_s 4: *IF* x is close to positive 2 *THEN* y is close to 1 (6.17.d)

Rule_s 5: *IF* x is close to negative 2 *THEN* y is close to 1 (6.17.e)

We may regard these rules (Eq.(6.17)) as the specific states of the system $\mathbf{x}_s = [0\ 1\ -1\ 2\ -2]^T$, which yields the specific output of the system $\mathbf{y}_s = [1\ -2\ -2\ 1\ 1\ ]^T$. So, there are five soft constraints (ns = 5) imposed on the performance criterion J.

In the optimization of $\mathbf{w}$ and penalty weighting parameters $\mathbf{p}$, we first assign random numbers as an initialization to penalty weighting parameters $\mathbf{p}_0$. Then we use the least squares method to find the optimal $\mathbf{w}$ based on the initial $\mathbf{p}_0$. Next, according to the generalized cross-validation criterion, we re-estimate the penalty weighting parameter values, and then the corresponding optimal $\mathbf{w}$ is re-estimated based on the current $\mathbf{p}$ values on every iteration step; see Figure 6.1. For comparison of the performance of the identified T-S model affected by adding different kinds of *a priori* information, we consider the following cases in this example.

* In fact, in this example, the soft constraints contain similar information as the default rules but they have a higher a accuracy when compared to the Mamdani fuzzy rule base, although it is not necessary to deduce the fuzzy rules as independent soft constraints from the default Mamdani fuzzy model in this way. In many physical systems, we can gather this kind of soft constraint information based on first principles, e.g., the averaged mass/energy balance or some initial operation settings.

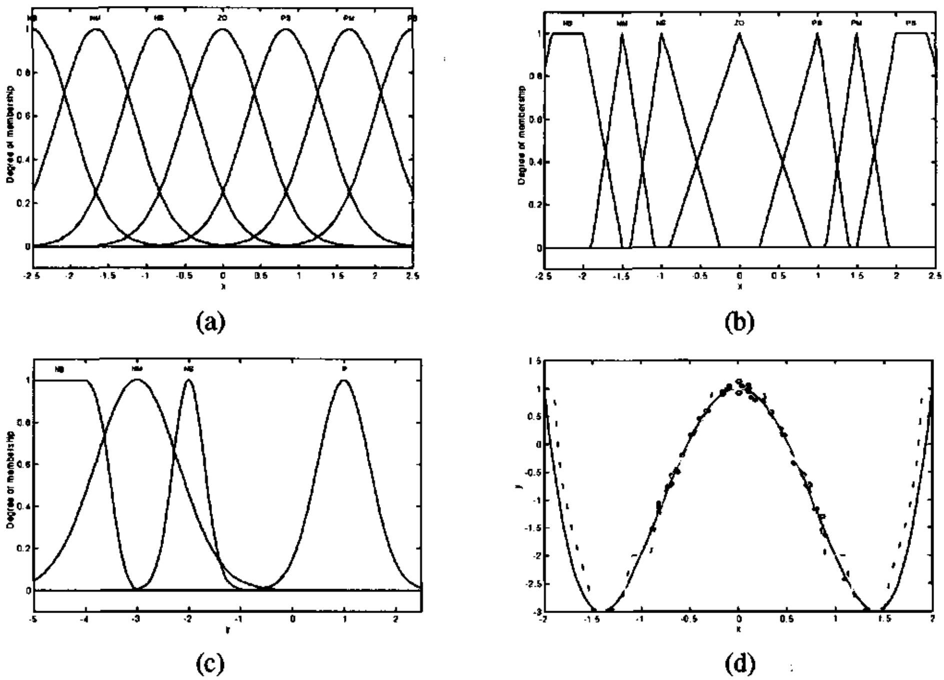

Figure 6.2: (a) The membership function of input x of the T-S fuzzy model. (b) and (c) the membership functions of input x and output y of the default Mamdani fuzzy model. (d) The real system response of validation data (solid line); training data (marked by 'o');the predicted output of the default Mamdani fuzzy model (dashed line).

Case_1: no *a priori* knowledge is added. I.e., the ordinary least squares method is used to identify the consequent weight parameters of the T-S fuzzy model, which means that the T-S fuzzy model merely makes use of the available training data.

Case_2: only regularization is considered. I.e., we request the T-S fuzzy model to be smooth and only the penalty weighting parameter λ is tuned.

Case_3: smoothness is required and the default Mamdani fuzzy model (Eq.(6.16)) is included into the identification of the T-S fuzzy model. I.e., we tune both parameters λ and α.

Case_4: smoothness and soft constraints (Eq.(6.17)) are required. I.e., we tune both parameters λ and β.

Case_5: smoothness and soft constraints (Eq.(6.17)) are required and the default Mamdani fuzzy model (Eq.(6.16)) is involved in the identification of the T-S fuzzy model. I.e., we tune all penalty weighting parameters λ, β, and α.

As a result, five models (denoted as Model_1, 2, 3, 4, and 5, shown next) based on the above different combinations of *a priori* qualitative information will be identified.

6.4 RESULTS

From the simulation, we have searched the non-trivial* optimal solutions for penalty weighting parameter values based on several trials of different initializations. The optimal penalty weighting parameters converge at the end of the re-estimation procedure and yield smaller values both on the generalized cross-validation criterion and the performance criterion. Shown below are the best values of the penalty weighting parameters of each model that have the minimal mean squared error to the training data. Besides, in order to evaluate the prediction performance of the identified T-S fuzzy models, we will compare the mean squared error based on training data and validation data, denoted by MSE_t and MSE_v, respectively, in the following table. If an identified T-S fuzzy model has a smaller MSE_v value than that of Model_1 (based on Case_1), then this will indicate that a better extrapolation ability can be obtained by incorporating the extra *a priori* knowledge into the modeling. Table 6.1 lists typical results of the simulation corresponding to the above five cases. The subscript 'f' denotes the resultant optimal value of penalty weighting parameters at the end of the re-estimation procedure. Predicted outputs of above five models based on the validation range of input x [-2 2] are depicted in Figure 6.3.

Table 6.1: The result of optimization of five models incorporating different *a priori* knowledge, where the optimal consequent weight of the corresponding T-S fuzzy model, $\mathbf{w}^*$, is shown here for reference. It is calculated from Eq.(6.5).

	λ_f	β_f	α_f	MSE_t	MSE_v
Model_1	-	-	-	0.00656	7.48987
Model_2	0.00061	-	-	0.00663	1.72106
Model_3	0.00001	-	0.04793	0.01274	0.18727
Model_4	0.00009	0.83666	-	0.00673	0.00805
Model_5	0.00001	0.03001	0.00244	0.00742	0.00371

	θ^*						
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7
Model_1	-8.9789	-5.3543	-1.0943	2.1150	-1.0791	-5.0078	-13.3197
Model_2	-0.1162	-5.4556	-1.0887	2.0990	-1.0190	-5.5094	-0.4563
Model_3	3.8351	-3.4875	-1.5665	2.3788	-1.4252	-4.3001	4.5412
Model_4	10.4161	-5.9694	-0.9987	2.0402	-0.9121	-6.1472	10.6412
Model_5	9.1615	-5.5461	-1.0929	2.1457	-1.1047	-5.5554	9.3180

* If the identified penalty weighting parameters **p** is $[0\ 0\ 0]^T$, we call it a trivial solution as it does not make sense in the present approach and, in fact, is the same solution as that of the ordinary least squares identification.

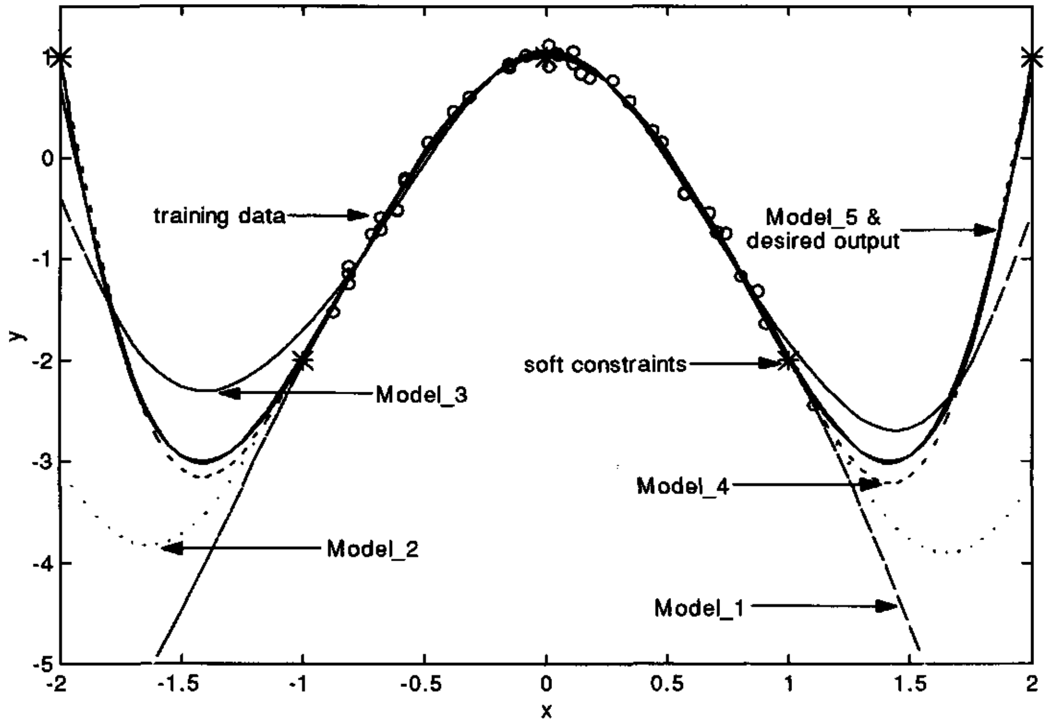

Figure 6.3: The predicted output of the identified models based on validation data. It is hard to distinguish between the output of Model_5 and the real system response (without noise). Points marked by circle 'o' denote the training data, whilst points marked by asterisk '$*$' the soft constraints, $x_s = [\ 0\ 1\ -1\ 2\ -2]^T$ and $y_s = [1\ -2\ -2\ 1\ 1]^T$. Note, some part of the output of Model_1 is cropped in order to stress the distinct outputs of other models.

From Table 6.1 we observe that Model_1, the T-S fuzzy model identified by ordinary least squares method without adding any further *a priori* knowledge, has very good interpolation performance in terms of the small MSE_t value. However, the extrapolation performance is very poor (MSE_v = 7.48987) and this can also be seen from Figure 6.3. Adding the extra *a priori* knowledge into the T-S fuzzy modeling, we can obtain comparable interpolation performance on the training range [-0.9 1.1] and much improved extrapolation performance on the validation range. (See Table 6.1; compare the MSE_v values). It is obvious from Figure 6.3 that the predicted output by the identified T-S fuzzy model, which involves any kind of *a priori* knowledge such as Model_2, 3, 4, and 5, is significantly distinct from that of Model_1. The responses of these models outside the training range, in particular, is notably improved from that of Model_1. All approach to the real system response, showing that superior extrapolation ability can be obtained by adding the extra *a priori* knowledge into modeling.

Simulation results show that the searched non-trivial values of the penalty weighting parameters do have noteworthy effects on the T-S fuzzy modeling. These values may serve as indicators on the relative correctness or relevance of the corresponding *a priori* knowledge to the used empirical training data. Yet, it shall be careful to make a direct interpretation of these penalty weighting parameters, because their correctness or relevance to the empirical training data is not verified. If the *a priori* knowledge can be proved to be correct in some extent, and if there is a large amount of data available covering all possible operation conditions, then the interpretations of the penalty weighting parameters may be feasible and meaningful [34] .

In this particular example, we notice that a larger λ value of each model results in smaller consequent weight values of the T-S fuzzy model. This shows that a bias term is introduced into the mean-squared-error evaluation. Consequently, the result favors solutions involving small absolute parameter values and the model then becomes smoother. This can be seen from Table 6.1 by comparing the **w** values of Model_2 to Model_1. Yet, Model_4 is an exception. This is probably caused by adding the soft constraints to the model, which has resulted in some underlying compensation effects. Furthermore, compared to the default Mamdani fuzzy model and noise contaminated training data, we know that the soft constraints imposed on the modeling is truly correct. The result of using correct *a priori* information is reflected by a larger final β value. This shows that it helps to incorporate correct soft constraints, like Eq.(6.17), into the T-S fuzzy model. On the other hand, since we are aware that the default Mamdani fuzzy model is not very precise (see Figure 6.2.(d)), the optimal value of α is not expected to be greater than β, provided both soft constraints and default Mamdani fuzzy model are imposed on the optimization criterion. As shown in Table 6.1, the α value is far smaller than the β value, certifying the difference in contribution to the fuzzy modeling. However, one shall be aware that the above example is particularly constructed to explain the modeling results by the optimization approach, rather than to deduce further information to interpret these penalty weighting parameters.

6.5 CONCLUSIONS

Often *a priori* qualitative knowledge can be put in a form like the ordinary Mamdani type of fuzzy rule. Thus, the Takagi-Sugeno type of fuzzy model is most suitable in using the quantitative information for modeling. In this chapter we have studied how to incorporate *a priori* knowledge into the T-S fuzzy model. It has been shown that combination of a Mamdani fuzzy model and a T-S fuzzy model in an optimization framework provides a basis for easy incorporation of the *a priori* knowledge into the fuzzy model. The resultant fuzzy model becomes more robust in terms of generalization on the extrapolation domain. Eventually, this approach can be extended to neural-fuzzy modeling.

An important condition in this optimization approach is that we have to presume that the available *a priori* knowledge is correct to some degree and relevant to the information content used for identification. Otherwise, the incorporation of *a priori* knowledge will yield misleading results. The principal idea is to regard *a priori* knowledge as constraints in the fuzzy modeling and to add penalties into the optimization performance criterion. The corresponding penalty weighting parameters are estimated in an optimal sense according to the generalized cross-validation criterion. With known penalty weighting parameters the consequent weights of the T-S fuzzy model with the extra *a priori* information, can be estimated optimally in the least square sense. The final estimates are obtained by iteration.

It often appears in neural networks identification that the amount of training data is limited, or sparsely distributed in the important regions of the input space. Hence, an overfitting problem may arise when too many parameters are used for modeling. This situation also occurs in the over-parameterization fuzzy model or neuro-fuzzy model. The over-parameterization problem becomes more significant to the first-order Takagi-Sugeno fuzzy model where the consequent part of the fuzzy rule is formed by a linear function of inputs, such that a large amount of free parameters have to be identified. Some studies, e.g., [3] [35] , have shown that regularization can reduce the overfitting problem. We have merely used the zeroth-order T-S fuzzy model as a basic model structure in our present example, and the over-parameterization problem does not seem to be serious in this case. Still, this problem will become more obvious when the number of fuzzy rules increases. One must try to avoid this problem. In addition to employing the orthogonal least squares method to reduce the redundant fuzzy rules (consequent weight parameters), it will be an interesting topic to check the effect of regularization on the over-parameterized T-S fuzzy model or neuro-fuzzy model.

Another pending question of this approach is the interpretation of the penalty weighting parameters. As mentioned before, the direct interpretation of these parameter values is not easy as the incorporated *a priori* knowledge is not yet justified. Provided that the *a priori* knowledge used can be certified as correct and relevant to larger operation conditions, then one could expect that these penalty weighting parameters would reveals the relative importance of the actual data vis-à-vis the *a priori* knowledge, from the point of view of prediction performance. More studies are needed on the interpretation of these penalty weighting parameters in conjunction with the *a priori* knowledge source used for modeling.

The example presented in this chapter shows significant effects can be achieved by adding penalties to the optimization performance criterion, resulting in an identified T-S fuzzy model having a convincing improvement on extrapolation. Looking at the interpolation aspect, we notice from the example that, even if the *a priori* knowledge is not completely correct, (refer to Figure 6.2.(d)), the identified T-S model still maintains a good accuracy in the interpolation (training) region, provided the training data really represent the underlying characteristics of the unknown system and the T-S fuzzy model uses a correct model structure (fuzzy rules). This situation shows that the empirical data always dominate the identified result of the T-S fuzzy model, just like any other black-box identification approach that is notably data-dependent. However, the present approach enables us to easily incorporate the *a priori* knowledge into the identification process, which is advantageous especially when we want to deal with extrapolation to regions where the training data are deficient.

7. CONCLUSIONS

There are five essentials for victory: (1) He will win who knows when to fight and when not to fight. (2) He will win who knows how to handle both superior and inferior forces. (3) He will win whose army is animated by the same spirit throughout all its ranks. (4) He will win who, prepared himself, waits to take the enemy unprepared. (5) He will win who has military capacity and is not interfered with by the sovereign.

- SunTzu, Art of the War

知可以戰與不可以戰者勝，識眾寡之用者勝，上下同欲者勝，以虞待不虞者勝，將能而君不御者勝。此五者，知勝之道也。

孫子兵法謀攻篇

This chapter concludes the studies presented in this thesis and provides some prospects for further research.

7.1 GENERAL CONCLUSIONS

As outlined by [88] , there is a considerable recent interest in *soft computing* dedicated to exploring the tolerance for imprecision and uncertainty, to learning from experience, and to adapting to changes under operation conditions. Soft computing points to an artificial intelligent system that consists of three principal components: fuzzy logic, neural network theory, and probabilistic reasoning. The fuzzy logic primarily deals with imprecision, the neural network with learning, and probabilistic reasoning with uncertainty. Although there are overlaps among these components, it is important to note that they function complementarily rather than competitively. Hence, advantages can usually be gained when they are employed in combination rather than exclusively. In this manner, remarkable results relating to soft computing have been achieved in recent times. The integrated neural-fuzzy model is exactly one instance of soft computing. In our present study, the integrated neural-fuzzy system, NUFZY, is used to deal with function approximation, where the fuzzy logic serves as a tool for approximate reasoning, and the neural network is in charge of the learning ability.

Although fuzzy logic theory is an extensive field that involves various concepts and principles as well as innumerous operators, it is limited in practice when actually being implemented in fuzzy reasoning. The first half of chapter 2 provides an introductory summary of both the Mamdani and the Takagi-Sugeno fuzzy models that is sufficient to explain the essence of fuzzy reasoning. The latter half of chapter 2 explicitly provides details of constructing the integrated neural-fuzzy system, which is functionally equivalent to a zeroth-order T-S fuzzy model, characterized by a transparent network structure and a self-explanatory representation of fuzzy rules. Two existing learning methods have been adapted and applied directly to the NUFZY model. This is illustrated in chapters 3 and 4, corresponding to the batch and the recursive learning schemes, respectively. Several practical examples with real data have been presented to demonstrate the capability of the NUFZY model for function approximation.

Research in soft computing is still ongoing and many answers are still pending. Two questions concerning the integrated neural-fuzzy model represented by the T-S fuzzy model were : how to obtain a linguistic interpretation from the fuzzy rules deduced by learning from training examples, and how to incorporate *a priori* knowledge into the T-S fuzzy model. This thesis has offered answers to both questions.

The first question arises since it has been often said that the T-S fuzzy rule cannot be easily interpreted linguistically due to its crisp consequent. In chapter 5, it is found that the fuzzy model has a property of dual representations. This property makes the T-S fuzzy model with a crisp consequent analogous to the Mamdani fuzzy model, if both models are defuzzified by weighed sum and all their rules are aggregated individually. As such, this offers a roundabout to transform the crisp consequent of the T-S fuzzy rule into a Mamdani - like fuzzy rule with an interpretable set of linguistic terms, where a new parameter set, the consequent significance level, is associated to the consequent of each Mamdani fuzzy rule. Hence, we have an extended Mamdani fuzzy model that has more flexible modeling ability than the ordinary Mamdani fuzzy model, allowing it to perform function approximation as well as the T-S fuzzy model.

Regarding the second question, it is not clearly shown in the literature how to incorporate *a priori* knowledge into the T-S fuzzy model. Since *a priori* knowledge is often qualitatively represented by a form like the Mamdani type of fuzzy rule and the T-S fuzzy model is most suitable in using quantitative information, a benefit for modeling shall be gained by combining both qualitative and quantitative information. In chapter 6, we employ an optimization approach to incorporate of *a priori* knowledge into the T-S fuzzy model. It has been shown that this approach constructs a basis for easily incorporating the *a priori* knowledge into the fuzzy model. The resultant fuzzy model becomes more robust in terms of generalization in the extrapolation domain. If desired, this approach can be extended to neural -fuzzy modeling without difficulty.

In summary, this thesis has touched aspects of soft computing by constructing an integrated neural-fuzzy model for function approximation, and by analyzing the problem of interpretability of the T-S fuzzy model, as well as the incorporation of *a priori* knowledge into the T-S fuzzy model.

7.2 FUTURE PERSPECTIVES

The work in this thesis is only the beginning of developing a comprehensive neural-fuzzy modeling technique. Much work still remains to be done. The studies made in this thesis provide a foundation for the prospective extension of the integrated neural-fuzzy modeling. We suggest the following directions.

- *Constrained learning of membership functions*: It has been pointed out in chapter 4 that the membership functions of input variables may lose their original linguistic interpretation after learning, due to the unawareness that the logical order on the universe of discourse of each input must be maintained. Another problem which may arise is that the original complete rule base on the linguistic level becomes incomplete on the numerical level after adaptation. The sparse rule base then gives blank intervals of non-overlapping membership functions in the input domains. As a result, the input-output mapping hypersurface of the neural-fuzzy model behaves discontinuously and the model may suffer from the problem of hysteresis when dealing with a dynamic system. Hence, it seems necessary to set constraints on the tuning of input membership functions in order to maintain a complete rule base and to obtain linguistically interpretable parameter sets of input variables. Learning methods, which are subject to the constraint of fuzzy partitions on the input variables, can achieve the above objectives.

- *Structure identification*: In the present learning process, we have only focused on identification in the parameter space based on a given model structure which is regarded as a reasonable representation of the unknown system. The model structure is established according to some *a priori* knowledge and trial-and-error. It is, however, necessary to carry out structure identification systematically in order to reduce the cost of trial-and-error, or to prevent abuse of *a priori* knowledge. In particular, the problem of curse of dimensionality arises when the number of system inputs become large and lead to a rapid increase of the fuzzy rule base. Several structure adaptation algorithms, which are based on principles of constructive or destructive learning, have been used in artificial neural network research for some time. In the fuzzy logic discipline, it appears that fuzzy c-mean clustering gradually becomes a possible alternative to determine the prototype fuzzy rule base using a set of training data. Besides, methods based on statistics and conventional system identification are also available. We are convinced that all the above methods can be applied successfully to improve the structure identification procedure for the integrated neural-fuzzy model.

- *Long-life learning*: It means that, in addition to recursive (or on-line) adaptation of the model parameters, the model shall have the ability to create new rules and to discard some inadequate old rules, so that the integrated system approaches the role of an expert, but is more flexible to meet reality. This is an extreme expectation concerning the issue of learning ability in the neural-fuzzy model.

Appendix A

OPTIMALITY OF THE GENERALIZED TAKAGI-SUGENO FUZZY MODEL

This appendix summarizes the results of the optimality of the generalized T-S fuzzy model. The details can be found in [33] . A MISO fuzzy system, the generalized T-S fuzzy rule is given by

$$\boldsymbol{R}^{r}_{(TS)}: \quad \boldsymbol{IF} \ \ \mathbf{x} \text{ is } A_r(\mathbf{x}) \quad \boldsymbol{THEN} \ \ y = g_r(\mathbf{x}) \tag{A.1}$$

where input $\mathbf{x} \in \boldsymbol{X} \subset \mathbb{R}^{ni}$, ni is the dimension of input x and the joint universe of discourse of x, $\boldsymbol{X}$, can be partitioned into R fuzzy subsets, $A_1(\mathbf{x})$, $A_2(\mathbf{x})$, .. , $A_r(\mathbf{x})$, .., $A_R(\mathbf{x})$. Each fuzzy subset is characterized by a membership function, $\mu_{Ar}(\mathbf{x})$, for r = 1, .. , R, that maps $\boldsymbol{X}$ into the bounded interval [0 1]. The grade value of membership function $\mu_{Ar}(\mathbf{x})$, is equivalent to the firing strength $v_r(\mathbf{x})$, in Eq.(2.31) and defined by Eq.(2.27.b). The local models (or consequent functions) $g_1(\mathbf{x})$, $g_2(\mathbf{x})$, .. , $g_r(\mathbf{x})$, .. , $g_R(\mathbf{x})$ are assumed to be continuous.

Let us suppose that there exists a global model y = G(x), then G(x) can be interpreted to be close to the local model $g_r(\mathbf{x})$ when " x is in $A_r(\mathbf{x})$ ". The fuzziness of $A_r(\mathbf{x})$, represented by $\mu_{Ar}(\mathbf{x})$, suggests that a penalty on mismatch between G(x) and $g_r(\mathbf{x})$ shall be large when $\mu_{Ar}(\mathbf{x})$ is large. This means the more the degree of fulfillment of fuzzy proposition " x is in $A_r(\mathbf{x})$ " (i.e., $\mu_{Ar}(\mathbf{x})$ is large), the more the global model G(x) shall approach the corresponding local model $g_r(\mathbf{x})$ under this proposition. Hence, we can define a criterion functional $J(G(\mathbf{x}))$ as a measure of the mismatch between the inferred global model G(x) and the local models, $g_r(\mathbf{x})$'s, which are derived from knowledge forming the T-S fuzzy rule base, as below,

$$J(G(\mathbf{x})) = \sum_{r=1}^{R} \int_{\mathbf{x} \in \mathbb{X}} [G(\mathbf{x}) - g_r(\mathbf{x})]^2 \cdot \mu_{A_r}(\mathbf{x})\, d\mathbf{x} \tag{A.2}$$

where J is defined in C($\boldsymbol{X}$), the space of all continuous functions on $\boldsymbol{X}$. The optimality theorem is then described below.

Theorem:

Suppose the fuzzy partition $A_1(\mathbf{x})$, $A_2(\mathbf{x})$, .. , $A_r(\mathbf{x})$, .., $A_R(\mathbf{x})$ of $\mathbb{X}$ is complete, and the membership functions $\mu_{A1}(\mathbf{x})$, $\mu_{A2}(\mathbf{x})$, .. , $\mu_{Ar}(\mathbf{x})$, .. , $\mu_{AR}(\mathbf{x})$, and the $g_1(\mathbf{x})$, $g_2(\mathbf{x})$, .. , $g_r(\mathbf{x})$, .. , $g_R(\mathbf{x})$ are elements of $C(\mathbb{X})$. Then the fuzzy inference described by a Takagi-Sugeno fuzzy model is optimal in the sense that the function

$$G(\mathbf{x}) = \sum_{r=1}^{R} \bar{\nu}_r(\mathbf{x}) \cdot g_r(\mathbf{x}) \tag{A.3}$$

minimizes the criterion functional $J(G(\mathbf{x}))$ on $C(\mathbb{X})$, where $\bar{\nu}_r(\mathbf{x})$ is the normalized firing strength as defined by Eq.(2.32).

It is easy to prove the theorem, since the completeness of the fuzzy partition $A_1(\mathbf{x})$, $A_2(\mathbf{x})$, .. , $A_r(\mathbf{x})$, .., $A_R(\mathbf{x})$ in $\mathbb{X}$ ensures that $J(G(\mathbf{x}))$ is convex and there exists a unique global minimum of $J(G(\mathbf{x}))$. Hence, taking the variation of $J(G(\mathbf{x}))$ with respect to any perturbation $\Delta G(\mathbf{x}) \in C(\mathbb{X})$, we have

$$\delta J(G;\Delta G) = 2\sum_{r=1}^{R} \int_{\mathbf{x}\in\mathbb{X}} [G(\mathbf{x}) - g_r(\mathbf{x})] \cdot \mu_{A_r}(\mathbf{x}) \cdot \Delta G(\mathbf{x})\, d\mathbf{x}$$

A necessary and sufficient condition for the minimum of $J(G(\mathbf{x}))$ is

$$\sum_{r=1}^{R} [G(\mathbf{x}) - g_r(\mathbf{x})] \cdot \mu_{A_r}(\mathbf{x}) = 0$$

Hence,

$$\sum_{r=1}^{R} G(\mathbf{x}) \cdot \mu_{A_r}(\mathbf{x}) - \sum_{r=1}^{R} g_r(\mathbf{x}) \cdot \mu_{A_r}(\mathbf{x}) = 0$$

then

$$G(\mathbf{x}) \cdot [\sum_{r=1}^{R} \mu_{A_r}(\mathbf{x})] = \sum_{r=1}^{R} \mu_{A_r}(\mathbf{x}) \cdot g_r(\mathbf{x})$$

Therefore, replacing $\mu_{Ar}(\mathbf{x})$ by the firing strength $\nu_r(\mathbf{x})$, we have

$$G(\mathbf{x}) = \frac{\sum_{r=1}^{R} \nu_r(\mathbf{x}) \cdot g_r(\mathbf{x})}{\sum_{p=1}^{R} \nu_p(\mathbf{x})} = \sum_{r=1}^{R} \bar{\nu}_r(\mathbf{x}) \cdot g_r(\mathbf{x})$$

This shows that the fuzzy inference mechanism by the generalized T-S fuzzy model, Eq.(A.1), is optimal with respect to the criterion functional $J(G(\mathbf{x}))$, defined in Eq.(A.2), which implies that it optimally mimics any continuous global mapping.

Appendix B

DERIVATION OF THE SENSITIVITY DERIVATIVES OF THE NUFZY SYSTEM

In this appendix we will show the derivation of the sensitivity derivatives of the NUFZY system. First, the derivatives of the NUFZY system with respect to the node's parameters on each layer are given. By means of the chain rule, the sensitivity derivative of the NUFZY system can be obtained. It is noted that $\mu_{ki}(x_i)$ and $\alpha_m(x_i)$ are functions of **x**, **c**, and **σ**. For simplicity, $\mu_{ki}(x_i,c_{i,ki},\sigma_{i,ki})$ is denoted as $\mu_{ki}(x_i)$ and $\alpha_m(x_i,c_{i,ki},\sigma_{i,ki})$ as $\alpha_m(x_i)$. If not particularly mentioned, parameter indexed by m will be used in this appendix for discussion, and the transformation of $\alpha_m(x_i)$ into $\mu_{ki}(x_i)$ is defined by Eq.(4.2.d). In the following derivation, index i runs from 1 to ni, m runs from 1 to M and n runs from 1 to nb. The tuning parameter set $\boldsymbol{\theta}$ can be defined as either $\boldsymbol{\theta} = \boldsymbol{\varpi}$, or $\boldsymbol{\theta} = [\boldsymbol{\varpi}\ \mathbf{c}]^T$, or $\boldsymbol{\theta} = [\boldsymbol{\varpi}\ \mathbf{c}\ \boldsymbol{\sigma}]^T$, where parameter vector $\boldsymbol{\theta}$ just stacks all the tuning parameter vector of $\boldsymbol{\varpi} = [\mathbf{w}_1^T\ ..\ \mathbf{w}_n^T\ ..\ \mathbf{w}_{nb}^T]^T$, with $\mathbf{w}_n = [w_{1n}\ ..\ w_{rn}\ ..\ w_{Rn}]^T$, and $\mathbf{c} = [c_1\ ..\ c_m\ ..\ c_M]^T$ as well as $\boldsymbol{\sigma} = [\sigma_1\ ..\ \sigma_m\ ..\ \sigma_M]^T$. Hence, the dimension of $\boldsymbol{\theta}$ can be nb×R, or nb×R+M, or nb×R+2M; M is defined in Eq.(4.3). Some notations of operations follow those used in Matlab®, i.e., .* and ./ represent array or element-by-element multiplication and division, respectively; kron means kronecker tensor product; $\Gamma(:,m)$ and $\Gamma(r,:)$ stand for the m^{th} column vector and the r^{th} row vector of matrix Γ, respectively.

B.1 NODE DERIVATIVES IN LAYER 1 OF THE NUFZY SYSTEM

(1) The derivative of the Gaussian membership function

(1-1) node parameter is **c**

$$\frac{\partial \alpha_m(x_i)}{\partial c_m} = \Phi_{c\,m} = \alpha_m\,(x_i - c_m)\,\sigma_m^{-2} \tag{B.1.a}$$

on the left edge of the Gaussian membership function,

$$\frac{\partial \mu_1(x_i)}{\partial c_{i,1}} = \Phi_{ci,1} = (\mu_1(x_i))^2 \exp((x_i - c_{i,1})\sigma_{i,1}) \cdot \sigma_{i,1} \tag{B.1.b}$$

on the right edge of the Gaussian membership function ,

$$\frac{\partial \mu_{Ni}(x_i)}{\partial c_{i,Ni}} = \Phi_{ci,Ni} = -(\mu_{Ni}(x_i))^2 \exp(-(x_i - c_{i,Ni})\sigma_{i,Ni}) \cdot \sigma_{i,Ni} \tag{B.1.c}$$

(1-2) node parameter is σ

$$\frac{\partial \alpha_m(x_i)}{\partial \sigma_m} = \Phi_{\sigma\,m} = \alpha_m\,(x_i - c_m)^2\,\sigma_m^{-3} \tag{B.2.a}$$

on the left edge,

$$\frac{\partial \mu_1(x_i)}{\partial \sigma_{i,1}} = \Phi_{\sigma i,1} = -(\mu_1(x_i))^2 \exp((x_i - c_{i,1})\sigma_{i,1}) \cdot (x_i - c_{i,1}) \tag{B.2.b}$$

on the right edge,

$$\frac{\partial \mu_{Ni}(x_i)}{\partial \sigma_{i,Ni}} = \Phi_{\sigma i,Ni} = (\mu_{Ni}(x_i))^2 \exp(-(x_i - c_{i,Ni})\sigma_{i,Ni}) \cdot (x_i - c_{i,Ni}) \tag{B.2.c}$$

(2) The derivative of the IMQ membership function

(2-1) node parameter is c

$$\frac{\partial \alpha_m(x_i)}{\partial c_m} = \Phi_{c\,m} = \alpha_m^{\;3}\,(x_i - c_m) \tag{B.3}$$

(2-2) node parameter is σ

$$\frac{\partial \alpha_m(x_i)}{\partial \sigma_m} = \Phi_{\sigma\,m} = -\,\alpha_m^{\;3}\,\sigma_m \tag{B.4}$$

B.2 NODE DERIVATIVES IN LAYER 2 OF THE NUFZY SYSTEM

(1) The partial derivative of firing strength, v_r, with respect to membership function value, α_m

$$\frac{\partial v_r}{\partial \alpha_m} = \frac{\partial(\prod_{i=1}^{ni} RM(r, a_i{:}b_i)\cdot \mu_i)}{\partial \alpha_m} = \begin{cases} \dfrac{v_r}{\alpha_m} & \text{if } RM(r,m)=1 \\ 0 & \text{if } RM(r,m)=0 \end{cases} \tag{B.5}$$

where RM(r,m) means the element on the r^{th} row and m^{th} column of the relationship matrix RM; notations $RM(r,a_i{:}b_i)$ and μ_i are defined in the same way as those in section 2.3.1.

(2) The partial derivative of $\bar{v}_r$ with respect to v_r

$$\frac{\partial \bar{v}_r}{\partial v_r} = \frac{\partial}{\partial v_r}[\frac{v_r}{\sum_{p=1}^{R} v_p}] = \frac{1-\bar{v}_r}{\sum_{p=1}^{R} v_p} \tag{B.6}$$

We can also use matrix notations to denote the above two partial derivatives. Firstly, let matrix Γ represent a R×M Jacobian matrix of $\frac{\partial \mathbf{v}}{\partial \boldsymbol{\alpha}}$, where $\mathbf{v} = [v_1 \; ..v_r \,..\, v_R]^T$ is a vector of firing strengths of all fuzzy rules and $\boldsymbol{\alpha} = [\alpha_1 \;\; \alpha_2 \;...\; \alpha_m \;...\; \alpha_M]^T$. One element of matrix Γ, Γ(r,m), is equal to $\frac{\partial v_r}{\partial \alpha_m}$. Then matrix Γ can be expressed by

$$\begin{aligned}
\boldsymbol{\Gamma} &= \frac{\partial \mathbf{v}}{\partial \boldsymbol{\alpha}} = [\frac{\partial \mathbf{v}}{\partial \alpha_1} \cdots \frac{\partial \mathbf{v}}{\partial \alpha_M}] \\
&= \boldsymbol{RM}.*\begin{bmatrix} \frac{v_1}{\alpha_1} & .. & \frac{v_1}{\alpha_M} \\ \vdots & \vdots & \vdots \\ \frac{v_R}{\alpha_1} & .. & \frac{v_R}{\alpha_M} \end{bmatrix} = \boldsymbol{RM}.*[\frac{\mathbf{v}}{\alpha_1} \cdots \frac{\mathbf{v}}{\alpha_M}] \\
&= \boldsymbol{RM}.*[\underbrace{\mathbf{v} \;\cdots\; \mathbf{v} \;\cdots\; \mathbf{v}}_{M}]./\left.\begin{bmatrix} \alpha_1..\alpha_m..\alpha_M \\ \vdots \\ \alpha_1..\alpha_m..\alpha_M \end{bmatrix}\right\}R \\
&= \boldsymbol{RM}.*[\mathrm{kron}(\mathbf{v}, \mathbf{I}_{1\times M})]./\,[\mathrm{kron}(\boldsymbol{\alpha}^T, \mathbf{I}_{R\times 1})]
\end{aligned} \tag{B.7}$$

Note that the m^{th} column vector of matrix $\mathbf{\Gamma}$ is given as

$$\Gamma(:,\mathrm{m}) = \left[\frac{\partial \mathbf{v}}{\partial \alpha_{\mathrm{m}}}\right] = \left[\frac{\partial v_1}{\partial \alpha_{\mathrm{m}}} \quad \cdots \quad \frac{\partial v_{\mathrm{r}}}{\partial \alpha_{\mathrm{m}}} \quad \cdots \quad \frac{\partial v_{\mathrm{R}}}{\partial \alpha_{\mathrm{m}}}\right]^{\mathrm{T}} \tag{B.8}$$

Secondly, the partial derivative of $\overline{\mathbf{v}}$ with respect to v, $\frac{\partial \overline{\mathbf{v}}}{\partial \mathbf{v}}$ is a R×R Jacobian matrix. Its element is expressed below.

$$\frac{\partial \overline{v}_{\mathrm{i}}}{\partial v_{\mathrm{j}}} = \begin{cases} \dfrac{1-\overline{v}_{\mathrm{i}}}{\sum_{\mathrm{r}=1}^{\mathrm{R}} v_{\mathrm{r}}} & , \quad \text{if } \mathrm{i}=\mathrm{j} \\ \dfrac{-\overline{v}_{\mathrm{i}}}{\sum_{\mathrm{r}=1}^{\mathrm{R}} v_{\mathrm{r}}} & , \quad \text{if } \mathrm{i}\neq\mathrm{j} \end{cases} \tag{B.9}$$

Therefore, giving

$$\frac{\partial \overline{\mathbf{v}}}{\partial \mathbf{v}} = \begin{bmatrix} \dfrac{\partial \overline{v}_1}{\partial v_1} & \cdot\cdot & \dfrac{\partial \overline{v}_1}{\partial v_{\mathrm{R}}} \\ \vdots & \cdot\cdot & \vdots \\ \dfrac{\partial \overline{v}_{\mathrm{R}}}{\partial v_1} & \cdot\cdot & \dfrac{\partial \overline{v}_{\mathrm{R}}}{\partial v_{\mathrm{R}}} \end{bmatrix} = \begin{bmatrix} 1-\overline{v}_1 & -\overline{v}_1 & \cdot\cdot & -\overline{v}_1 \\ -\overline{v}_2 & 1-\overline{v}_2 & \cdot\cdot & \vdots \\ \vdots & \cdot\cdot & \vdots & -\overline{v}_{\mathrm{R}-1} \\ -\overline{v}_{\mathrm{R}} & \cdot\cdot & -\overline{v}_{\mathrm{R}} & 1-\overline{v}_{\mathrm{R}} \end{bmatrix} / \sum_{\mathrm{p}=1}^{\mathrm{R}} v_{\mathrm{p}} \tag{B.10}$$

The diagonal of $\frac{\partial \overline{\mathbf{v}}}{\partial \mathbf{v}}$ is given as

$$\begin{aligned} \mathrm{diag}\left(\frac{\partial \overline{\mathbf{v}}}{\partial \mathbf{v}}\right) &= \begin{bmatrix} \dfrac{\partial \overline{v}_1}{\partial v_1} \\ \vdots \\ \dfrac{\partial \overline{v}_{\mathrm{R}}}{\partial v_{\mathrm{R}}} \end{bmatrix} = \begin{bmatrix} 1-\overline{v}_1 \\ \vdots \\ 1-\overline{v}_{\mathrm{R}} \end{bmatrix} / \sum_{\mathrm{p}=1}^{\mathrm{R}} v_{\mathrm{p}} \\ &= \frac{\mathbf{I}_{\mathrm{R}\times 1} - \overline{\mathbf{v}}}{\sum_{\mathrm{p}=1}^{\mathrm{R}} v_{\mathrm{p}}} \end{aligned} \tag{B.11}$$

B.3 NODE DERIVATIVES IN LAYER 3 OF THE NUFZY SYSTEM

Node derivative of $\hat{y}_n$ with respect to $\bar{v}_r$ is

$$\frac{\partial \hat{y}_n}{\partial \bar{v}_r} = w_{rn} \tag{B.12}$$

so

$$\begin{aligned}\frac{\partial \hat{y}_n}{\partial \bar{\mathbf{v}}} &= \left[\frac{\partial \hat{y}_n}{\partial \bar{v}_1} \cdot\cdot \frac{\partial \hat{y}_n}{\partial \bar{v}_r} \cdot\cdot \frac{\partial \hat{y}_n}{\partial \bar{v}_R}\right]^T \\ &= [w_{1n}\ w_{2n} \ldots w_{rn} \ldots w_{Rn}]^T \\ &= \mathbf{w}_n\end{aligned} \tag{B.13}$$

In the form of a matrix notation, let $\hat{\mathbf{y}} = [\hat{y}_1 \cdot\cdot \hat{y}_n \cdot\cdot \hat{y}_{nb}]^T$, then the derivative of $\hat{\mathbf{y}}$ with respect to $\bar{\mathbf{v}}$ is a R×nb Jacobian matrix, $\nabla_{\bar{\mathbf{v}}}\hat{\mathbf{y}}$, defined by

$$\begin{aligned}\nabla_{\bar{\mathbf{v}}}\hat{\mathbf{y}} &= \frac{\partial \hat{\mathbf{y}}}{\partial \bar{\mathbf{v}}} \\ &= \left[\frac{\partial \hat{y}_1}{\partial \bar{\mathbf{v}}} \cdots \frac{\partial \hat{y}_n}{\partial \bar{\mathbf{v}}} \cdots \frac{\partial \hat{y}_{nb}}{\partial \bar{\mathbf{v}}}\right] \\ &= \begin{bmatrix} \frac{\partial \hat{y}_1}{\partial \bar{v}_1} & \cdots & \frac{\partial \hat{y}_n}{\partial \bar{v}_1} & \cdots & \frac{\partial \hat{y}_{nb}}{\partial \bar{v}_1} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ \frac{\partial \hat{y}_1}{\partial \bar{v}_R} & \cdots & \frac{\partial \hat{y}_n}{\partial \bar{v}_R} & \cdots & \frac{\partial \hat{y}_{nb}}{\partial \bar{v}_R} \end{bmatrix}_{(R\times nb)} \\ &= \mathbf{W} = [\mathbf{w}_1 \cdots \mathbf{w}_n \cdots \mathbf{w}_{nb}]\end{aligned} \tag{B.14}$$

The n^{th} column of $\nabla_{\bar{\mathbf{v}}}\hat{\mathbf{y}}$, $\frac{\partial \hat{y}_n}{\partial \bar{\mathbf{v}}}$ ($= \mathbf{w}_n$) , will be used later in Eq.(B.19).

B.4 THE SENSITIVITY DERIVATIVE OF THE NUFZY SYSTEM WRT TUNING PARAMETERS

(1) The consequent weight parameters **w** are taken as tuning parameters.

In this case, the parameter set $\boldsymbol{\theta}$ is defined as

$$\boldsymbol{\theta}_{\varpi} = \boldsymbol{\varpi} = [\mathbf{w}_1^T \,..\, \mathbf{w}_n^T \,..\, \mathbf{w}_{nb}^T]^T \qquad ((nb \cdot R)\times 1 \text{ vector}) \tag{B.15}$$

For single output $\hat{y}_n$ (i.e., nb = 1), the partial derivative of $\hat{y}_n$ with respect to $\boldsymbol{\theta}_{\varpi}$, i.e., the sensitivity derivative $\boldsymbol{\Psi}_{\varpi_n} = \left[\frac{\partial \hat{y}_n}{\partial \boldsymbol{\theta}_{\varpi}}\right]$, becomes a R×1 vector

$$\begin{aligned}
\boldsymbol{\Psi}_{\varpi_n} &= \left[\frac{\partial \hat{y}_n}{\partial \boldsymbol{\theta}_{\varpi}}\right] \\
&= \left[\left[\frac{\partial \hat{y}_n}{\partial \mathbf{w}_1}\right]^T \cdots \left[\frac{\partial \hat{y}_n}{\partial \mathbf{w}_n}\right]^T \cdots \left[\frac{\partial \hat{y}_n}{\partial \mathbf{w}_{nb}}\right]^T\right]^T \\
&= \left[\left[\frac{\partial (\mathbf{w}_n^T \overline{\mathbf{v}})}{\partial \mathbf{w}_1}\right]^T \cdots \left[\frac{\partial (\mathbf{w}_n^T \overline{\mathbf{v}})}{\partial \mathbf{w}_n}\right]^T \cdots \left[\frac{\partial (\mathbf{w}_n^T \overline{\mathbf{v}})}{\partial \mathbf{w}_{nb}}\right]^T\right]^T \\
&= [\mathbf{0}^T \,..\, \overline{\mathbf{v}}^T \,..\, \mathbf{0}^T]^T \\
&= \begin{bmatrix} \mathbf{0} \\ \vdots \\ \overline{\mathbf{v}} \\ \vdots \\ \mathbf{0} \end{bmatrix}
\end{aligned} \tag{B.16}$$

where $\mathbf{0}$ is a R by 1 zero vector.

For the multi-output case, the sensitivity derivative $\boldsymbol{\Psi}_{\varpi} = \left[\frac{\partial \hat{\mathbf{y}}}{\partial \boldsymbol{\theta}_{\varpi}}\right]$, becomes a (nb·R)×nb matrix

$$\Psi_{\varpi} = \frac{\partial \hat{\mathbf{y}}}{\partial \theta_{\varpi}}$$
$$= \left[\left[\frac{\partial \hat{\mathbf{y}}_1}{\partial \theta_{\varpi}}\right] \cdots \left[\frac{\partial \hat{\mathbf{y}}_n}{\partial \theta_{\varpi}}\right] \cdots \left[\frac{\partial \hat{\mathbf{y}}_{nb}}{\partial \theta_{\varpi}}\right]\right] \tag{B.17}$$
$$= \begin{bmatrix} \overline{\mathbf{v}} & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \vdots & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \overline{\mathbf{v}} & \cdots & \vdots \\ \vdots & \cdots & \vdots & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \cdots & \overline{\mathbf{v}} \end{bmatrix}_{((nb\cdot R)\times nb)}$$

(2) **c** and **σ** are taken as parameters.

Where parameter set **θ** is defined as

$$\theta_c = \mathbf{c} = [c_1 \,..\, c_m \,..\, c_M]^T \qquad (\text{or}\ ,\ \theta_c = \mathbf{c} = [c_{11} \,..\, c_{i\,ki} \,..\, c_{ni\,Nni}]^T,\ M\times 1 \text{ vector}) \tag{B.18}$$

then element of the sensitivity derivative of Ψ_c (= $\left[\frac{\partial \hat{\mathbf{y}}}{\partial \theta_c}\right]$, an M×nb matrix), $\Psi_c(n,m)$ (= $\frac{\partial \hat{\mathbf{y}}_n}{\partial c_m}$), can be obtained.

$$\Psi_c(n,m) = \frac{\partial \hat{\mathbf{y}}_n}{\partial c_m} = \hat{\Sigma}_{r=1}^{R} \frac{\partial \hat{\mathbf{y}}_n}{\partial \overline{v}_r} \frac{\partial \overline{v}_r}{\partial v_r} \frac{\partial v_r}{\partial \alpha_m} \frac{\partial \alpha_m}{\partial c_m}$$
$$= \boldsymbol{RM}(:,m)^T * \left(\begin{bmatrix} \frac{\partial \hat{\mathbf{y}}_n}{\partial \overline{v}_1} \\ \vdots \\ \frac{\partial \hat{\mathbf{y}}_n}{\partial \overline{v}_r} \\ \vdots \\ \frac{\partial \hat{\mathbf{y}}_n}{\partial \overline{v}_R} \end{bmatrix} .* \begin{bmatrix} \frac{\partial \overline{v}_1}{\partial v_1} \\ \vdots \\ \frac{\partial \overline{v}_r}{\partial v_r} \\ \vdots \\ \frac{\partial \overline{v}_R}{\partial v_R} \end{bmatrix} .* \begin{bmatrix} \frac{\partial v_1}{\partial \alpha_m} \\ \vdots \\ \frac{\partial v_r}{\partial \alpha_m} \\ \vdots \\ \frac{\partial v_R}{\partial \alpha_m} \end{bmatrix} .* \begin{bmatrix} \frac{\partial \alpha_m}{\partial c_m} \\ \vdots \\ \frac{\partial \alpha_m}{\partial c_m} \\ \vdots \\ \frac{\partial \alpha_m}{\partial c_m} \end{bmatrix} \right) \tag{B.19}$$
$$= \boldsymbol{RM}(:,m)^T * \left[\mathbf{w}_n .* \mathrm{diag}\left(\frac{\partial \overline{\mathbf{v}}}{\partial \mathbf{v}}\right) .* \Gamma(:,m) .* \mathrm{kron}(\Phi_{cm}, \mathbf{I}_{R\times 1}) \right]$$

where $\mathbf{w}_n$ follows from Eq.(B.15), representing the n^{th} column vector of weight matrix **W** of Eq.(B.14); diag(.) follows from Eq.(B.11), representing the diagonal of the derivative matrix of $\overline{\mathbf{v}}$ with respect to **v** ; Γ(:,m) follows from Eq.(B.8), representing the m^{th} column vector of the Jacobian matrix Γ; Φ_{cm} follows from Eq.(B.1) or Eq.(B.3), representing the derivative of α_m with respect to c_m.

Similarly, where parameter set θ is defined as

$$\theta_\sigma = \sigma = [\sigma_1 \,..\, \sigma_m \,..\, \sigma_M]^T \qquad (\text{or } \theta_\sigma = [\sigma_{11} \,..\, \sigma_{i\,ki} \,..\, \sigma_{ni\,Nni}]^T,\ M\times 1 \text{ vector}) \tag{B.20}$$

then element of the sensitivity derivative of Ψ_σ ($=\left[\dfrac{\partial \hat{y}}{\partial \theta_\sigma}\right]$, an M×nb matrix), $\Psi_\sigma(n,m)$ ($= \dfrac{\partial \hat{y}_n}{\partial \sigma_m}$), can be obtained.

$$\begin{aligned}
\Psi_\sigma(n,m) &= \frac{\partial \hat{y}_n}{\partial \sigma_m} = \hat{\Sigma}_{r=1}^{R} \frac{\partial \hat{y}_n}{\partial \bar{v}_r}\frac{\partial \bar{v}_r}{\partial v_r}\frac{\partial v_r}{\partial \alpha_m}\frac{\partial \alpha_m}{\partial \sigma_m} \\
&= \boldsymbol{RM}(:,m)^T * \left(\begin{bmatrix} \frac{\partial \hat{y}_n}{\partial \bar{v}_1} \\ \vdots \\ \frac{\partial \hat{y}_n}{\partial \bar{v}_r} \\ \vdots \\ \frac{\partial \hat{y}_n}{\partial \bar{v}_R} \end{bmatrix} .* \begin{bmatrix} \frac{\partial \bar{v}_1}{\partial v_1} \\ \vdots \\ \frac{\partial \bar{v}_r}{\partial v_r} \\ \vdots \\ \frac{\partial \bar{v}_R}{\partial v_R} \end{bmatrix} .* \begin{bmatrix} \frac{\partial v_1}{\partial \alpha_m} \\ \vdots \\ \frac{\partial v_r}{\partial \alpha_m} \\ \vdots \\ \frac{\partial v_R}{\partial \alpha_m} \end{bmatrix} .* \begin{bmatrix} \frac{\partial \alpha_m}{\partial \sigma_m} \\ \vdots \\ \frac{\partial \alpha_m}{\partial \sigma_m} \\ \vdots \\ \frac{\partial \alpha_m}{\partial \sigma_m} \end{bmatrix} \right) \\
&= \boldsymbol{RM}(:,m)^T * \left[\mathbf{w}_n .* \mathrm{diag}(\frac{\partial \bar{\mathbf{v}}}{\partial \mathbf{v}}) .* \Gamma(:,m) .* \mathrm{kron}(\Phi_{\sigma m}, \mathbf{I}_{R\times 1}) \right]
\end{aligned} \tag{B.21}$$

where $\Phi_{\sigma m}$ follows fromEq.(B.2) or Eq.(B.4).

Therefore, the sensitivity derivative of the NUFZY system, Ψ, can be denoted either as Ψ_ϖ, or $[\Psi_\varpi\ ;\ \Psi_c\ ]$, or $[\Psi_\varpi\ ;\ \Psi_c\ ;\ \Psi_\sigma]$; representing (nb·R)×nb, or (nb·R+M)×nb, or (nb·R+2M)×nb matrix, respectively. The corresponding components of Ψ are obtained from Eq.(B.16), Eq.(B.17), Eq.(B.19), and Eq.(B.21).

Appendix C

DERIVATION OF DERIVATIVES OF σ^2_{GCV} WITH RESPECT TO p

In this appendix, we will show the derivation of the derivatives of ($\mathbf{A}^{-1}$) and ($\mathbf{e}^T\mathbf{e}$) with respect to penalty weighting parameters $\mathbf{p}$, which is defined by $\mathbf{p} = [\lambda\ \beta\ \alpha]^T$.

C.1 THE DERIVATIVE OF A^{-1} WITH RESPECT TO PENALTY WEIGHTING PARAMETERS p

According to Eq.(6.6) we have

$$\mathbf{A} = \overline{\mathbf{V}}^T\overline{\mathbf{V}} + \lambda\,\mathbf{I}_R + \zeta\,\beta\,\overline{\mathbf{V}}_s^T\overline{\mathbf{V}}_s + \eta\,\alpha\,\overline{\mathbf{V}}_a^T\overline{\mathbf{V}}_a \tag{C.1}$$

where the third and the fourth terms on the right hand side of matrix $\mathbf{A}$ can be expressed by symmetrical matrix products, $\overline{\mathbf{V}}_s^T\Lambda_s\overline{\mathbf{V}}_s$ and $\overline{\mathbf{V}}_a^T\Lambda_a\overline{\mathbf{V}}_a$, respectively. Square matrices Λ_s and Λ_a are R×R diagonal matrices with diagonal elements of $\zeta\beta$ and $\eta\alpha$, respectively. In order to reduce the complexity of formulation, we denote $\mathbf{F}$ as

$$\mathbf{F} = \zeta\,\beta\,\overline{\mathbf{V}}_s^T\overline{\mathbf{V}}_s + \eta\,\alpha\,\overline{\mathbf{V}}_a^T\overline{\mathbf{V}}_a \tag{C.2}$$

Hence, Eq.(C.1) becomes

$$\mathbf{A} = \mathbf{A}_0 + \mathbf{F} = \mathbf{A}_0 + \mathbf{I}_R\cdot\mathbf{F}\cdot\mathbf{I}_R \qquad \text{where } \mathbf{A}_0 = \overline{\mathbf{V}}^T\overline{\mathbf{V}} + \lambda\mathbf{I}_R \tag{C.3}$$

First we take the inversion of matrix $\mathbf{A}$, following the well-known matrix inversion lemma that

$$\mathbf{A}^{-1} = (\mathbf{A}_0 + \mathbf{XBY})^{-1} = \mathbf{A}_0^{-1} - \mathbf{A}_0^{-1}\mathbf{X}(\mathbf{Y}\mathbf{A}_0^{-1}\mathbf{X} + \mathbf{B}^{-1})^{-1}\mathbf{Y}\,\mathbf{A}_0^{-1}$$

Then

$$\mathbf{A}^{-1} = (\mathbf{A}_0 + \mathbf{F})^{-1} = \mathbf{A}_0^{-1} - \mathbf{A}_0^{-1}\,\mathbf{I}_R(\mathbf{I}_R\,\mathbf{A}_0^{-1}\,\mathbf{I}_R + \mathbf{F}^{-1})^{-1}\,\mathbf{I}_R\,\mathbf{A}_0^{-1}$$

$$= \mathbf{A}_0^{-1} - \mathbf{A}_0^{-1}(\mathbf{A}_0^{-1} + \mathbf{F}^{-1})^{-1}\mathbf{A}_0^{-1}$$

If we denote matrix **G** as

$$\mathbf{G} = \mathbf{A}_0^{-1} + \mathbf{F}^{-1} \tag{C.4}$$

Therefore, $\mathbf{A}^{-1}$ becomes

$$\mathbf{A}^{-1} = \mathbf{A}_0^{-1} - \mathbf{A}_0^{-1}\mathbf{G}^{-1}\mathbf{A}_0^{-1} \tag{C.5}$$

Premultiplying $\mathbf{A}_0$ and postmultiplying $\mathbf{A}_0$ to both sides of Eq.(C.5), it is easy to obtain the inversion of matrix **G** as

$$\mathbf{G}^{-1} = \mathbf{A}_0 - \mathbf{A}_0\mathbf{A}^{-1}\mathbf{A}_0 \tag{C.6}$$

Hence, from Eq.(C.5), the derivative of $\mathbf{A}^{-1}$ with respect to **p** is

$$\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right] = \left[\frac{\partial \mathbf{A}_0^{-1}}{\partial \mathbf{p}}\right] - \left\{\left[\frac{\partial \mathbf{A}_0^{-1}}{\partial \mathbf{p}}\right]\mathbf{G}^{-1}\mathbf{A}_0^{-1} + \mathbf{A}_0^{-1}\left[\frac{\partial \mathbf{G}^{-1}}{\partial \mathbf{p}}\right]\mathbf{A}_0^{-1} + \mathbf{A}_0^{-1}\mathbf{G}^{-1}\left[\frac{\partial \mathbf{A}_0^{-1}}{\partial \mathbf{p}}\right]\right\} \tag{C.7}$$

where, $\mathbf{G}^{-1}$ is defined by Eq.(C.6) and its derivative with respect to **p**, together with Eq.(C.4), is derived below.

$$\left[\frac{\partial \mathbf{G}^{-1}}{\partial \mathbf{p}}\right] = \left[\frac{\partial (\mathbf{A}_0^{-1} + \mathbf{F}^{-1})^{-1}}{\partial \mathbf{p}}\right] = -\mathbf{G}^{-2}\left[\frac{\partial \mathbf{G}}{\partial \mathbf{p}}\right] = -(\mathbf{A}_0^{-1} + \mathbf{F}^{-1})^{-2}\left\{\left[\frac{\partial \mathbf{A}_0^{-1}}{\partial \mathbf{p}}\right] + \left[\frac{\partial \mathbf{F}^{-1}}{\partial \mathbf{p}}\right]\right\} \tag{C.8}$$

In Eq.(C.7), the derivative of $\mathbf{A}^{-1}$ with respect to **p** mainly contains the derivatives of $\mathbf{A}_0^{-1}$ and $\mathbf{F}^{-1}$ with respect to **p**, respectively. We will derive the derivative of $\mathbf{A}_0^{-1}$ with respect to **p** foremost, and then, the derivative of $\mathbf{F}^{-1}$ with respect to **p** can be obtained accordingly. Both derivatives will be expressed in terms of matrices $\mathbf{A}_0$, **F** and **A** at the end.

Since $\mathbf{A}_0 \cdot \mathbf{A}_0^{-1} = \mathbf{I}_R$ and

$$\left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right] \cdot \mathbf{A}_0^{-1} + \mathbf{A}_0 \cdot \left[\frac{\partial \mathbf{A}_0^{-1}}{\partial \mathbf{p}}\right] = \frac{\partial \mathbf{I}_R}{\partial \mathbf{p}} = \mathbf{0}$$

therefore,

$$\left[\frac{\partial \mathbf{A}_0^{-1}}{\partial \mathbf{p}}\right] = -\mathbf{A}_0^{-1} \cdot \left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right] \cdot \mathbf{A}_0^{-1} \tag{C.9}$$

From Eq.(6.12.a), we have

$$\left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right]=\begin{bmatrix}\dfrac{\partial \mathbf{A}_0}{\partial \lambda}\\ \dfrac{\partial \mathbf{A}_0}{\partial \beta}\\ \dfrac{\partial \mathbf{A}_0}{\partial \alpha}\end{bmatrix}=\left[\frac{\partial(\overline{\mathbf{V}}^{\mathrm{T}}\overline{\mathbf{V}}+\lambda\mathbf{I}_{\mathrm{R}})}{\partial \mathbf{p}}\right]=\begin{bmatrix}\dfrac{\partial(\overline{\mathbf{V}}^{\mathrm{T}}\overline{\mathbf{V}}+\lambda\mathbf{I}_{\mathrm{R}})}{\partial \lambda}\\ \dfrac{\partial(\overline{\mathbf{V}}^{\mathrm{T}}\overline{\mathbf{V}}+\lambda\mathbf{I}_{\mathrm{R}})}{\partial \beta}\\ \dfrac{\partial(\overline{\mathbf{V}}^{\mathrm{T}}\overline{\mathbf{V}}+\lambda\mathbf{I}_{\mathrm{R}})}{\partial \alpha}\end{bmatrix}=\begin{bmatrix}\mathbf{I}_{\mathrm{R}}\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix} \tag{C.10}$$

Similarly, we have

$$\left[\frac{\partial \mathbf{F}^{-1}}{\partial \mathbf{p}}\right]=-\mathbf{F}^{-1}\cdot\left[\frac{\partial \mathbf{F}}{\partial \mathbf{p}}\right]\cdot\mathbf{F}^{-1} \tag{C.11}$$

and from Eq.(6.12.b),

$$\left[\frac{\partial \mathbf{F}}{\partial \mathbf{p}}\right]=\left[\frac{\partial(\zeta\beta\overline{\mathbf{V}}_s^{\mathrm{T}}\overline{\mathbf{V}}_s+\eta\alpha\overline{\mathbf{V}}_a^{\mathrm{T}}\overline{\mathbf{V}}_a)}{\partial \mathbf{p}}\right]=\begin{bmatrix}\dfrac{\partial \mathbf{F}}{\partial \lambda}\\ \dfrac{\partial \mathbf{F}}{\partial \beta}\\ \dfrac{\partial \mathbf{F}}{\partial \alpha}\end{bmatrix}=\begin{bmatrix}\dfrac{\partial(\zeta\beta\overline{\mathbf{V}}_s^{\mathrm{T}}\overline{\mathbf{V}}_s+\eta\alpha\overline{\mathbf{V}}_a^{\mathrm{T}}\overline{\mathbf{V}}_a)}{\partial \lambda}\\ \dfrac{\partial(\zeta\beta\overline{\mathbf{V}}_s^{\mathrm{T}}\overline{\mathbf{V}}_s+\eta\alpha\overline{\mathbf{V}}_a^{\mathrm{T}}\overline{\mathbf{V}}_a)}{\partial \beta}\\ \dfrac{\partial(\zeta\beta\overline{\mathbf{V}}_s^{\mathrm{T}}\overline{\mathbf{V}}_s+\eta\alpha\overline{\mathbf{V}}_a^{\mathrm{T}}\overline{\mathbf{V}}_a)}{\partial \alpha}\end{bmatrix}=\begin{bmatrix}\mathbf{0}\\ \zeta\overline{\mathbf{V}}_s^{\mathrm{T}}\overline{\mathbf{V}}_s\\ \eta\overline{\mathbf{V}}_a^{\mathrm{T}}\overline{\mathbf{V}}_a\end{bmatrix} \tag{C.12}$$

Hence, substitute Eq.(C.9) and Eq.(C.11) into Eq.(C.7), we have

$$\begin{aligned}\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]&=-\mathbf{A}_0^{-1}\left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right]\mathbf{A}_0^{-1}+\mathbf{A}_0^{-1}\left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right]\mathbf{A}_0^{-1}\mathbf{G}^{-1}\mathbf{A}_0^{-1}\\ &\quad+\mathbf{A}_0^{-1}\mathbf{G}^{-2}\left\{\left[\frac{\partial \mathbf{A}_0^{-1}}{\partial \mathbf{p}}\right]+\left[\frac{\partial \mathbf{F}^{-1}}{\partial \mathbf{p}}\right]\right\}\mathbf{A}_0^{-1}+\mathbf{A}_0^{-1}\mathbf{G}^{-1}\mathbf{A}_0^{-1}\left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right]\mathbf{A}_0^{-1}\\ &=\mathbf{A}_0^{-1}\left\{-\left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right]+\left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right]\mathbf{A}_0^{-1}\mathbf{G}^{-1}+\mathbf{G}^{-1}\mathbf{A}_0^{-1}\left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right]\right\}\mathbf{A}_0^{-1}\\ &\quad-\mathbf{A}_0^{-1}\mathbf{G}^{-2}\left\{\mathbf{A}_0^{-1}\left[\frac{\partial \mathbf{A}_0}{\partial \mathbf{p}}\right]\mathbf{A}_0^{-1}+\mathbf{F}^{-1}\left[\frac{\partial \mathbf{F}}{\partial \mathbf{p}}\right]\mathbf{F}^{-1}\right\}\mathbf{A}_0^{-1}\end{aligned} \tag{C.13}$$

Therefore, substitute Eq.(C.6), Eq.(C.10) and Eq.(C.12) into Eq.(C.13), we can obtain Eq.(6.11) as follows:

$$\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]=\begin{bmatrix}\frac{\partial \mathbf{A}^{-1}}{\partial \lambda}\\ \frac{\partial \mathbf{A}^{-1}}{\partial \beta}\\ \frac{\partial \mathbf{A}^{-1}}{\partial \alpha}\end{bmatrix}=\begin{bmatrix}-\mathbf{A}_0^{-1}\mathbf{A}^{-1}+\mathbf{A}_0^{-2}-\mathbf{A}^{-1}\mathbf{A}_0^{-1}-\mathbf{A}_0^{-1}(\mathbf{A}_0-\mathbf{A}_0\mathbf{A}^{-1}\mathbf{A}_0)^2\mathbf{A}_0^{-3}\\ -\zeta\mathbf{A}_0^{-1}(\mathbf{A}_0-\mathbf{A}_0\mathbf{A}^{-1}\mathbf{A}_0)^2\mathbf{F}^{-1}\overline{\mathbf{V}}_s^T\overline{\mathbf{V}}_s\mathbf{F}^{-1}\mathbf{A}_0^{-1}\\ -\eta\mathbf{A}_0^{-1}(\mathbf{A}_0-\mathbf{A}_0\mathbf{A}^{-1}\mathbf{A}_0)^2\mathbf{F}^{-1}\overline{\mathbf{V}}_a^T\overline{\mathbf{V}}_a\mathbf{F}^{-1}\mathbf{A}_0^{-1}\end{bmatrix} \quad \text{(C.14)}$$

C.2 THE DERIVATIVE OF e^Te WITH RESPECT TO PENALTY WEIGHTING PARAMETERS p

First, we denote the following terms,

$$\mathbf{K} = \mathbf{I}_{np} - \overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}^T \quad \text{(C.15.a)}$$

$$\mathbf{K}_b = \zeta\beta\,\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_s^T \quad \text{(C.15.b)}$$

$$\mathbf{K}_a = \eta\alpha\,\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_a^T \quad \text{(C.15.c)}$$

Then the derivatives of these matrices $\mathbf{K}$, $\mathbf{K}_b$, and $\mathbf{K}_a$ with respect to the penalty weighting parameters **p** are expressed below.

$$\left[\frac{\partial \mathbf{K}}{\partial \mathbf{p}}\right]=(-\overline{\mathbf{V}}\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]\overline{\mathbf{V}}^T) \quad \text{(C.16.a)}$$

$$\left[\frac{\partial \mathbf{K}_b}{\partial \mathbf{p}}\right]=\left[\frac{\partial(\zeta\beta\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_s^T)}{\partial \mathbf{p}}\right]=\zeta\beta\overline{\mathbf{V}}\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]\overline{\mathbf{V}}_s^T+\begin{bmatrix}\mathbf{0}\\ \zeta\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_s^T\\ \mathbf{0}\end{bmatrix} \quad \text{(C.16.b)}$$

$$\left[\frac{\partial \mathbf{K}_a}{\partial \mathbf{p}}\right]=\left[\frac{\partial(\eta\alpha\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_a^T)}{\partial \mathbf{p}}\right]=\eta\alpha\overline{\mathbf{V}}\left[\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{p}}\right]\overline{\mathbf{V}}_a^T+\begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ \eta\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_a^T\end{bmatrix} \quad \text{(C.16.c)}$$

According to Eq.(6.9) the error **e** is expressed by

$$\mathbf{e} = (\mathbf{I}_{np} - \overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}^T)\cdot\mathbf{y}_d - \zeta\beta\,\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_s^T\mathbf{y}_s - \eta\alpha\,\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}_a^T\mathbf{y}_a$$

$$= \mathbf{K}\mathbf{y}_d - \mathbf{K}_b\mathbf{y}_s - \mathbf{K}_a\mathbf{y}_a$$

and the square of error is,

$$
\begin{aligned}
\mathbf{e}^T\mathbf{e} &= (\mathbf{K}\mathbf{y}_d - \mathbf{K}_b\mathbf{y}_s - \mathbf{K}_a\mathbf{y}_a)^T\cdot(\mathbf{K}\mathbf{y}_d - \mathbf{K}_b\mathbf{y}_s - \mathbf{K}_a\mathbf{y}_a)\\
&= \mathbf{y}_d^T\cdot\mathbf{K}^T\mathbf{K}\cdot\mathbf{y}_d - 2\,\mathbf{y}_d^T\mathbf{K}^T\cdot(\mathbf{K}_b\mathbf{y}_s + \mathbf{K}_a\mathbf{y}_a) + (\mathbf{K}_b\mathbf{y}_s - \mathbf{K}_a\mathbf{y}_a)^T\cdot(\mathbf{K}_b\mathbf{y}_s - \mathbf{K}_a\mathbf{y}_a)
\end{aligned}
$$

Since matrices $\mathbf{K}$, $\mathbf{K}_b$, and $\mathbf{K}_a$ involve the penalty weighting parameters $\mathbf{p}$, the derivative of $\mathbf{e}^T\mathbf{e}$ with respect to $\mathbf{p}$ can be derived and expressed by derivatives that are defined from Eq.(C.16) and Eq.(C.14).

$$
\begin{aligned}
\left[\frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial\mathbf{p}}\right] &= \mathbf{y}_d^T\cdot\left\{\left[\frac{\partial\mathbf{K}^T}{\partial\mathbf{p}}\right]\mathbf{K}+\mathbf{K}^T\left[\frac{\partial\mathbf{K}}{\partial\mathbf{p}}\right]\right\}\cdot\mathbf{y}_d\\
&\quad -2\mathbf{y}_d^T\cdot\left[\frac{\partial(\mathbf{K}^T\cdot(\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a))}{\partial\mathbf{p}}\right]\\
&\quad +\left[\frac{\partial((\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a)^T\cdot(\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a))}{\partial\mathbf{p}}\right]
\end{aligned}
\tag{C.17}
$$

where

$$
\begin{aligned}
&\left[\frac{\partial\mathbf{K}^T}{\partial\mathbf{p}}\right]\\
&= -\overline{\mathbf{V}}^T\left[\frac{\partial\mathbf{A}^{-1}}{\partial\mathbf{p}}\right]^T\overline{\mathbf{V}} = (-\overline{\mathbf{V}}\left[\frac{\partial\mathbf{A}^{-1}}{\partial\mathbf{p}}\right]\overline{\mathbf{V}}^T)^T
\end{aligned}
\tag{C.18.a}
$$

and

$$
\begin{aligned}
&\left[\frac{\partial(\mathbf{K}^T\cdot(\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a))}{\partial\mathbf{p}}\right]\\
&=\left[\frac{\partial\mathbf{K}^T}{\partial\mathbf{p}}\right]\cdot(\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a)+\mathbf{K}^T\cdot\left[\frac{\partial(\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a)}{\partial\mathbf{p}}\right] \quad (\text{Note, } \mathbf{K}^T=\mathbf{K})\\
&=(-\overline{\mathbf{V}}\left[\frac{\partial\mathbf{A}^{-1}}{\partial\mathbf{p}}\right]\overline{\mathbf{V}}^T)^T\cdot(\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a)+\mathbf{K}^T\cdot(\left[\frac{\partial\mathbf{K}_b}{\partial\mathbf{p}}\right]\mathbf{y}_s+\left[\frac{\partial\mathbf{K}_a}{\partial\mathbf{p}}\right]\mathbf{y}_a)
\end{aligned}
\tag{C.18.b}
$$

and

$$\left[\frac{\partial((\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a)^T\cdot(\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a))}{\partial\mathbf{p}}\right]$$

$$=\left[\frac{\partial(\mathbf{y}_s^T\mathbf{K}_b^T\mathbf{K}_b\mathbf{y}_s+\mathbf{y}_s^T\mathbf{K}_b^T\mathbf{K}_a\mathbf{y}_a+\mathbf{y}_a^T\mathbf{K}_a^T\mathbf{K}_b\mathbf{y}_s+\mathbf{y}_a^T\mathbf{K}_a^T\mathbf{K}_a\mathbf{y}_a)}{\partial\mathbf{p}}\right]$$

$$=2\mathbf{y}_s^T\left[\frac{\partial\mathbf{K}_b^T}{\partial\mathbf{p}}\right]\mathbf{K}_b\mathbf{y}_s+(\mathbf{y}_s^T\left[\frac{\partial\mathbf{K}_b^T}{\partial\mathbf{p}}\right]\mathbf{K}_a\mathbf{y}_a+\mathbf{y}_s^T\mathbf{K}_b^T\left[\frac{\partial\mathbf{K}_a}{\partial\mathbf{p}}\right]\mathbf{y}_a) \tag{C.18.c}$$

$$+(\mathbf{y}_a^T\left[\frac{\partial\mathbf{K}_a^T}{\partial\mathbf{p}}\right]\mathbf{K}_b\mathbf{y}_s+\mathbf{y}_a^T\mathbf{K}_a^T\left[\frac{\partial\mathbf{K}_b}{\partial\mathbf{p}}\right]\mathbf{y}_s)+2\mathbf{y}_a^T\left[\frac{\partial\mathbf{K}_a^T}{\partial\mathbf{p}}\right]\mathbf{K}_a\mathbf{y}_a$$

$$=2\mathbf{y}_s^T\left[\frac{\partial\mathbf{K}_b^T}{\partial\mathbf{p}}\right]\mathbf{K}_b\mathbf{y}_s+2(\mathbf{y}_s^T\left[\frac{\partial\mathbf{K}_b^T}{\partial\mathbf{p}}\right]\mathbf{K}_a+\mathbf{y}_s^T\mathbf{K}_b^T\left[\frac{\partial\mathbf{K}_a}{\partial\mathbf{p}}\right])\mathbf{y}_a+2\mathbf{y}_a^T\left[\frac{\partial\mathbf{K}_a^T}{\partial\mathbf{p}}\right]\mathbf{K}_a$$

Substitute Eq.(C.14), Eq.(C.16) and Eq.(C.18) into Eq.(C.17), we have the derivative of $(\mathbf{e}^T\mathbf{e})$ with respect to **p**.

$$\left[\frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial\mathbf{p}}\right]=\left[\frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial\lambda}\ \vdots\ \frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial\beta}\ \vdots\ \frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial\alpha}\right]^T$$

$$=\mathbf{y}_d^T\left\{(-\overline{\mathbf{V}}\left[\frac{\partial\mathbf{A}^{-1}}{\partial\mathbf{p}}\right]\overline{\mathbf{V}}^T)^T\mathbf{K}+\mathbf{K}^T(-\overline{\mathbf{V}}\left[\frac{\partial\mathbf{A}^{-1}}{\partial\mathbf{p}}\right]\overline{\mathbf{V}}^T)\right\}\mathbf{y}_d$$

$$-2\mathbf{y}_d^T\left\{(-\overline{\mathbf{V}}\left[\frac{\partial\mathbf{A}^{-1}}{\partial\mathbf{p}}\right]\overline{\mathbf{V}}^T)^T\cdot(\mathbf{K}_b\mathbf{y}_s+\mathbf{K}_a\mathbf{y}_a)+\mathbf{K}^T(\left[\frac{\partial\mathbf{K}_b}{\partial\mathbf{p}}\right])\mathbf{y}_s+\mathbf{K}^T(\left[\frac{\partial\mathbf{K}_a}{\partial\mathbf{p}}\right])\mathbf{y}_a\right\} \tag{C.19}$$

$$+2\left\{\mathbf{y}_s^T\left[\frac{\partial\mathbf{K}_b}{\partial\mathbf{p}}\right]^T\mathbf{K}_b\mathbf{y}_s+\mathbf{y}_s^T(\left[\frac{\partial\mathbf{K}_b}{\partial\mathbf{p}}\right]^T\mathbf{K}_a+\mathbf{K}_b^T\left[\frac{\partial\mathbf{K}_a}{\partial\mathbf{p}}\right])\mathbf{y}_a+\mathbf{y}_a^T\left[\frac{\partial\mathbf{K}_a}{\partial\mathbf{p}}\right]^T\mathbf{K}_a\mathbf{y}_a\right\}$$

From Eq.(6.10) we have the expression of the derivative of σ^2_{GCV} with respect to **p** $(=[\lambda\ \beta\ \alpha]^T)$ as below,

$$\left[\frac{\partial\sigma^2_{GCV}}{\partial\varphi}\right]=\frac{-2np^2}{(\mathrm{tr}(\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}^T))^3}\cdot\mathrm{tr}(\overline{\mathbf{V}}\left[\frac{\partial\mathbf{A}^{-1}}{\partial\varphi}\right]\overline{\mathbf{V}}^T)\cdot(\frac{\mathbf{e}^T\mathbf{e}}{np})+\frac{np}{(\mathrm{tr}(\overline{\mathbf{V}}\mathbf{A}^{-1}\overline{\mathbf{V}}^T))^2}\cdot\left[\frac{\partial(\mathbf{e}^T\mathbf{e})}{\partial\varphi}\right];$$

$$\varphi=\lambda\ ,\ \beta\ ,\ \text{or}\ \alpha \tag{C.20}$$

Hence, substituting the corresponding components of Eq.(C.14) and Eq.(C.19) into Eq.(C.20), we obtain derivatives $\left[\frac{\partial\sigma^2_{GCV}}{\partial\lambda}\right]$, $\left[\frac{\partial\sigma^2_{GCV}}{\partial\beta}\right]$, and $\left[\frac{\partial\sigma^2_{GCV}}{\partial\alpha}\right]$.

REFERENCES

[1] Beale, R. and T. Jackson (1990) *Neural Computing: An Introduction*. Adam Hilger, Bristol, UK.

[2] Bertin, N. and E. Heuvelink (1993) Dry matter production in a tomato crop: comparison of two simulation models. *Journal of Horticultural Science*, **68** : 995-1011.

[3] Bishop, C. (1991) Improving the generalization properties of radial basis function neural networks. *Neural Computation*, **3** : 579-588.

[4] Bot, G. P. A. (1983) *Greenhouse climate: from physical processes to a dynamic model, PhD thesis*. Wageningen Agricultural University, Wageningen, The Netherlands.

[5] Brown, M. and C. Harris (1994) *Neurfuzzy adaptive modelling and control*. Prentice Hall, Englewood Cliffs, NJ, USA.

[6] Castro, J. L. (1995) Fuzzy controller are universal approximators. *IEEE Trans. Syst., Man, Cybern.*, **25** : 629-635.

[7] Chen, S. and S. A. Billings (1992) Neural networks for nonlinear dynamic system modelling and identification. *Int. J. Control*, **56** (2): 319-346.

[8] Chen, S., S. A. Billings and W. Luo (1989) Orthogonal least squares methods and their application to non-linear system identification. *Int. J. Control*, **50** : 219-269.

[9] Chen, S., S.A. Billings, C. F. N. Cowan, and P.M. Grant (1990) Non-linear systems identification using radial basisi functions. *Int. J. Systems Science*, **21** (12): 2513-2539.

[10] Chen, S., C. F. N. Cowan, S.A. Billings, and P.M. Grant (1990) Parallel recursive prediction error algorithm for training layered neural networks. *Int. J. Control*, **51** (2): 1215-1228.

[11] Chen, S., C. F. N. Cowan, and P. M. Grant (1991) Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Trans. Neural Networks*, **2** : 302-309.

[12] Chen, S., G. J. Gibson, C. F. N. Cowan, and P. M. Grant (1991) Reconstruction of binary signals using an adaptive radial-basis-function equalizer. *Signal Processing*, **22** : 77-93.

[13] de Zwart, H. F. (1996) *Analyzing energy-saving options in greenhouse cultivation using a simulation model, PhD thesis*, Wageningen Agricultural University, Wageningen, The Netherlands.

[14] Draper, N. R. and R. C. van Nostrand (1979) Ridge regression and James-Stein estimation: review and comments. *Technometrics*, **21** (4) : 451-466.

[15] Demuth, H. and M. Beale (1992) *Neural net toolbox user's guide*. The MathWorks, Natick, Massachusetts, USA.

[16] Dubois, D. and H. Prade (1980) *Fuzzy stes and systems: theory and applications*. Academic Press.

[17] Geman, S., E. Bienenstock and R. Doursat (1992) Neural Networks and the Bias/Variance Dielmma. *Neural Computation*, **4** : 1-58.

[18] Girosi, F., M. Jones, and T. Poggio (1995) Regularization theory and neural network architectures . *Neural Computation*, **7** : 219-269.

[19] Golding, A. R. and P. S. Rosenbloom (1996) Improving accuracy by combining rule-based and case-based reasoning. *Artificial Intelligence*, **87** (1-2) : 215-254.

[20] Golub, G. H., M. Heath and G. Wahba (1979) Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, **21** (2) : 215-223.

[21] Harris, C. J., M. Brown, K. M. Bossley, D. J. Mills, and F. Ming (1996) Advances in neurofuzzy algorithms for real-time modelling and control. *Engineering Applications of Artificial Intelligence*, **9** (1) : 1-16.

[22] Harris, C. J., C. G. Moore and M. Brown (1993) *Intelligent control - aspects of fuzzy logic and neural nets*. World Science, Singapore.

[23] Hauptmann, W. and K. Heesche (1994) A prototype of an integrated fuzzy-neuro system. *Proceedings of EUFIT'94, the second European Congress on Fuzzy and Intelligent Technologies*, Aachen, Germany. September 20-23, 1994. pp.1101-1104.

[24] Hemmerle, W. J. (1975) An explicit solution for generalized ridge regression. *Technometrics*, **17** (3) : 309-314.

[25] Hoerl, A. E. and R. W. Kennard (1976) Ridge regression iterative estimation of the biasing parameter. *Commun. Statis.*, **A5** (1) : 77-88.

[26] Horikawa, S. I., T. Furuhashi and Y. Uchikawa (1992) On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm. *IEEE Trans. Neural Networks*, **3** : 801-806.

[27] Hornik, K., M. Stinchcombe and H. Uchikawa (1989) Multilayer feedforward networks are universal approximators. *Neural Networks*, **2** : 359-366.

[28] Hunt, K. J., D. Sbarbaro, R. Zbikowski, and P. J. Gawthrop (1992) Neural networks for control systems - a survey. *Automatica*, **28** (6): 1083-1112.

[29] Hush, D. R. and B. G. Horne (1993) Progress in supervised neural networks - What's new since Lippmann?. *IEEE Signal Processing Magazine*, January 1993 : 8-39.

[30] Jager, R. (1995) *Fuzzy logic in control, PhD thesis.* Delft University of Technology, Delft, The Netherlands.

[31] Jang, J.-S. R. (1993) ANFIS: adaptive-network-based fuzzy inference system. *IEEE Trans. Syst., Man, Cybern.*, **23** : 665-685.

[32] Jang, J.-S. R. and C.-T. Sun (1995) Neuro-Fuzzy Modeling and Control. *The proceedings of the IEEE*, **83** (3) : 378-406. March 1995.

[33] Johansen, T. A. (1994) *Operating Regime based Process Modeling and Identification, PhD thesis.* Department of Engineering Cybernetics, Norwegian Institute of Technology, N-7034 Trondheim, Norway.

[34] Johansen, T. A. (1996) Identification of non-linear systems using empirical data and priori knowledge - an optimization approach. *Automatica*, **32** (3) : 337-356.

[35] Johansen, T. A. (1996a) Robust Identification of Takagi-Sugeno-Kang Fuzzy Model using Regulariztion . *Proceedings of IEEE conference on Fuzzy systems*, New Orleans. September 1996.

[36] Katayama, R.,Y. Kajitani, and Y. Nishda (1994) A self generating and tuning method for fuzzy modeling using interior penalty method and its application to knowledge acquisition of fuzzy controller. *In: Kandel A. and G. Langholz (Eds.) Fuzzy control systems, Chapter 9*, pp. 197-224. CRC Press., Florida, USA.

[37] Kaymak, U., R. Babuška and H. R. van Nauta Lemka (1995) Fuzzy control - Theory and Design. *Journal A*, **36** (3): 4-14.

[38] Klir, G. J. and B. Yuan (1995) *Fuzzy sets and fuzzy logic: Theory and applications.* Prentice-Hall, Engelwood Cliffs, NJ, USA.

[39] Kosko, B. (1992) *Neural Networks and Fuzzy systems - a dynamical systems approach to machine intelligence.* Prentice Hall, Englewood Cliffs, NJ, USA.

[40] Krijgsman, A. J. (1993) *Artificial Intelligence in Real-Time Control, PhD thesis.* Delft University of Technology, Delft, The Netherlands.

[41] Lee, C. C. (1990) Fuzzy logic in control systems: Fuzzy logic controller, part I and II. *IEEE Trans. Syst., Man, Cybern.*, **20** : 404-435.

[42] Lin, C. T. and C. S. G. Lee (1991) Neural-network-based fuzzy logic control and decision system. *IEEE Trans. Computers*, **40** : 1320-1336.

[43] Ljung, L. (1987) *System identification - Theory for the* user. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, USA.

[44] Ljung, L. and T. Stöderström (1983) *Theory and Practice of Recursive Identification.* MIT press., Cambridge, Massachusetts, USA.

[45] Mamdani, E. H. and S. Assilian (1975) An experiment in linguistic synthesis with a fuzzy logic controller. *Int. J. Man-Machine Studies*, **7** : 1-13.

[46] Mizumoto, M. (1994) Fuzzy controls under product-sum-gravity methods and new fuzzy control methods.. *In: Kandel A. and G. Langholz (Eds.) Fuzzy control systems, Chapter 13*, pp. 275-294. CRC Press., Florida, USA.

[47] Myers, R. H. (1986) *Classical and modern regression with applications.* Duxbury Press., Boston, Massachusetts, USA.

[48] Narendra, K. S. and K. Parthasarathy (1990) Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Trans. Neural Networks*, **1** : 4-27.

[49] Orr, M. J. L. (1996) Introduction to radial basis function networks. *Research report 67 pages*, Center for Cognitive Science, University of Edinburg. Available via http://www.cns.ed.ac.uk/people/mark.html.

[50] Pedrycz, W. (1993) *Fuzzy Control and Fuzzy Systems.* John Willey and Sons, New York, USA.

[51] Ross, T. J. (1995) *Fuzzy Logic with Engineering Applications.* McGraw-Hill, Inc.

[52] Sjöberg, J. (1995) *Non-linear system identification with neural networks. PhD thesis.* Department of Electrical Engineering, Linköpling University, S-581 83 Linköpling, Sweden.

[53] Sjöberg, J., H. Hjalmarsson, and L. Ljung (1994) Neural Networks in System Identification. *Preprints of the 10th IFAC symposium on system identification*, Copenhagen, Denmark. July 4-6, 1994. pp.49-71

[54] Sjöberg, J., Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.-Y. Glorennec, H. Hjalmarsson and A. Juditsky (1995) Nonlinear Black-box Modeling in System Identification: a Unified Overwiew. *Automatica*, **31** (12) : 1691-1724.

[55] Stone, M. (1977) An asymptotic equivalence of choice of model by cross-validation and Akaike's criterion. *J. R. Statist.*, **Soc. B** (39): 44-47.

[56] Sugeno, M. (1985) An introductory survey of fuzzy control. *Information Science*, **36**: 59-83.

[57] Sugeno, M. and M. Nishida (1985) Fuzzy control of model car. *Fuzzy stes and systems*, **16** : 103-113.

[58] Sugeno, M. and G. T. Kang (1986) Fuzzy modelling and control of multilayer incinerator. *Fuzzy sets and systems*, **18** : 329-346.

[59] Sugeno, M. and G. T. Kang (1988) Structure identification of fuzzy model. *Fuzzy sets and systems*, **28** : 15-33.

[60] Sugeno, M. and K. Tanaka (1991) Successive identification of a fuzzy model and its applications to prediction of a complex system. *Fuzzy sets and systems*, **42** : 315-334.

[61] Sugeno, M. and T. Yasukawa (1993) A fuzzy-logic-based approach to qualitative modeling. *IEEE Trans. Fuzzy Systems*, **1** (1) : 7-31.

[62] Takagi, T. and M. Sugeno (1985) Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. Syst., Man, Cybern.*, **15** : 116-132.

[63] Tap, R. F. (1996) Receding horizon optimal control of greenhouse climate. *In preparation.*

[64] Tap, R. F., L. G. van Willigenburg and G. van Straten (1995) Experimental results of receding horizon optimal control of greenhouse climate. *Acta Horticulturae*, **406** : 229-238.

[65] Tchamitchian, M., L. G. van Willigenburg and G. van Straten (1993) Short term dynamic optimal control of the greenhouse climate. *MRS Report 92-3, 22 pages*, Department of Agricultural Engineering and Physics, Wageningen Agricultural University, The Netherlands.

[66] Terano, T., K. Asai and M. Sugeno (Eds.) (1992) *Fuzzy Systems Theory and its applications*. Academic Press.

[67] Terano, T., K. Asai and M. Sugeno (Eds.) (1994) *Applied Fuzzy Systems*. Academic Press.

[68] Tien, B. T. (1995) Recurrsive prediction error algorithm for the NUFZY system. *MRS Report 95-10, 42 pages*, Department of Agricultural Engineering and Physics, Wageningen Agricultural University, The Netrherlands.

[69] Tien, B. T. and G. van Straten (1995) Neural-Fuzzy systems for non-linear system identification - orthogonal least squares training algorithms and fuzzy rule reduction. *Preprints of the 2nd IFAC/IFIP/EurAgEng Workshop on AI in Agriculture*, Wageningen, The Netherlands. May 29-31, 1995. pp.249-254.

[70] Tien, B. T. and G. van Straten (1996) Recursive prediction error algorithm for the NUFZY system to identify nonlinear systems. *Proceedings of the 9th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems IEA/AIE-96*, Fukuoka, Japan. June 4-7, 1996. pp.569-574.

[71] Tien, B. T. (1997) On the interpretation of two types of fuzzy rules. *Proceedings of the Second International ICSC Symposium on Fuzzy Logic and Applications ISFL-97*, Zürich, Switzerland, February 12-14, 1997. pp. 240-246.

[72] Tien, B. T. and G. van Straten (1997) A neuro-fuzzy approach to identify lettuce growth and greenhouse climate. *Artificial Intelligence Review - special issue of AI applications in Biology and Agriculture*, Kluwer Academic Publisher. To appear in 1997.

[73] Thompson, M. L. and M. A. Kramer (1994) Modeling chemical process using priori knowledge and neural networks. *AIChE J.*, **40**: 1328-1340.

[74] Udink ten Cate, A. J. (1983) *Modeling and (adaptive) control of greenhouse climates, PhD thesis*, Wageningen Agricultural University, Wageningen, The Netherlands.

[75] Udink ten Cate, A. J. (1985) Modeling and simulation in greenhouse climate control. *Acta Horticulturae*, **174**: 461-467.

[76] van Luenen, W. T. C. (1993) *Neural Network for Control: on knowledge representation and learning, PhD thesis.* Technology University Twente, Enschede, The Netherlands.

[77] van Henten, E. J. (1994) *Greenhouse Climate Management: An Optimal Control Approach, PhD thesis.* Wageningen Agricultural University, Wageningen, The Netherlands.

[78] Wang, L. X. (1994) *Adaptive Fuzzy Systems and Control : Design and* Stability Analysis. Prentice Hall, Englewood Cliffs, NJ, USA.

[79] Wang, L. X. (1995) Design and analysis of fuzzy identifiers of nonlinear dynamic systems. *IEEE Trans. Automatic Control*, **40** (1): 11-23.

[80] Wang, L. X. and J. M. Mendel (1992) Fuzzy basis function, universal approximation, and orthogonal least-squares learning . *IEEE Trans. Neural Networks*, **3**: 807-814.

[81] Wang, L. X. and J. M. Mendel (1992a) Generating fuzzy rules by learning from examples. *IEEE Trans. Syst., Man, Cybern.*, **23** (6): 1414-1427.

[82] White, H. (1990) Connectionist nonparametric regression : multilayer feedforward networks can learn arbitrary mappings. *Neural Networks*, **3**: 535-549.

[83] Yager, R. R. and D. P. Filev (1994) On a Flexible Structure for Fuzzy Systems Models. *In: Yager R. R. and L. A. Zadeh (Eds.) Fuzzy Sets, Neural Networks, and Soft Computing, Chapter 1*, pp. 1-28. Van Nostrand Reinhold, ITP publishing company, New York, USA.

[84] Young , P. C. (1984) *Recursive Estimation and Time-Series Analysis.* Springer-Verlag, New York, USA.

[85] Zadeh, L. A. (1965) Fuzzy sets. *Information and Control*, **8**: 338-353.

[86] Zadeh, L. A. (1973) Outline a new approach to the analysis of complex systems and decision processes. *IEEE Trans. Syst, Man, Cybern.*, **3** : 28-44.

[87] Zadeh, L. A. (1975) The concept of a linguistic variable and its applications to approximation reasoning - I. *Information Science*, **8**: 199-249.

[88] Zadeh, L. A. (1995) Fuzzy logic, Neural Networks and Soft Computing. *Journal A*, **36** (2): 21-28.

[89] Zimmermann, H. - J. (1994) *Fuzzy set theory and its applications (2nd edition).* Kluwer Academic Publishers.

SUMMARY

Most real-world processes have nonlinear and complex dynamics. Conventional methods of constructing nonlinear models from first principles are time consuming and require a level of knowledge about the internal functioning of the system that is often not available. Consequently, in such cases a nonlinear system identification procedure from observational data is a more attractive alternative. If the model structures to be investigated are purely chosen from a set of mathematically convenient structures, without incorporation of knowledge about the internal functioning, this is called black-box modeling. In case that some qualitative *a priori* information can be used in the above modeling procedure, it is sometimes referred to as gray-box modeling.

Artificial neural network models and fuzzy models are typical examples of black-box and gray-box modeling, respectively. They have the same property of parallel processing and both serve as universal function approximators to perform nonlinear mapping. Each of them has its own weak and strong points. The fuzzy model has a transparent knowledge representation but has restricted learning ability. A neural network model can easily learn from new data, but it is difficult to interpret the information contained in its internal configuration.

This thesis investigates how to construct an integrated neural-fuzzy model that can perform approximation of an unknown system via a set of given input-output observations. The result is the integrated neural-fuzzy model NUFZY, which combines the advantages of the above two paradigms, and concurrently compensates for their weaknesses. Thus, it has a transparent network structure and a self-explanatory representation of fuzzy rules.

The NUFZY system is a special type of neural network, which is characterized by partial connections in its first and second layers. Through its network connections the NUFZY system carries out a particular type of fuzzy reasoning. Also, the NUFZY system is functionally equivalent to a zeroth-order Takagi-Sugeno fuzzy model, so that it is an universal function approximator as well.

Two existing learning methods, i.e., the orthogonal least squares and the prediction error algorithms, can be applied directly to the developed NUFZY model. The former method, referred to as batch learning, can be used to detect redundant fuzzy rules from the prototype rule base and to find the weight parameters of the NUFZY model by one-pass estimation. The latter, referred to as recursive learning, allows a fast adaptation of parameters of the NUFZY model. Several practical examples with real data of agricultural problems, which address the tomatoes growth and the greenhouse temperature, have been presented in this thesis, showing the capability of the NUFZY system for modeling nonlinear dynamic systems.

Two questions concerning the integrated neural-fuzzy model are addressed by studying the equivalent T-S fuzzy model: how to obtain a linguistic interpretation of fuzzy rules deduced by learning from training examples, and how to incorporate *a priori* knowledge into the T-S fuzzy model.

It is found out that it is possible to have linguistic interpretations of the crisp consequent of the T-S fuzzy rules by transforming them into Mamdani - like fuzzy rules. A new parameter set, the consequent significance level, is associated to the consequent of each Mamdani fuzzy rule to form an extended Mamdani fuzzy model. This model has a more flexible modeling ability than the ordinary Mamdani fuzzy model and has a comparable model accuracy as that of the T-S fuzzy model.

Regarding the second question, an optimization approach is employed to systematically incorporate the *a priori* knowledge into the T-S fuzzy model. If the knowledge about the system behavior outside the identification data range is expressed in the form of a qualitative Mamdani fuzzy model, then this model can be incorporated in the objective function of the parameter estimation problem as an additional penalty term. Thus, the estimation of the parameters of the T-S fuzzy model from the identification data is constrained by the involvement of *a priori* knowledge. As a consequence, the resultant fuzzy model becomes more robust in the extrapolation domain. This approach can be extended to neural -fuzzy modeling without difficulty.

To conclude, the beauty of the integrated neural-fuzzy model, NUFZY, developed in this thesis is that it is a neural network, enabling the implementation of efficient learning algorithms in an easy way, and that it is a fuzzy model at the same time, allowing incorporation of priori knowledge and transparent interpretation of its internal network structure. So, among the various methods of nonlinear system identification, the NUFZY model can serve as an attractive alternative.

SAMENVATTING

De meeste processen in de praktijk hebben niet-lineaire en complexe dynamica. De conventionele methode om niet-lineaire modellen op te bouwen op basis van elementaire beginselen is tijdrovend, en vereist een mate van kennis over het intern functioneren van het systeem die vaak niet aanwezig is. Daarom is het in zulke situaties vaak aantrekkelijker modellen op te bouwen uit waarnemingsgegevens via een niet-lineaire systeem identificatie procedure. Indien de in aanmerking komende modelstructuren worden gekozen uit een verzameling van mathematisch handige structuren, zonder dat kennis over het intern functioneren daarbij wordt betrokken, dan spreekt men van 'zwarte doos' (black-box) modellering. In het geval dat wel enige kwalitatieve *a priori* informatie in de modelbouw kan worden meegenomen spreekt men van 'grijze doos' (gray-box) modellering.

Artificiële neurale netwerk modellen en fuzzy modellen zijn typische voorbeelden van 'zwarte' resp. 'grijze' modelbouw. Zij werken beide parallel, en van beide is aangetoond dat zij universele functie approximators zijn, zodat zij niet-lineaire afbeeldingen kunnen verzorgen. Elk heeft zijn eigen sterke en zwakke punten. Het fuzzy model heeft een transparante kennisrepresentatie, maar kent slechts een beperkte leermogelijkheid. Neurale netten daarentegen kunnen gemakkelijk bijleren als nieuwe gegevens beschikbaar komen, maar het is moeilijk om de informatie die in de interne structuur is opgeslagen te interpreteren.

In dit proefschrift wordt bestudeerd hoe een geïntegreerd neuraal-fuzzy model kan worden geconstrueerd waarmee een onbekend systeem uit gegeven ingangs- en uitgangswaarnemingen kan worden benaderd. Het resultaat is het geïntegreerd neuraal-fuzzy model NUFZY, dat de voordelen van bovengenoemde paradigma's in zich verenigt, en tegelijkertijd de zwakheden compenseert. Het NUFZY systeem heeft een transparante netwerkstructuur en een zichzelf verklarende weergave van fuzzy regels.

Het NUFZY systeem is een speciaal type neuraal net dat wordt gekarakteriseerd door een partiële verbindingsstructuur tussen de eerste en tweede laag. Door zijn netwerk structuur voert het NUFZY systeem een bepaald soort fuzzy redeneerwijze uit. Het is functioneel equivalent aan een nulde orde Takagi-Sugeno fuzzy model, zo dat het eveneens een universele functie approximator is.

Twee bestaande leermethodes, te weten het orthogonale kleinste kwadraten algoritme en het predictiefout algoritme, kunnen direct worden toegepast op het ontwikkelde NUFZY model. De eerste methode, die een ladingsgewijze leermethode is, kan worden gebruikt om overbodige regels uit de verzameling fuzzy regels van het prototype te elimineren, en om op een niet-iteratieve wijze de gewichtsparameters te vinden van het NUFZY model. De tweede methode, die een recurrente leermethode is, maakt het mogelijk de parameters van het NUFZY model snel aan te passen aan nieuwe omstandigheden. In het proefschrift worden enkele praktische voorbeelden gegeven met gegevens ontleend aan agrarische problemen - met name tomatengroei en kastemperatuur modellering - die laten zien wat het vermogen van het NUFZY systeem is voor het modelleren van niet-lineaire dynamische systemen.

Er zijn twee vragen betreffende het geïntegreerde neuraal-fuzzy model die kunnen worden beantwoord door het equivalente Takagi-Sugeno model te bestuderen: hoe kan men een linguistische interpretatie geven aan de fuzzy regels die ontstaan door training op beschikbare gegevens, en hoe kan men *a priori* kennis in het T-S fuzzy model verwerken.

Een belangrijke bevinding van dit proefschrift is dat het mogelijk is om een linguistische interpretatie toe te kennen aan de getalsmatig geformuleerde consequent van de T-S fuzzy regels door deze te transformeren in Mamdani-achtige fuzzy regels met een interpreteerbare verzameling van linguistische termen, waar aan de consequent van elke Mamdani fuzzy regel een nieuwe parameter is toegekend: het consequent significantie niveau. Het op deze manier uitgebreide Mamdani fuzzy model is flexibeler dan het gewone Mamdani fuzzy model, en heeft een vergelijkbare nauwkeurigheid als het T-S fuzzy model.

Voor de oplossing van de tweede vraag is een optimalisatie benadering toegepast teneinde de *a priori* kennis op een systematische manier in het T-S fuzzy model op te nemen. Kennis van het systeemgedrag buiten het gebied waarvoor directe meetgegevens beschikbaar zijn kan in een kwalitatief Mamdani model als additionele strafterm in de optimalisatiedoelstelling van de parameterschatting worden meegenomen. Aldus wordt bereikt dat de schatting van de parameters van het T-S fuzzy model uit de voor identificatie beschikbare gegevens wordt ingeperkt door de beschikbare voorkennis. Het gevolg is dat het uiteindelijke fuzzy model robuuster is in het extrapolatiedomein. Deze werkwijze kan gemakkelijk worden uitgebreid naar een neuraal-fuzzy model.

Ter afsluiting: het mooie van het in dit proefschrift ontwikkelde geïntegreerde neuraal-fuzzy model NUFZY is dat het enerzijds een neuraal netwerk is, wat het gemakkelijk maakt efficiënte leeralgoritmen toe te passen, terwijl het tegelijkertijd een fuzzy model is, waardoor aan de interne netwerkstructuur een interpretatie kan worden gegeven, en waardoor voorkennis kan worden ingebracht. Het NUFZY model kan derhalve een aantrekkelijk alternatief zijn voor andere methoden van niet-lineaire systeemidentificatie.

結論

大多數的實體系統都具有非線性及複雜的動態特性，因此，在傳統的方法中，依質能守恆第一定律而建立的非線性數學模型是相當費時費工的。再者，應用此方法於建立系統模型時所需有關系統內部機能的知識則通常是不足的。在此情況下，另一較引人注意的替代方式則爲利用實際可得的量測數據，以非線性系統辨識的方法來建立系統模型。如果所欲建立的系統模型，其架構僅由一組合宜的數學模式所形成，而且沒有利用任何有關系統內部機能的知識，那麼這種建立系統模型的方法，在系統辨識理論中稱爲黑箱模型法。如果在系統模型的建立過程中，導入定性的先前經驗知識，則此方式通常稱爲灰箱模型法。

類神經網路模型及乏晰模型分別是黑箱模型法及灰箱模型法的典型代表。對系統具有平行處理能力，是它們之間共有的特徵。另外，兩者皆可做爲泛用型的函數近似器，以執行非線性映射的功能。它們亦各有其長處及弱點。就乏晰模型而言，它所呈現的內涵知識具有相當的透明性，但它的學習能力卻有所限制。反之，類神經網路模型則能容易地從實例中學習，但從其網路的內部結構中，卻不易了解隱涵其中的有效資訊。

本論文探討如何建構一個整合性的類神經網路－乏晰模型，稱之爲NUFZY模型。本研究利用已知的輸入及輸出變數的實際量測值來建立NUFZY模型，使該整合性模型能夠近似地描述未知系統的動態特性。所發展出來的NUFZY模型同時結合上述類神經網路模型及乏晰模型的優點，亦即具有透明易解的內部網路結構，且其乏晰規則所表達的內涵知識則不言而明；同時，它們之間的弱點亦得以相互補償。

NUFZY模型可視爲一特殊的類神經網路，即三層前饋式網路，但其第一層與第二層的神經元節點之間僅有部份的連結。輸入的資訊經由網路間的連結及傳遞，使得NUFZY模型得以執行一特別形態的乏晰推論。此外，由於NUFZY模型的函數近似功能相當於一個零階的Takagi-Sugeno乏晰模型，因此，它同時可以視爲一個泛用型的函數近似器。

在類神經網路的研究中，現有的兩個學習演算法，即正交最小平方法及預測誤差法，均能直接應用於NUFZY模型。正交最小平方演算法可用來偵測NUFZY模型中多餘的乏晰規則，而且其網路的加權參數值僅需一次估算即可獲得。因此，此種學習演算法又通稱爲批次學習法。預測誤差法則可以迅速地調整NUFZY模型的參數值以適應系統周遭環境的變化，一般稱之爲遞迴式學習法。在本論文中，我們列舉數個農業方面的應用實例，其中包括番茄的生長模型及溫室內部溫控模型。結果顯示NUFZY模型足以模擬實際的非線性系統之動態特性。

另外，以Takagi-Sugeno乏晰模型做爲研究NUFZY模型的觀點，本論文亦探討其衍生而得的兩個問題。其一是，利用實際量測數值辨識而得的Takagi-Sugeno乏晰規則，如何賦予語意上的解釋？另一問題則是，在建立Takagi-Sugeno乏晰模型的過程中，如何系統化地利用及導入定性的先前經驗知識。

首先，研究結果顯示，欲將辨識而得Takagi-Sugeno乏晰規則轉換爲類似於Mamdani型式的乏晰規則是可行的。在這轉換的過程中，我們利用一組新的參數，稱爲後件部顯著水準，加諸於每一條一般化的Mamdani型式乏晰規則的後件部。如此一來，形成一個所謂的擴大型Mamdani乏晰模型，同時，在特殊的條件下，該模型可轉換爲一般的零階Takagi-Sugeno 乏晰模型。因此，Takagi-Sugeno 乏晰規則的後件部之單一實數值可以賦予語意上的解釋。此外，擴大型的Mamdani乏晰模型的模擬能力，比一般常用的Mamdani乏晰模型更具有彈性，且其模擬的精確度近似於Takagi-Sugeno乏晰模型。

其次，本研究利用一最佳化的方法，將定性的先前經驗知識，有系統地導入 Takagi-Sugeno 乏晰模型的建構過程中。如果關於系統特性的定性知識在實際可蒐集到量測值的範圍之外爲已知，而且此種定性知識可以表達爲類似於一般的Mamdani型式的乏晰規則。那麼，在估算Takagi-Sugeno乏晰模型的最佳化參數值的求解過程中，這些定性的知識及資訊便可視爲額外的限制條件。因此，所求得的Takagi-Sugeno乏晰模型的最佳參數值，便可將定性的先前經驗知識融入於模型的建構過程之中，而且其最後所獲致的乏晰模型，具有較強韌的外插預測能力。此一最佳化方法可以很容易地推展及應用於整合性的類神經網路－乏晰模型。

總而言之，本論文所發展的整合性NUFZY模型，其優點在於它可視爲一個類神經網路，它可以便利地運用現有的學習演算法；同時它也可視爲一個乏晰模型，使得先前的經驗知識易於融入所建構的系統模型之中，而且其內部的網路結構擁有透明的語意釋涵。因此，在諸多的非線性系統辨識方法中，整合性的NUFZY模型法不失爲一有用的替代方案。

CURRICULUM VITAE

Biing-Tsair Tien was born on August 31st, 1965 in Taipei, the capital of Republic Of China in Taiwan. In 1987, he completed his bachelor degree from the Department of Agricultural Machinery, National Taiwan University. After graduation, he continued his MSc study at the Institute of Agricultural Engineering of the same university. In 1989, he received the Master degree with a thesis 'Color sorting of citrus fruits'. In the same year, he also obtained the fellowship of Dr. Tomotake Takasaka of 1989. One of the published papers 'Development of color sorter for lemons' published by him and co-authors won the Annual Thesis Prize form the Society of Chinese Agricultural Engineering in 1990.

After two years (1989 - 1991) of army service, he joined the Department of Agricultural Machinery Engineering, National Taiwan University, as a lecturer from 1991 till 1993. During this period, he also attended research projects of 'Automation of Seeding Production' and 'Binding Machine for Cut Flowers'. In 1992, he was granted a Ph.D. scholarship (1993-1997) from the Ministry of Education of the government of Taiwan R.O.C. to study subjects relating to 'Agricultural Automation' in Europe. In September 1993, he started his Ph.D. programme at the Systems and Control Group of the Department of Agricultural, Environmental and Systems Technology (former Department of Agricultural Engineering and Physics), Wageningen Agricultural University, The Netherlands. The present thesis is the result of the four years Ph.D. programme in WAU.