UNIVERSITY OF ALBERTA

SELF-LEARNING PREDICTIVE CONTROL USING RELATIONAL-BASED FUZZY LOGIC

BY

MARY MARGARET BOURKE

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY.

IN

PROCESS CONTROL

DEPARTMENT OF CHEMICAL ENGINEERING

EDMONTON, ALBERTA

FALL 1995

ISBN 0-612-06188-4

ELSEVIER SCIENCE PUBLISHERS B.V.
COPYRIGHT TRANSFER

Formal written transfer of copyright from the author(s) to the publisher is required for each article. Such transfer enables the publisher to provide for the widest possible dissemination of the article in printed, electronic and other existing and future formats. Without this transfer of copyright, such activities on the part of the publisher and the dissemination of information may be limited. Transfer of copyright is also required to enable the publisher to act against plagiarism and other forms of copyright infringement. We therefore ask that you complete and return this form promptly. Your cooperation is essential and appreciated.

Copyright (including all rights thereunder and the right to obtain copyrights registration in the name of the publisher, whether separately or as part of a journal issue or otherwise) in the article (provisionally) entitled:

The complete resolution of cartesian products

to be published in the Journal *Fuzzy Sets and Systems*

is hereby transferred, throughout the world and for the full term and all extensions and renewals thereof, to:

ELSEVIER SCIENCE PUBLISHERS B.V.

The author(s) reserve(s) trademark rights and patent rights, if any.

The author(s) may use all or part of the information contained in this article in future, non-commercial works of the author's own, or, if the article is a 'work-for-hire' made within the scope of the author's employment, the employer may use all or part of the information contained in this article for intra-company use, provided acknowledgements are given regarding copyright notice and reference to the original publication.

The author(s) warrant(s) that the article is the author(s)'s original work, and has not been published before. If excerpts from copyrighted works are included, the author(s) has obtained or will obtain written permission from the copyright owners and shall credit the sources in the article. The author(s) also warrant(s) that the article contains no libellous or unlawful statements, and does not infringe on the rights of others.

If the article was prepared jointly with other author(s), the author has informed the co-author(s) of the terms of this copyright transfer and is signing on their behalf as their agent and represents that the author is authorized to do so. In the case of an employed author(s) acting within the scope of his employment ('work-for-hire') the employer or an authorized representative of the employer must sign.

Signature of copyright owner:
(author or corporate representative) ....Mary M. Bourke........

Name (printed): ....Mary M. Bourke........

Date: ....Feb. 14/94........

PLEASE RETURN THIS FORM PROMPTLY TO:

ELSEVIER SCIENCE PUBLISHERS B.V.
P.O. BOX 1991, 1000 BZ AMSTERDAM
THE NETHERLANDS

SSDI: *016501149400004Q*

FSS 1210

EvE 07-04-1992

ELSEVIER SCIENCE B.V.
COPYRIGHT TRANSFER

Formal written transfer of copyright from the author(s) to the publisher is required for each article. Such transfer enables the publisher to provide for the widest possible dissemination of the article in printed, electronic and other existing and future formats. Without this transfer of copyright, such activities on the part of the publisher and the dissemination of information may be limited. Transfer of copyright is also required to enable the publisher to act against plagiarism and other forms of copyright infringement. We therefore ask that you complete and return this form promptly. Your cooperation is essential and appreciated.

Copyright (including all rights thereunder and the right to obtain copyrights registration in the name of the publisher, whether separately or as part of a journal issue or otherwise) in the article (provisionally) entitled:

Calculation and application of a minimum fuzzy relational matrix

to be published in the Journal *Fuzzy Sets and Systems*

is hereby transferred, throughout the world and for the full term and all extensions and renewals thereof, to:

ELSEVIER SCIENCE B.V.

The author(s) reserve(s) trademark rights and patent rights, if any.

The author(s) may use all or part of the information contained in this article in future, non-commercial works of the author's own, or, if the article is a 'work-for-hire' made within the scope of the author's employment, the employer may use all or part of the information contained in this article for intra-company use, provided acknowledgements are given regarding copyright notice and reference to the original publication.

The author(s) warrant(s) that the article is the author(s)'s original work, and has not been published before. If excerpts from copyrighted works are included, the author(s) has obtained or will obtain written permission from the copyright owners and shall credit the sources in the article. The author(s) also warrant(s) that the article contains no libellous or unlawful statements, and does not infringe on the rights of others.

If the article was prepared jointly with other author(s), the author has informed the co-author(s) of the terms of this copyright transfer and is signing on their behalf as their agent and represents that the author is authorized to do so. In the case of an employed author(s) acting within the scope of his employment ('work-for-hire') the employer or an authorized representative of the employer must sign.

Signature of copyright owner:
(author or corporate representative) .... Mary M Bourke ....

Name (printed): .... Mary M Bourke ....

Date: .... Jan 15 /95 ....

PLEASE RETURN THIS FORM PROMPTLY TO:*

SSDI: 0165-0114(94)00340-8

FSS 1546

ELSEVIER SCIENCE B.V.
Journal ref. FSS
P.O. BOX 1991, 1000 BZ AMSTERDAM
THE NETHERLANDS

***WE REQUIRE THE ORIGINAL SIGNED FORM, NOT A FAXED COPY.**

EvE 07-04-1992

ELSEVIER SCIENCE B.V.

COPYRIGHT TRANSFER

Formal written transfer of copyright from the author(s) to the publisher is required for each article. Such transfer enables the publisher to provide for the widest possible dissemination of the article in printed, electronic and other existing and future formats. Without this transfer of copyright, such activities on the part of the publisher and the dissemination of information may be limited. Transfer of copyright is also required to enable the publisher to act against plagiarism and other forms of copyright infringement. We therefore ask that you complete and return this form promptly. Your cooperation is essential and appreciated.

Copyright (including all rights thereunder and the right to obtain copyrights registration in the name of the publisher, whether separately or as part of a journal issue or otherwise) in the article (provisionally) entitled:

Convergence, eigen fuzzy sets and stability analysis of relational matrices

to be published in the Journal *Fuzzy Sets and Systems*

is hereby transferred, throughout the world and for the full term and all extensions and renewals thereof, to:

ELSEVIER SCIENCE B.V.

The author(s) reserve(s) trademark rights and patent rights, if any.

The author(s) may use all or part of the information contained in this article in future, non-commercial works of the author's own, or, if the article is a 'work-for-hire' made within the scope of the author's employment, the employer may use all or part of the information contained in this article for intra-company use, provided acknowledgements are given regarding copyright notice and reference to the original publication.

The author(s) warrant(s) that the article is the author(s)'s original work, and has not been published before. If excerpts from copyrighted works are included, the author(s) has obtained or will obtain written permission from the copyright owners and shall credit the sources in the article. The author(s) also warrant(s) that the article contains no libellous or unlawful statements, and does not infringe on the rights of others.

If the article was prepared jointly with other author(s), the author has informed the co-author(s) of the terms of this copyright transfer and is signing on their behalf as their agent and represents that the author is authorized to do so. In the case of an employed author(s) acting within the scope of his employment ('work-for-hire') the employer or an authorized representative of the employer must sign.

Signature of copyright owner:
(author or corporate representative) Mary M. Bourke

Name (printed): Mary M. Bourke

Date: Aug 25/95

PLEASE RETURN THIS FORM* PROMPTLY TO:

SSDI: 0165-0114(95)00212-X

ELSEVIER SCIENCE B.V.
Journal ref. FSS
P.O. BOX 1991, 1000 BZ Amsterdam

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: Mary Margaret Bourke

TITLE OF THESIS: SELF-LEARNING PREDICTIVE CONTROL USING RELATIONAL-BASED FUZZY LOGIC

DEGREE: Doctor of Philosophy

YEAR THIS DEGREE GRANTED: 1995

Mary M Bourke

Mary M. Bourke

14615-103 Avenue
Edmonton, Alberta
T5N 0T6

Dated: Oct. 4/95

"The universe is strange. Not only is the universe stranger the we think, it may be stranger than we can think. One result is that the physicist, who must deal with the real world, has at least three logics: one for ordinary state variables, classical mechanics, one for the microscope, quantum mechanics, and one for the macroscopic, relativity theory.

Thus consistency gives way to utility. It is possible that one theory would handle all three cases, but it would probably be too complex to use.

We definitely need more logics to deal with uncertainty."

[Bellman and Zadeh, 1977]

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled

SELF-LEARNING PREDICTIVE CONTROL USING RELATIONAL-BASED FUZZY LOGIC

submitted by **MARY MARGARET BOURKE** in partial fulfillment of the requirements for the degree of Doctor of Philosophy in **PROCESS CONTROL**.

Dr. D.G. Fisher, Supervisor
(Chemical Engineering)

Dr. S.L. Shah, Examiner
(Chemical Engineering)

Dr. R.G. Wood, Examiner
(Chemical Engineering)

Dr. H.J. Hoover, Examiner
(Computing Science)

Dr. S.W. Willard, Examiner
(Mathematical Sciences)

Dr. W. Pedrycz, Examiner
(Electrical & Computer Engineering, University of Manitoba)

Dated: October 2, 1995

DEDICATION

I dedicate this thesis to:

My husband, Steve,
for his constant love, unfailing support and gentle companionship,

My daughter, Beth, and
My son, Adam,
both born during the course of this work,
for keeping me focused on what was really important,

Mon Héritage Français,

and to

Old Man River.

ABSTRACT

This thesis documents the development of a *Model-based Self-Learning Predictive Fuzzy Logic (MSPF) Controller* for use in applications where the inherent uncertainty in the process model and/or data precludes the use of conventional discrete control algorithms. This work required not only a translation of the concepts of discrete model-based control systems into the fuzzy domain but also significant extensions to *fuzzy logic theory.*

The extensions to fuzzy logic theory in this thesis pertain mostly to the *max-product* composition, which several authors have shown to produce *better* results than the widely used *max-min* composition. The superiority of the *max-product* composition was also confirmed in this thesis for a variety of process oriented applications. The new theory developed for the *max-product* composition includes *eigen fuzzy stability, powers of* ***R*** *stability*, an *estimate of a minimum* ***R***, and the *complete Cartesian product solution*, parts of which have been published in the *Fuzzy Sets and Systems* journal.

Since the *max-product* composition has not been used extensively, there was very little existing literature on effective identification algorithms for this composition. This thesis therefore reviews and compares several important fuzzy identification strategies for the *max-min* composition and then applies them using the *max-product* composition. Based on this work, a new identification algorithm was developed that is *better*, from a least squares perspective, than the existing algorithms when applies to the Box-Jenkins gas furnace data. The new identification algorithm also includes a new procedure that permits an identification algorithm to adapt quickly to process changes while maintaining a *complete* solution.

Most of the rule-based fuzzy logic controller designs in the literature are based on a ***PI*** controller structure. The development of the control algorithm in this thesis parallels that of conventional *k*-step-ahead model-based predictive controllers, but implementation is significantly different due to the fuzzy environment. An important feature of this new controller is that the control action is determined based on the *discrete* error between the output and the setpoint. Results from this thesis clearly show that minimization of a control objective defined by a fuzzy criterion does not imply minimization of the corresponding discrete criterion. Therefore the proposed *MSPF* controller is ideal for practical control applications because, in the majority of cases, the objective is discrete even though the methodology is fuzzy in order to handle the unavoidable uncertainties.

The *MSPF* controller gave *very good* closed-loop performance in simulation using underdamped, overdamped and non-linear processes plus processes with large time delays and/or disturbances. A direct comparison of the *MSPF* controller versus a conventional discrete ***PI*** controller using a very (smoothly) non-linear process showed that (based on minimization of the *discrete* control error) the *MSPF* controller gave *better* performances over the full operating domain than ***PI*** control even when three-level gain scheduling was used.

The development and evaluating of the *Self-Learning Predictive Fuzzy Logic Controller* described above is complemented by a extensive *fuzzy logic* tutorial which includes a literature survey and examples for each aspect of the controller development.

ACKNOWLEDGMENT

I would like to take this opportunity to thank Dr. Fisher to his support and encouragement during the course of this work. In particular, I appreciate his patience and understanding of the many diversions which paralleled my studies.

Merci.

I would like to express thanks to the all the graduate students and staff of the Department of Chemical Engineering who I have gotten to know through the course of my graduate studies and who have enriched this university experience. In particular, a special thanks and farewell to my offices mates over the years, Raj Mutha, Viral Maniar and Kent Qi.

I would like to thank the Department of Chemical Engineering, the Province of Alberta and the University of Alberta and the Natural Sciences and Engineering Research council of Canada for financial support received during the course of this thesis work.

Et finalement, je aimerais remercier ma famille, mon mari and mon amour, Steve, d'avoir cru en moi, ma fille, Beth, qui me fait chanter dans ma tête, et mon fils, Adam, qui est mon soleil.

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF TABLES

Chapter 2

Chapter 3

Chapter 5

Chapter 7

LIST OF TABLES

Chapter 8

Appendix 3

LIST OF FIGURES

Chapter 1

Chapter 2

Chapter 3

LIST OF FIGURES

Chapter 4

Chapter 5

Chapter 6

Chapter 7

LIST OF FIGURES

Chapter 7 (Cont'd)

Chapter 8

Chapter 9

LIST OF FIGURES

Chapter 9 (Cont'd)

Chapter 10

LIST OF FIGURES

Chapter 10 (Cont'd)

Appendix 1

Appendix 5

LIST OF SYMBOLS

Chapter 1

$\bar{A}$	complement of the set A
$I_A(x)$	indicator function of the component x of set A
$m_A(x)$	membership function of component x of set A
$t(S)$	truth value of S

Greek Symbols

μ	degree of membership

Other Symbols

$\varnothing$	the null or empty set

Chapter 2

a_1, a_2 etc.	are parameters defining triangular or trapezoidal membership functions
h_j	is the height of the i-th membership function (i.e. $h_j = \mu(y_j)$)
m	is the mean or is the number of maximal elements
S_i	area of the i-th membership function
w_f	is the width of the fuzzy membership function
y	is the defuzzified output
y_i	is the characteristic output value of the i-th reference fuzzy set

Greek Symbols

μ	degree of membership
$\mu_A(x)$	is the fuzzy membership function for A
σ	is the standard deviation
σ_n	is the standard deviation of the noise

LIST OF SYMBOLS

Chapter 3

$\bar{A}$	compliment of $A = 1 - A$
e	error
Δe	change of error
F	fuzzy subset
H	high
k	a specific time instant
L	low
M	medium
p	model order
p_i	possibility of belonging to the i-th fuzzy referential set
R, S	relational matrix
s	*t-conorm* operator
t	*t-norm* operator
$\mathcal{U}, \mathcal{V}$	universes of discourse
u	an object of the universe of discourse, continuous $\{u\}$ or discrete $\{u_i\}$
$\boldsymbol{u}$	fuzzy input vector
$\boldsymbol{x}$	fuzzy state vector
$\boldsymbol{y}$	fuzzy output vector
u_i	fuzzy input value
x_j	fuzzy state value
y_l	fuzzy output value
u	discrete input
$\tilde{z}$	fuzzy vector

LIST OF SYMBOLS

Chapter 3 (Cont'd)

Greek Symbols

τ	system delay
$\mu_F(u)$	a function which assigns to each element u of $\mathcal{U}$ a number in the interval [0,1] which specifies the grade of membership.
λ	arbitrary multiplier (i.e. $\in [0, 1]$)
ρ	number of fuzzy clusters
λ_{ijl}	possibility measure
ϕ_ρ	fuzzification operator
ϕ_ρ^{-1}	defuzzification operator

Other Symbols

$\cap$	intersection
$\cup$	union
$\times$	Cartesian product
$\hat{+}$	probabilistic sum (i.e. $= a + b - ab$)
	arithmetic sum (i.e. $= 1/2(a + b)$)
$\cdot$	algebraic product (i.e. ab)
$\wedge$	bold intersection (i.e. max(0, $a+b-1$))
$*$	an arbitrary binary operator
$\circledast$	an arbitrary binary composition
$\textcircled{t}$	*max-t* composition
$\textcircled{\alpha}$	alpha composition
$\mathcal{O}$	the family of composition operators
$\circ \in \mathcal{O}$	*max-min* or *max-product* composition

LIST OF SYMBOLS

Chapter 4

$\mathbf{I}$	the interval [0,1]
$\mathbf{n}$	negation
p	selection parameter for Yager operator
$\mathbf{s}$	*t-conorm* operator
$\mathbf{t}$	*t-norm* operator
$\bar{x}$	compliment (i.e. $\bar{x} = 1-x$)
x, y, z	variables $\in \mathbf{I}$

Other Symbols

∞	infinity

Chapter 5

a_i	components of $\boldsymbol{a}$
$\boldsymbol{a}$	fuzzy vector
$\boldsymbol{a}'$	alternate fuzzy vector $\boldsymbol{a}$
b_i	components of $\boldsymbol{b}$
$\boldsymbol{b}$	fuzzy vector
$\boldsymbol{b}'$	alternate fuzzy vector $\boldsymbol{b}$
i	index of $\boldsymbol{u}$
$\mathbf{I}_m$	interval of $\boldsymbol{u}$
j	index of $\boldsymbol{v}$
$\mathbf{J}_n$	interval of $\boldsymbol{v}$
k	index of $\boldsymbol{w}$
$\mathbf{K}_p$	interval of $\boldsymbol{w}$
n	cluster number of $\boldsymbol{u}$
$\boldsymbol{M}$	lower solution of $\boldsymbol{R}$
m	cluster number of $\boldsymbol{v}$

LIST OF SYMBOLS

Chapter 5 (Cont'd)

p	cluster number of $\boldsymbol{w}$
PP	peak pattern
q_{ij}	components of $\boldsymbol{Q}$
$\boldsymbol{Q}$	fuzzy relational matrix
r_{ij}	components of $\boldsymbol{R}$
$\boldsymbol{R}$	fuzzy relational matrix
$\boldsymbol{S}$	maximum fuzzy solution or relation result of delta operator
$\boldsymbol{S}_o$	result $\boldsymbol{S}$ matrix after deleting all zero columns vectors
$\boldsymbol{S}^*$	non-redundant $\boldsymbol{S}$ relational matrices
s_{il}	components of relational results of delta operator
s	t-conorm
t	t-norm
t_{ij}	components of $\boldsymbol{T}$
$\boldsymbol{T}$	fuzzy relational matrix
$\boldsymbol{x}$	fuzzy input
$\boldsymbol{y}$	fuzzy output
u_i	components of $\boldsymbol{u}$
$\boldsymbol{u}$	fuzzy set
$\mathcal{U}$	fuzzy universe of discourse
v_j	components of $\boldsymbol{v}$
$\boldsymbol{v}$	fuzzy set
$\mathcal{V}$	fuzzy universe of discourse
w_j	components of $\boldsymbol{w}$
$\boldsymbol{w}$	fuzzy sets
$\mathcal{W}$	fuzzy universe of discourse
$\boldsymbol{W}$	minimum fuzzy solution

LIST OF SYMBOLS

Chapter 5 (Cont'd)

Greek Symbols

α	alpha operator
ⓐ	alpha composition
$\hat{\alpha}$	alternate alpha operator
β	beta operator
σ	sigma operator
ⓢ	sigma composition
δ	delta operator
Γ	cardinality
Λ	set of non-empty lower solutions
Σ	minimum of maximum and minimum fuzzy solutions
ϕ	phi-fuzzy sets
Φ	union of all phi-fuzzy sets of an ***R*** and ***b***

Other Symbols

$\circ$	fuzzy composition (i.e. *max-min*, *max-product*, etc.)
$\bigvee_j$	maximum over j
$\wedge$	minimum
$\cap$	intersection
$\cup$	union
ⓣ	*max-t-norm* composition
$\emptyset$	empty set
$\times$	cartesian cross product
$\exists$	there exists
$\ni$	such that
$\forall$	for all

Chapter 5 (Cont'd)

Superscripts

$^{-1}$	inverse or transpose of a relational matrix
^{t}	transpose
$^{1, 2 \text{ etc.}}$	i-th fuzzy solution
$\wedge$	normal fuzzy vector

Subscripts

$_{i}$	solution associated with the i-th fuzzy vector element

Chapter 6

a_{ij}	adjoint matrix
a_1, a_2, etc.	miscellaneous fuzzy sets
C	fuzzy constraint
$C_{\alpha, k}$	is the controllability index
D	fuzzy vector
D^c	converged fuzzy vector
d_i	elements of fuzzy vector
$e(t)$	is the error $y_{sp}(t)$ - $y(t)$
e_n	is the error at the nth sampling instant, $y_{sp}(n)$ - $y(n)$
$E(x_k)$	fuzzy energy
G	fuzzy goal
$\mathbf{I}$	identity matrix
K_c	is the controller gain
K_p	is the proportional gain
K_I	is the integral gain
K_1, K_2, etc.	miscellaneous fuzzy controller tuning parameters
N	Negative

LIST OF SYMBOLS

Chapter 6 (Cont'd)

$p(t)$	is the controller output at time t
p_n	is the controller output at the nth sampling instant, $n = 1, 2, \ldots$
$\bar{p}$	is the controller bias
PP	peak pattern
P	Positive
$\boldsymbol{R}^c$	converged relational matrix
$\boldsymbol{R}, \boldsymbol{Q}, \boldsymbol{T}$	fuzzy relational matrices
r_{ij}	relational matrix elements
$S_\alpha(x_o)$	index of stability
t	t-norm operator
Δt	is the sampling period
$\mathcal{U}$	universe of discourse
u	discrete input
$\boldsymbol{u}$	fuzzy input
$\boldsymbol{u}_{ss}$	steady state fuzzy input
$w(x_i)$	is a mapping that takes into account the fuzzy set position on the universe of discourse
$\boldsymbol{W}$	combined goal, $\boldsymbol{G}$, and constraint, $\boldsymbol{C}$
$\boldsymbol{x}_o$	an arbitrary initial fuzzy state
$\boldsymbol{x}_k$	fuzzy state at k-th sampling instant
$\boldsymbol{y}$	fuzzy output
$\boldsymbol{y}_{opt}$	optimal fuzzy output
$y_{sp}(t)$	setpoint at time t
$y(t)$	output at time t
$y_{sp}(n)$	setpoint at sampling period n
$y(n)$	output at sampling period n
Z	Zero

LIST OF SYMBOLS

Chapter 6 (Cont'd)

Greek Symbols

α	$\in [0,1]$, fuzzy tuning factor or alpha cut parameter
Φ	a measure of fuzziness
τ	oscillation period of powers of fuzzy sets
τ_I	is the integral time or reset time
τ_D	is the derivative time
μ_j	membership value in *j*-th membership function
ψ	inverse of *t*-norm operator
Δ	change in ...
Γ	cardinality
Σ	minimum of maximum and minimum fuzzy solutions

Other Symbols

$*$	is a *t*-norm operator (e.g. *min*, *product*)
$\circledast$	is the corresponding inverse composition of the *t*-norm operator, $*$

Chapter 7

a_l	tuning parameters which adjusts the speed of learning
b_l	relative contribution of each of the n (for $l = 1, \ldots, n$) rules to $\tilde{y}_k$
$\lvert e_k \rvert$	the amplitude of error, based on a non-fuzzy difference
h	constant used to control the range of a_l
i	index associated with input or control fuzzy vector $\{1,2, ..., m\}$
j	index associated with output fuzzy vector $\{1,2, ..., n\}$
J	non-fuzzy or discrete minimization criteria
l	index associated with output fuzzy vector for first order system $\{1,2, ..., n\}$
p	order of fuzzy state space equation

LIST OF SYMBOLS

Chapter 7 (Cont'd)

q	power of the minimization criteria, or, miscellaneous index
Q	fuzzy minimization criteria
$\mathcal{R}$	set of all relations, $\boldsymbol{R}$, that satisfy a set of input-output data
$\boldsymbol{R}$	relational matrix
r_{ijl}	relational matrix membership values associated with $\boldsymbol{R}$
$\boldsymbol{S}$	relational matrix
s_{ij}	relational matrix membership values associated with S
$\mathcal{U}$	input or control universe of discourse
U_i	fuzzy input or control membership functions/referential fuzzy sets
$\boldsymbol{u}$	fuzzy input or control vector
u_i	fuzzy input or control membership value
$\mathbf{u}$	discrete input or control variable
W	length of identification window
$\mathcal{X}$	state universe of discourse
X_j	fuzzy state membership functions/referential fuzzy sets
$\boldsymbol{x}$	fuzzy state vector
x_j	fuzzy state membership value
$\mathbf{x}$	discrete state variable
$\mathcal{Y}$	output universe of discourse
Y_j	fuzzy output membership functions/referential fuzzy sets
$\boldsymbol{y}$	fuzzy output vector
y_j	fuzzy output membership value
$\mathbf{y}$	discrete output variable

Greek Symbols

ρ	number of referential fuzzy sets
μ	membership value
τ	time delay

LIST OF SYMBOLS

Chapter 7 (Cont'd)

Other Symbols

$\circ$	*max-min* operator
$\odot$	*max-product* operator
$\textcircled{\alpha}$	*maximum* inverse for the *max-min* composition
$\textcircled{\psi}$	*maximum* inverse for the *max-product* composition
$\textcircled{\sigma}$	union of the *minimum* inverse for both *max-min* and *max-product*
$\bar{\sigma}_*$	operator of the *estimate* of the *minimum*, $* = u$ for unrestricted, $* = n$ for normal,
$\textcircled{\bar{\sigma}}$	operator for the *estimate* of the *minimum* fuzzy relations, first order state space

Superscripts

k	the k-th iteration
$\sim$	minimum estimate or estimate of the relational matrix
$\wedge$	maximum relational matrix
$-$	minimum relational matrix
M	Mamdani identified relational matrix

Subscripts

k	the k-th input-output data pair $\{k = 1, 2, \ldots N\}$

Chapter 8

A	fuzzy set
a, b	upper and lower limits on an arbitrary variable x
a_s, b	probabilistic descent derivative factors
$\mathbf{a}_1, \mathbf{a}_2$	discrete model parameters

LIST OF SYMBOLS

Chapter 8 (Cont'd)

$\boldsymbol{B}$	fuzzy set
$\mathbf{b}_1, \mathbf{b}_2$	discrete model parameters
c	component of quasi-newton tuning factor
d_s, c	probabilistic descent derivative factors
$\boldsymbol{f}$	a fuzzy mapping
$\tilde{f}_1$	equivalent to $s_1\tilde{\boldsymbol{f}}$
i	index associated with input or control fuzzy vector $\{1,2, ..., m\}$
j	index associated with output fuzzy vector $\{1,2, ..., n\}$
$\boldsymbol{J}$	non-fuzzy or discrete minimization criterion
k	probabilistic descent tuning parameter
l	index associated with output fuzzy vector for first order system $\{1,2, ..., n\}$
$\boldsymbol{Q}$	fuzzy minimization criterion
$\boldsymbol{R}$	relational matrix
r_{ij}	membership values associated with $\boldsymbol{R}$
$s_\alpha\tilde{\boldsymbol{f}}$	the α-*section* of $\tilde{\boldsymbol{f}}$
$\mathbf{t}$	*t*-norm
$\mathcal{V}, \mathcal{U}, \mathcal{Y}$	universes of discourse
w	≥ 0, component of quasi-newton tuning parameter
$\boldsymbol{u}$	fuzzy input or control vector
u_i	fuzzy input membership value
$\boldsymbol{x}$	fuzzy state vector
x_*	($* = 1, 2, 3$) represents the minimum *y/r* ratio for the given row
$\mathbf{y}$	discrete output value
$\boldsymbol{y}$	fuzzy output vector
y_j	fuzzy output membership value

LIST OF SYMBOLS

Chapter 8 (Cont'd)

Greek Symbols

α — $\in [0,1]$, arbitrary minimum or cut on fuzzy set

α_k — quasi-newton tuning parameter for *k*-th iteration

α — ≥ 0 is the momentum term for neural learning

ε — probabilistic descent tuning parameter

η — $\in [0,1]$ is the learning rate or step size parameter for neural learning

μ — membership value

Other Symbols

$\circ$ — *max-min* or *max-product* composition

$\exists$ — there exists

$\ni$ — such that

$\forall$ — for all

$*$ — arbitrary binary or *t*-norm operator

$\circledast$ — is the appropriate inverse for the t operator

Superscript

derivative fuzzy mapping or estimate

k — *k*-th iteration

Superscripts

q — power of minimization criteria Q and J

k — *k*-th sampling instant

LIST OF SYMBOLS

Chapter 9

a	input tuning factor
d	process disturbance
e	error
E	eigen fuzzy matrix
$f()$	a function of
G	fuzzy gain matrix
G_c	controller transfer function
G_c^*	model-based controller transfer function
G_{ff}	feed forward controller transfer function
G_m	process model transfer function
G_p	process transfer function
k	*k*-th sample instant
K	intermediate fuzzy matrices
L	process load
N_2	prediction horizon
Q,	intermediate fuzzy matrix
R	fuzzy dynamic matrix
r	process residuals
u	discrete input
u	*fuzzy* input
u_{gain}	calculated gain input
u_{dync}	calculated dynamic input
y	discrete output
y	*fuzzy* output
y_{sp}	output setpoint

LIST OF SYMBOLS

Chapter 9 (Cont'd)

Greek Symbols

β	> 1 determines the aggressiveness of the input tuning factor.
γ	≥ 1 is a tuning parameter for the convergence rate
τ	is the time delay
ω	≥ 1 is a tuning parameter for the window of error averaging

Other Symbols

$\circ$	*max-min* or *max-product* composition
o	stands for the family of composition (i.e. *max-min*, *max-product*, etc.)
$\circledcirc$ (Ψ)	is the inverse operator
$\cap$	is the intersection

Superscripts

~	estimate

Subscripts

k	*k*-th sampling instant

Chapter 10

a	input tuning factor
a_1, a_2	discrete model parameters
b_1, b_2	discrete model parameters
e	error
G	gain matrix
J	non-fuzzy or discrete minimization criterion
Q	is the reboiler duty for the Shell Model

LIST OF SYMBOLS

Chapter 10 (Cont'd)

$\boldsymbol{R}$	relational matrix
S_j,	is the area under the j-th membership function.
$\mathcal{U}$	fuzzy universe of discourse
u_{gain}	calculated gain input
$\boldsymbol{u}$	fuzzy input or control vector
u_i	fuzzy input membership value
$\mathbf{X}$	is bottoms impurity for the Shell Model
$\mathcal{X}$	fuzzy universe of discourse
$\boldsymbol{x}$	fuzzy state vector
x_j	fuzzy state membership value
$\mathcal{Y}$	fuzzy universe of discourse
$\boldsymbol{y}$	fuzzy output vector
y_l	fuzzy output membership value
$\mathbf{y}$	discrete output value

Greek Symbols

α	varies the weighting between $\mathbf{u}_{gain}$ and $\mathbf{u}_{dync}$ in the final value of $\mathbf{u}$
β	varies the intensity of the tuning factor a for $\mathbf{u}_{dync}$
η	a filter parameter that varies the weighting between current actual error and predicted error
ε	is the difference tolerance (i.e. $\lvert \mathbf{y}_{sp} - \tilde{\mathbf{y}} \rvert \le \varepsilon$)
γ	is the tuning factor for the iterative input search algorithm
κ	denominator calculation for gain matrix identification
σ	sigmoid function
σ	standard deviation
τ	is the time delay
ω	a filter parameter that varies the length of the window over which the errors are averaged

LIST OF SYMBOLS

Chapter 10 (Cont'd)

Superscript

$^{-1}$	inverse or transpose of a relational matrix
$\sim$	derivative fuzzy mapping or estimate
k	k-th iteration

Superscripts

q	power of minimization criterion J
k	k-th sampling instant

CHAPTER 1 INTRODUCTION

"The closer one looks at a real-world problem, the fuzzier becomes its solution".

[Zadeh, 1973]

Conventional quantitative techniques currently employed in system analysis are not well suited for dealing with systems of a humanistic nature, or any system that has a complexity comparable to humanistic systems. Zadeh [1973] summarized this contention with his *principle of incompatibility*, which states, *"as the complexity of a system increases, our ability to make precise and yet significant statements about its behaviour diminishes until a threshold is reached beyond which precision and significance, or relevance, become almost mutually exclusive characteristics".*

With this perspective in mind, it is clear that traditional techniques of scientific analysis fail with humanistic systems because they are unable to mimic the fuzziness of human thinking and behaviour. What is required is a methodology which can tolerate data imprecision and model vagueness. And so enters *fuzzy logic.*

1.1 What is *fuzzy logic*?

Fuzzy logic is a concept that brings together the reasoning used by computers and the reasoning used by people. The concept of *fuzzy logic* was first presented by Zadeh [1965], known as the father of fuzzy theory. Zadeh contends that human thinking does not embrace precise definitions, but classes of definitions in which the transition from membership to non-membership is gradual rather than abrupt. The degree of membership is specified by a number between 1, full membership, and 0, non-membership.

The *fuzzy logic* approach has three distinguishing features:

(1) it uses *linguistic variables* in place of, or in addition to, *numerical values*

(2) simple relations between the variables are characterized by *conditional fuzzy statements*

(3) complete relations are characterized by *fuzzy algorithms*

The theoretical foundation of *fuzzy logic* is actually quite precise and mathematical. The source of imprecision in this approach is not the theory but the way the data is described. The *data* can be expressed vaguely, such as *high temperature, medium pressure* and take on linguistic quantifiers, such as *very high*. This information is then combined and processed methodically to produce a results which can be either *vague* or *precise.*

Fuzzy logic is not fuzzy. Fuzzy logic gets its name from the *fuzzy sets* which are the building blocks of the *fuzzy logic* system. These *fuzzy sets* differ from classical or crisp sets in that they allow for partial degrees of membership.

The *fuzzy set* has three principal features, as shown in Figure 1.1:

(1) the domain or range of values over which the fuzzy set is valid (the x-axis)

(2) the degree of membership axis (the y-axis)

(3) the fuzzy set function which maps the domain to the degree of membership

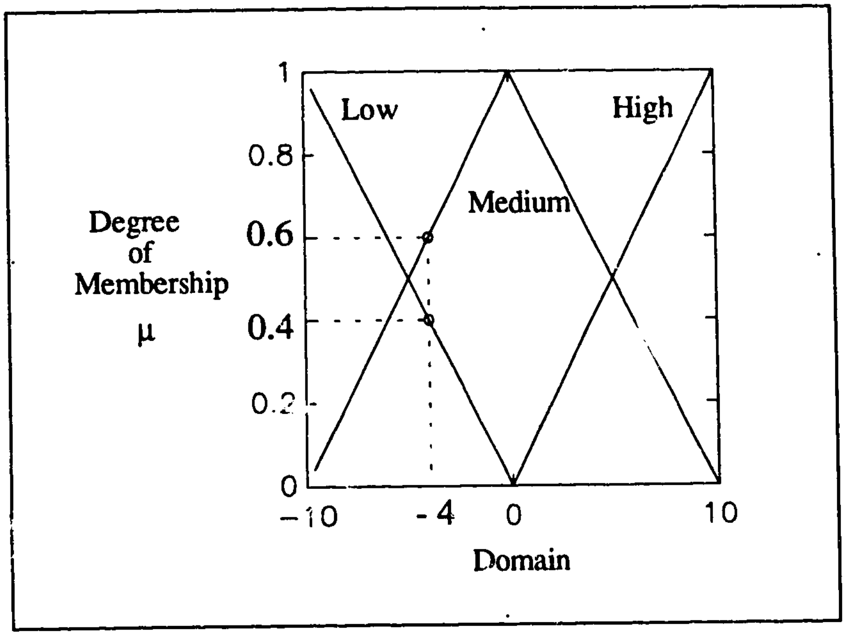


Figure 1.1: Fuzzy Membership Functions

The *fuzzy membership function* is a measure of the degree of compatibility of the object with its definition. So, if a value from the domain is known then the membership in the fuzzy set can be determined. For example from Figure 1.1, a discrete value of -4 has a membership or $\mu = 0.4$ in the fuzzy set low and membership or $\mu = 0.6$ in the fuzzy set medium.

Fuzzy logic involves gradual decision making. It attempts to model human thought processes or the *soft decisions* that occur in these processes. Often the fuzzy membership values are combined with membership values from other fuzzy states, through *union, intersection* and *complementation* operators, to produce a composite truth value. The fuzzy inferencing or implication has a mathematical basis so a fuzzy model can provide the same kind of discrete or deterministic result that is obtained from conventional knowledge-based systems.

Fuzzy logic is a *local* logic in that the truth-values determined, as well as, the connectives used (i.e. *and, or, if... then,...*) have a variable rather that fixed meaning. A distinctive feature of fuzzy logic is that the meaning of terms such as *beautiful, tall, small*, etc. are not only subjective but are also *local* in that they have a restricted meaning valid only in the domain that has been specified. Thus the meanings are not universally valid and apply only to the *local* problem [Bellman *et al.*, 1977].

Fuzzy logic is not a probability. Probability describes event occurrence. *Fuzzy logic* measures the degree to which an event occurs, not if it occurred. Both *fuzziness* and *probability* describe their uncertainly numerically with numbers in the unit interval [0,1]. They also both combine their sets and propositions through the operations of associativity, commutativity and distributivity. But probability theory and set theory require that :

$$A \cap \bar{A} = \varnothing \tag{1.1}$$

where $\varnothing$ is the null or empty set.

Fuzzy set theory, on the other hand, begins with the contradiction:

$$A \cap \bar{A} \neq \varnothing \tag{1.2}$$

The bivalent nature of equation (1.1) results in numerous paradoxes in which a statement S and its negation $\bar{S}$ have the same truth value, $t(S)$. A typical example is a card on which one side the statement is, *"The sentence on the other side is true"*. And on the other side the card says, *"The sentence on the other side is false"*. Considering this problem analytically, we have:

$$\begin{aligned} t(S) &= t(\bar{S}) \\ &= t(1 - S) \\ &= 1 - t(S) \end{aligned} \tag{1.3}$$

With classical set theory:

If $t(S) = 1$,

then $t(S) = 1 - t(S) \Rightarrow \quad 1 = 0$

which is a contradiction.

Fuzzy logic simply solves for the value of $t(S)$:

$$t(S) = 1 - t(S)$$

$$2t(S) = 1$$

$$\therefore \quad t(S) = 1/2$$

so the paradox is reduced to $1/2$ truths.

1.2 A Brief History

Many rule-based expert systems or applications of artificial intelligence are based on two-valued or classical Aristotelian logic, which as the name suggests, was developed by the philosopher, logician and scientist Aristotle about 400 B.C. The basic assumption of two-valued logic is that truth is two-valued, it is either true (i.e. has a membership of 1) or false (i.e. with a membership of 0). However, classifications in the real world often do not have these sharp boundaries. Consider the characteristics of *tallness* or *intelligence* both of which, in many cases, are only true to a degree. Classical two-valued logic is not designed to deal with properties that are a matter of degree. Three valued logic solved some of these vagueness issues with the categories; truth, falsehood and indeterminacy. Then entered multi-valued logic in which an attribute can be possessed to a degree (i.e. a person can be tall to degree 0.8 on a scale of 0 to 1).

The person who contributed the most toward the development of multi-valued logic was the Polish mathematician J. Lukasiewicz (also the inventor of reverse Polish notation used in Hewlett-Packard calculators). During the early 1920's Lukasiewicz extended the range of truth values from the three-valued logic $\{0, 1/2, 1\}$ to all rational numbers in $[0,1]$ and then finally to all numbers in $[0,1]$. In the 1930's quantum philosopher Max Black applied continuous logic to sets of lists of elements. Black also drew the first fuzzy-set membership functions and called the uncertainty of the structures *vagueness* In 1951, Menger, a French mathematician, coined the term *ensemble flou* which has become the French counterpart of *fuzzy set*, with *ensemble* meaning set, group, collection, series, and *flou* meaning blurred, hazy, vague, fuzzy, out of focus. But multi-valued logic systems were not used extensively because they did not go far enough. It was not until the landmark paper by Zadeh [1965], in which the work was expanded and given enough mathematical theory, that meaningful work with the concept of fuzzy sets could begin.

Zadeh extended the two-valued indicator function of a non-fuzzy set A of X,

$$I_A(x) = \begin{cases} 1 & if \quad x \in A \\ 0 & if \quad x \notin A \end{cases} \tag{1.4}$$

to a multi-valued indictor called a membership function,

$$m_A(x): X \rightarrow [0,1] \tag{1.5}$$

The membership value $m_A(x) = \text{degree}(x \in A)$

What differentiates fuzzy logic, as defined by Zadeh, from multi-valued logic is that in fuzzy logic one can deal with fuzzy quantifiers, also called linguistic hedges, like *very*, *somewhat*, *most*, *few* or *several*. Multi-valued logic has only two quantifiers, *all* and *some*. Another key difference between the two is that with fuzzy logic, truth itself can be fuzzy. So it is acceptable to say something is *quite true*. It is through these linguistic hedges that fuzzy logic provides a system which is flexible enough to serve as a framework for linguistic control.

1.3 Comparison of Fuzzy Systems to Conventional Expert Systems

When rule-based artificial intelligence (AI) was first conceived in the mid-1950's it was supposed to provide the ability to simulate human decision making in an uncertain environment. Based on symbol manipulation and first-order logic, AI served as the basis for expert systems and had some success, (e.g. game-playing systems and to a lesser extent natural language processing). However, AI has not been able to simulate common sense reasoning, and so has not lived up to its expectations. Conventional rule-based AI has not contributed significantly to the solution of real-world problems such as robotics, computer vision, speech recognition and machine translation and as a result has not led to a better understanding of thought processes, concept formation or pattern recognition [Kosko, 1992].

According to L.A. Zadeh, AI may have made more progress toward its original goals if it had not been so committed to symbol manipulation and first-order logic. Thus AI is unable to cope with methods that involve numerical computations, the way neural networks and fuzzy logic methods can, and cannot handle problems that involve uncertainty and imprecision. Unfortunately, most real-world problems fall into the latter category. Thus, there are a wide range of numerically-based real-world problems that conventional rule-based AI is unable to address or solve which are easily handled through the numerical techniques of neural network theory and fuzzy logic theory [Kosko, 1992].

In a typical rule-based AI system, the knowledge is acquired, stored and processed as symbols, not numerical entities. The collection of rules are defined as the *knowledge base*. The framework of the *if... then* rules is *chained* through, *firing* a premise only if it is true. This symbolic framework allows quick representation of structured knowledge but prevents numerical analysis. Knowledge is searched through logical paths of the knowledge tree via an inferencing process. Forward chaining through the rule base is input-data driven, answers *what-if* questions and generates or identifies effects. Backward chaining answers *how-come* or *why* questions and suggests a cause for an observed or specified effect. This *chaining* through large knowledge trees may be prohibitive for real-time processes, thus requiring approximate search strategies. These systems exploit structured knowledge when it is available, but in most cases the experts are unable to define the propositional rules in the format required to approximate the behaviour of the expert.

Neural networks evolved from work done in the late 50's and early 60's [Saleem *et al.*, 1994]. This AI modeling system introduced numerical processing, a change from symbolic processing, in order to reproduce the human thinking process.

Neural networks consist of numerous processing units, called *neurons*, which are connected by *synapses*. These networks can be trained to store, recognize and retrieve pattern information, to solve problems, to filter noise from measurement data, or, to provide a non-linear input-output relationship for ill-defined processes. Neural nets store patterns of information by a distributed encoding system. They superimpose the pattern information onto a *synaptic* web interconnecting the *neurons*. However, the system superimposes several input-output samples on the same synaptic black-box web, so it is difficult to tell what the system has *learned*, as well as, what it might *forget* with additional learning. Neural networks have an efficient numerical algorithm in order to exploit numerical information, but they cannot directly encode structured knowledge.

Fuzzy systems are newer that neural systems, having been introduced in the mid '60's. *Fuzzy logic systems* encode structured *common sense rules* or *principles* in a numerical framework. Each entry describes an input-output relation. These fuzzy logic rules can also be represented in a matrix format for numerical analysis. The fuzzy system *fires* each fuzzy rule in parallel, but to different degrees and then infers a conclusion from a weighted combination of the consequence from each *fired* rule. *Fuzzy systems*, like neural networks, have the ability to *learn* the system knowledge, using numerical or linguistic data as input, and produce an estimate of the input-output relationships.

Neural networks and fuzzy systems process inexact information inexactly. Neural nets recognize ill-defined patterns without an explicit set of rules. Fuzzy systems estimate relations and control systems with partial descriptions of system behaviour. Both neural nets and fuzzy systems are trainable systems that define input-output relations. They are *model-free* estimators in that they learn numerical and linguistic data from experience. Both systems use a numerical framework and encode sampled information in a parallel-distributed framework [Kosko, 1992].

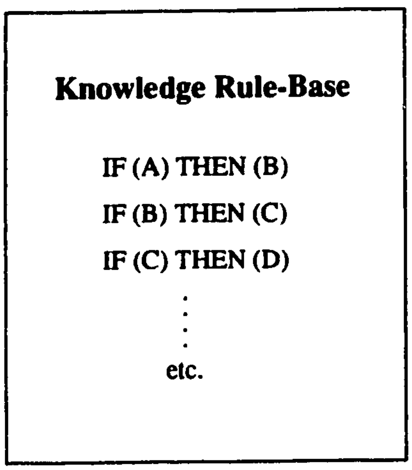
The main difference between neural networks and fuzzy logic is that fuzzy relational models contain the relationships between qualitative states and therefore represent the same type of qualitative models used in human reasoning. Neural networks on the other hand attempt to imitate the hardware involved in thinking. Thus results are generated from complex interactions between the network elements.

Figure 1.2 relates *fuzzy logic systems* to AI expert systems and neural networks. *Fuzzy logic systems* encompass both the structure knowledge of the expert system and the numerical framework of the neural systems, giving it the best of both designs.

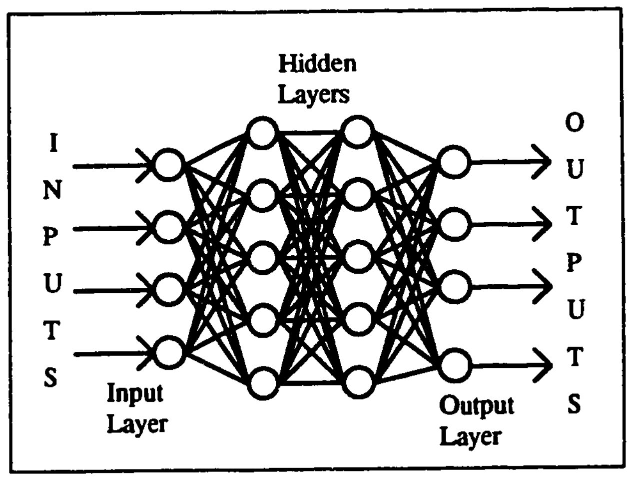
Figure 1.3 compares the architecture of the three systems just discussed. The strict *rule-based* format of conventional AI expert systems, the information processing of *fuzzy systems* versus the unstructured knowledge of neural nets. As shown in Figure 1.3, *fuzzy systems* are distinguished from neural networks in that there are no hidden layers, so each input can be related directly to each output.

	Symbolic Framework	Numerical Framework
Structured Representation	*AI Expert Systems*	*Fuzzy Systems*
Unstructured Representation	---------	*Neural Systems*

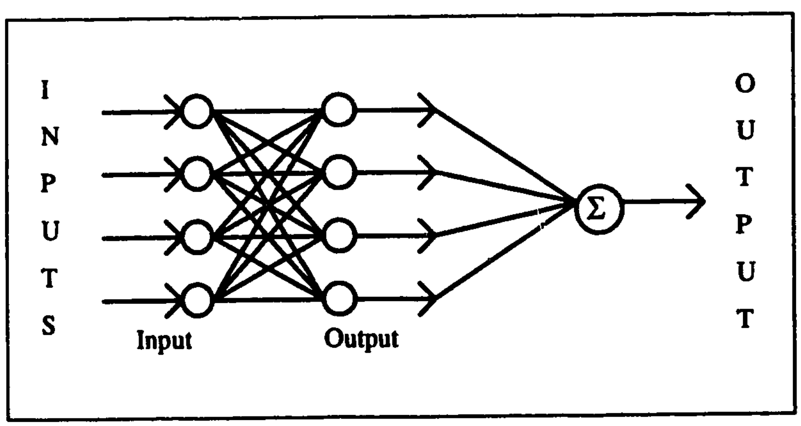
Figure 1.2: Fuzzy Logic vs. Neural Networks vs. AI Expert Systems [Kosko, 1992]



(a) AI Expert Systems



(b) Neural Networks [Treleaven, 1991]



(c) Fuzzy Logic Systems

Figure 1.3: System Architecture (a) AI Expert Systems, (b) Neural Networks, (c) Fuzzy Logic Systems

1.4 Motivation For Fuzzy Logic

The main purpose of *fuzzy logic systems* is to deal with systems which are inherently *fuzzy*. The *fuzziness* is usually due to lack of knowledge about the system or the distortions associated with the system. The main candidates for *fuzzy logic* control applications are those processes too complicated to be fully understood in terms of exact mathematical models and therefore incapable of being controlled in a precise way using classical control techniques. However, in many cases these processes are being controlled successfully by human operators using *rules of thumb* information. *Fuzzy logic systems* have demonstrated, since the mid-70's [Mamdani, 1974], the ability to deal with the goals and constraints that are required of these ill-defined *fuzzy systems*.

Examples of the need for *fuzzy technology* can be seen in industries or processes that have a vague measurement scale. Examples of fuzzy application areas from the recent literature are:

(*i*) ***Combustion Control of a Refuse Incineration Plant*** [Ono *et al.*, 1989]

A refuse incineration plant operation involves many kinds of uncertainty, such as, the variable physical properties of the refuse and the complexity of the burning process which may extinguish due to oversupply of refuse, or, experience poor combustion due to insufficient supply of refuse or oxygen. Attempts made with conventional control methods have resulted in extremely complex control logic which resulted in inadequate system robustness. The application of fuzzy logic to this situation resulted in stabilized combustion that matched that of an experienced operator.

(*ii*) ***Control of Biological Processes*** [Czogala *et al.*, 1989]

The cultivation of microorganisms is an important process in biochemical research, the fermentation industry and in the biological treatment of municipal waste. These processes are usually not well defined and often depend on unpredictable probability factors. Control of these ill-defined processes using classical control techniques is difficult and sometimes impossible due to the fact that:

- the biological mechanisms involved are not completely understood
- the measurement devices cannot be used for on-line measurements because the existing sensors are not reliable or accurate enough

A fuzzy application in this area was compared against open-loop response and PID control in bringing the unbalanced growth rate to steady state after a stepwise change in the dilution rate. Without control the new steady state was reached after 15 hours, with PID control steady state was reached in 5 hours. Application of fuzzy control further reduced this time to less than 3 hours with the additional benefit of minimal overshoot.

(iii) ***Meteorological Forecasting*** [Cao *et al.*, 1983]

Most of the meteorological phenomena is of a fuzzy nature, such as *today is hot*. There is no exact definition about which degree of temperature is considered *hot*. For weather forecasting, particularly long range forecasting, one usually predicts the trend of the future weather rather than specific numerical values.

Fuzzy forecasting was accomplished by collecting historical seasonal circulation patterns and partitioning these into membership functions. This information was stored year by year and forecasting was determined by the resemblance of the current patterns with the historical data. In spite of the simplicity of this approach, it proved quite effective.

(iv) ***Civil Engineering*** [Blockley, 1979]

The general public is prepared to accept the relatively high probability of death while driving in a car, but it expects an extremely low probability of death as a result of a bridge, over which a car may pass, collapsing. Even though there is a vast amount of scientific theory to help in making design decisions, there is still a large uncertainty concerning the application of the theories to the actual problem and rules of thumb are used extensively.

The actual likelihood of a structure failing is a function of one or more of several factors, such as,

- a random extremely high value of load or extremely low value of strength
- damage by an external random occurrence, (i.e. fire, earthquake)
- unknown or poorly understood system behaviour
- designer error
- error during construction
- misuse or improper alterations.

The current reliability theory used in civil engineering projects normally only deals with the first category and ignores the rest. However, through the use of fuzzy sets, an inclusive analysis with respect to all these factors can be undertaken by investigating past failures.

From these examples it can be seen that a problem area is a candidate for fuzzy logic if [Pedrycz, 1983]:

(1) the system being considered is complex or ill-defined

(2) there are major difficulties creating an exact mathematical model

(3) there is extensive experience and intuition available from process operators

(4) lack of measurements due to costs or noise makes it impossible to apply conventional statistical and/or control methods

So how is *fuzzy logic* applied to control situations? *Fuzzy systems* can be of two types: the more common *rule-based system*, or, a *relational-based system*, which permits numerical analysis.

In *Rule-Based Fuzzy Systems* a series of rules are developed that equate the fuzzy input membership functions to the fuzzy output membership functions. The rules can be formulated using the same *if ... then ... rules*, typical of expert systems, or by using *look-up-tables*, which consolidate all the *rule-base* information.

Rule-based systems are most commonly used in applications of fuzzy control. Other names for the *if... then* rules are production, premise-action or antecedent-consequent rules. The rules describe in qualitative terms how the controller output will behave when subjected to various inputs. The consequent part of the rule assigns a value to the output set, based on the conditional part of the statement. The degree of this assignment modifies the value of the output membership by applying to it the degree of truth of the conditional expression. Each rule produces a fuzzy output set, the union of which is the overall output.

The biggest problem with *rule-based systems* is obtaining the appropriate rules and once obtained to ensure that the rules are consistent and complete [Graham *et al.*, 1988]. Adaptive techniques are available for some types of applications which allow the rule-base to learn and self-modify [Graham *et al.*, 1989]. As well, genetic algorithm techniques allow the process to self-generate the rules [Karr, 1991(a), 1991(b)].

Fuzzy logic can be applied to control problems through the use of *fuzzy relational equations*, called *Relational-Based Fuzzy Systems*. These equations represent the relationships that exist between input and output states. A relational model is a matrix composed of values which represent the degrees of truth of all the possible cause-and-effect relationships that exist between the inputs and outputs. *The advantage of the relational-based system approach is that several techniques exists which allow the relational matrix to be identified directly from input-output data and provide a basis for NUMERICAL analysis of system properties.*

1.5 Focus of Thesis

Relational-based fuzzy systems will be the focus of this thesis because this representation provides a basis for formal numerical analysis of control concepts such as stability, controllability and reachability. *Fuzzy relational equations* provide a state space problem formulation and this structure lends itself to both *system identification* and to *goal oriented calculations* of control policy.

After considering the similarities between *fuzzy control* and conventional control the thesis compares the applicability of the standard fuzzy composition of *max-min* versus the often superior *max-product* composition. Stability theory for the *max-product* composition is established as a basis for practical applications of the proposed controller.

Fuzzy relational equations have a non-unique inverse, as outlined in Chapter 5. The initial thesis investigation focused on relational inverses as a possible basis for designing model based controllers. However, the inverse formulation proved to be unsuitable for process control because of the possibility of non-existent solutions. Iterative search or optimization techniques were therefore used in place of formal relational inverses. During the study of the relational inverses, the theory was expanded in the area of *fuzzy cartesian products* for both the *max-min* and *max-product* compositions as a potential basis for on-line identification.

Fuzzy identification and *fuzzy objective function optimization* are considered as part of the *self-learning* or *adaptive fuzzy logic control* being developed in this thesis. *Fuzzy identification* for relational equations is a relatively new area with publications dating back only to the early 1980's. Several of the identification and optimization algorithms currently available in the literature are reviewed and a new identification algorithm is developed.

Finally, a *predictive fuzzy logic controller* is presented and tested using simulated plant data. Results of the simulations presented show that the controller is capable of both servo and regulatory control in noisy environments.

The *fuzzy logic* controller development is from a strictly conventional control perspective. Someone with a very basic understanding of fuzzy logic theory and a strong interest in control can start reading this thesis at Chapter 9, which presents the parallel *fuzzy*/conventional controller development, and then continue on with Chapter 10, which presents the applicational results of the *fuzzy* controller. Chapters 2 to 8 contain some original contributions but, in a sense, are a *fuzzy* tutorial for the reader which includes *fuzzy logic control theory* and a literature survey for each aspect of the controller development.

REFERENCES

[1] Bellman, R.E., Zadeh, L.A., "Local and Fuzzy Logics", In: *Modern Uses of Multiple Valued Logic*, D. Reidel Publishing Co., Dordrecht-Holland, pp. 103-1[illegible], 1977.

[2] Blockley, D.I., "The Role of Fuzzy Sets in Civil Engineering", *Fuzzy Sets and Systems*, Vol. 2, pp. 267-278, 1979.

[3] Cao, H., Chen, G., "Some Applications of Fuzzy Sets to Meteorological Forecasting", *Fuzzy Sets and Systems*, Vol. 9, pp. 1-12, 1983.

[4] Czogala, E., Rawlik, T., "Modeling of a Fuzzy Controller with Application to the Control of Biological Processes", *Fuzzy Sets and Systems*, Vol. 31, pp. 13-22, 1989.

[5] Cox, E., "The Great Myths of Fuzzy Logic", *AI Expert*, pp. 40-45, January, 1992. (Erratum: *AI Expert*, pp. 5, February, 1992.)

[6] Graham, B.P., Newel, R.B., "Fuzzy Identification and Control of a Liquid Level Rig", *Fuzzy Sets and Systems*, Vol. 26, pp. 255-273, 1988.

[7] Graham, B.P., Newell, R.B., "Fuzzy Adaptive Control of a First-Order Process", *Fuzzy Sets and Systems*, Vol. 31, pp. 47-65, 1989.

[8] Karr, C., "Genetic Algorithms for Fuzzy Controllers", *AI Expert*, pp. 26-33, Feb. 1991(a).

[9] Karr, C., "Applying Genetics to Fuzzy Logic", *AI Expert*, pp. 38-43, Mar. 1991(b).

[10] Kiszka, J.B., Kochanska, M.E., Sliwinska, D.S., "The Influence of some Fuzzy Implication Operators on the Accuracy of a Fuzzy Model", *Fuzzy Sets and Systems*, Part I: Vol. 15, pp. 111-128, 1985; Part II: Vol. 15, pp. 223-240, 1985.

[11] King, P.J., Mamdani, E.H., "The Application of Fuzzy Control Systems to Industrial Processes", *Automatica*, Vol. 13, pp. 235-242, 1977.

[12] Konopasek, M., Jayaraman, S., "Expert Systems for Personal Computers [illegible] *Byte*, pp 137-1154,1984.

[13] Kosko, B., *Neural Networks and Fuzzy Systems*, Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1992.

[14] Mamdani, E.H., Assilian, S., "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", *International Journal of Man-Machine Studies*, Vol. 7, pp. 1-13, 1975.

[15] Ono, H., Ohnishi, T., Terada, Y., "Combustion Control of Refuse Incineration Plant by Fuzzy Logic", *Fuzzy Sets and Systems*, Vol. 32, pp. 193-206, 1989.

[16] Saleem, R.M., Postlethwaite, B.F., "A Comparison of Neural Networks and Fuzzy Relational Systems in Dynamic Modeling", *Proceeding from IEE, Control' 94*, pp. 1448-1452, March, 1994.

[17] Treleaven, P., "Introduction to Neural Computing", *European Control Conference*, Grenoble, France, pp. 1040-1041, July, 1991.

[18] Zadeh, L.A., "Fuzzy Sets", *Information and Control*, Vol. 8, pp. 338- 353, 1965.

[19] Zadeh, L.A., "Fuzzy Algorithms", *Information and Control*, Vol. 12, pp. 94-102, 1968.

[20] Zadeh, L.A., "Outline of a New Approach to the Analysis of Complex Systems and Decision Process", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 3, pp. 28-44, 1973.

[21] Zimmermann, H.-J., *Fuzzy Sets Theory - and Its Applications*, Kluwer-Nijhoff Publishing, Boston, 1985.

CHAPTER 2 *FUZZY* RULE-BASED SYSTEMS

The bottom line is that fuzzy rule-based systems are simpler, cheaper and more robust than their conventional counterparts.

[Zadeh, 1992]

2.1 Introduction

Lee [1990] presents an excellent review of the research-to-date on rule-based fuzzy logic controllers. This two paper series by Lee [1990] analyzes the structural parameters of rule-based controllers, including a brief summary of fuzzy set theory and fuzzy logic. The concept of a fuzzy logic controller is presented with strategies for fuzzification and defuzzification. As well, the paper discusses the construction of a data base for a fuzzy logic controller. The section on rule bases explains the derivation of fuzzy control rules and techniques for rule-modification. Also discussed are basic aspects of decision-making logic, including definitions for fuzzy implication, compositional operators, the interpretations of connectives and fuzzy inferencing mechanisms. A brief overview of some recent industrial and laboratory applications of fuzzy logic controllers is also given.

Although relational systems, the basis of this thesis, are not discussed in depth in the papers by Lee [1990], there is a brief mention of this methodology and some references are given. Some of the information provided in the papers by Lee [1990], such as, fuzzy implication, compositional operators, interpretation of connective and fuzzy inferencing mechanisms can be applied to both relational-based systems and rule-based systems.

Most of the commercial applications of fuzzy logic available today involve either rule-based methodology or look-up table implementations. As well, most introductory level publications on fuzzy logic controllers discuss these designs, so these are the concepts most universally understood. Chapter 2 will review the development of a fuzzy rule-based control system, since most commercial applications of fuzzy technology involve this structure. As well, rule-based structure reveal more of the inner workings of the fuzzy inferencing used to obtain a solution, and is therefore a good starting point to the understanding of fuzzy relational-based systems.

2.2 *Fuzzy Rule-Based* Controller Design

Design of a fuzzy logic controller, whether rule-based or relational-based involves:

(1) specification of the universe of the operating variables (operating range)

(2) specification of the terms or adjectives of the linguistic variables (membership function definition)

(3) the fuzzy mapping that relates specific values of the operating variables to the linguistic variables (fuzzy model)

2.2.1 Operating Range

The specification of the entire operating range of each input/output variable is known as defining the universe of discourse. This operation must cover the complete operating range such that each operating variable can be expressed by a fuzzy linguistic definition.

2.2.2 Membership Function Definition

Once the operating range has been specified, the range is divided into subsets or reference ranges, known as referential fuzzy sets or membership function. Then the adjectives or linguistic terms associated with these subset ranges are specified for each operating variable. These adjective can be of the differential form:

- positive big
- positive small
- no change
- negative small
- negative big

The convention is to use an odd number of incremental adjective descriptions centered around a ZERO or NO CHANGE function. Braae *et al.* [1979] suggests typically 2-10 term sets or adjective with a compromise between flexibility (many terms) and simplicity (few terms). The user can also specify adjectives using absolute values such as:

- zero
- small
- medium
- large

Additionally, these adjectives can be quantized as:

2, 1, 0 -1, -2

in reference to the differential linguistic adjectives above. So the 2 ≡ positive big, 1 ≡ positive small, 0 ≡ no change, etc. This quantization feature is more compatible with computer implementation and manipulation. As well, it more readily lends itself to discritization or defuzzification of the output, or transformation into a fuzzy relational-based system.

The power of fuzzy logic is that one can deal with fuzzy quantifiers, also called linguistic hedges. The developer can specify any number of linguistic hedges that will operate on the base adjectives previously specified. Hedges such as:

- very
- somewhat
- not

The *very* hedge serves to concentrate the effect of the rule by squaring the membership function, $0 \leq \mu \leq 1$. So,

$$very\ small\ = small^2$$

as shown in Figure 2.1.

The *somewhat* hedge functions to dilate the effect of the rule by taking the root of the membership function. So,

$$somewhat\ large\ = large^{1/2}$$

as shown if Figure 2.2.

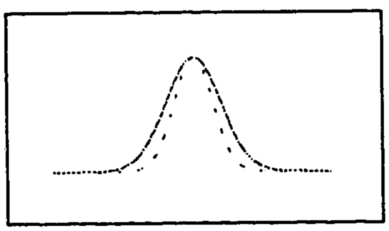


Figure 2.1: Fuzzy Concentration
small **(—);**
very small **(---)**

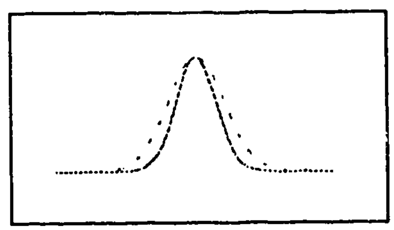


Figure 2.2: Fuzzy Dilation
large **(—);**
somewhat large **(---)**

The *not* is the complement of the membership function.

$$not\ medium\ = 1 - \ medium$$

Care must be taken when using the *not* operator as the complement may not imply the opposite, such as *not small* does not necessarily mean *large*.

The type or general shape of the membership function must be specified over the identified range. These shapes can be *quantized, continuous, triangular, trapezoidal* or *open-ended*.

(1) *Quantized*

With quantized membership functions the range of the variable is divided into a number of sub-ranges and the membership function for the fuzzy set consists of the grades of membership for each sub-range, as shown in the Figure 2.3.

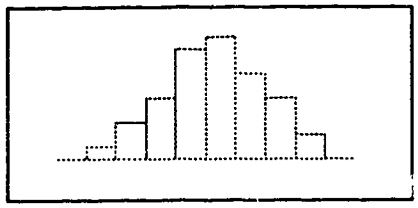


Figure 2.3: Quantized Membership Function [Postlethwaite, 1990]

Quantized membership functions can be represented in tabular form which have computational advantages.

(2) *Discretized*

Discretized membership functions also offer computational advantages because of their tabular form, shown in Table 2.1. Discretization is another ways of representing triangular or trapezoidal membership, which will be discussed below.

	(-2)	(-1)	(0)	(1)	(2)
Negative Big	1.0	0.6	0.0	0.0	0.0
Negative Small	0.5	1.0	0.3	0.0	0.0
Zero	0.0	0.5	1.0	0.5	0.0
Positive Small	0.0	0.0	0.3	1.0	0.5
Positive Big	0.0	0.0	0.0	0.6	1.0

Table 2.1: Quantized Membership Function [Czogala *et al.*, 1981]

(3) ***Continuous***

The continuous membership functions are usually bell-shaped, as shown in Figure 2.4, however other shapes are permitted.

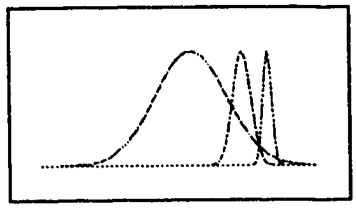


Figure 2.4: Continuous Membership Function

The normalized statistical formula for the normal distribution [Walpole *et al.*, 1978.], can generate various widths and means for a fuzzy membership function depending on the values of σ and m, respectively:

$$\mu_A(x) = \exp(-[(x - m)/\sigma]^2) \quad (2.1)$$

where	$\mu_A(x)$	is the fuzzy membership function for A
	σ	is the standard deviation
	m	is the mean

The advantage of using continuous functions is they allow for the inclusion of a wide range of values at low grades of membership. However, these functions produced *edge-effects* which cause difficulties during defuzzification. These problems will be discussed later in this Chapter.

(4) ***Triangular or Trapezoidal***

Triangular and trapezoidal functions are semi-continuous functions which have the advantage of being completely specified with three or four values, as Figures 2.5 and 2.6 illustrate.

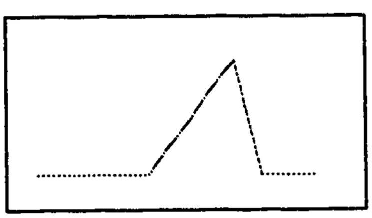


Figure 2.5: Triangular Membership Function
[Kaufmann *et al.*, 1988]

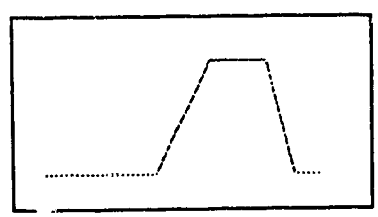


Figure 2.6: Trapezoidal Membership Function
[Kaufmann *et al.*, 1988]

A *triangular* membership function is defined as [Kaufmann *et al.*, 1988]:

$$\mu_A(x) = \begin{cases} 0 & x < a_1 \\ \dfrac{x - a_1}{a_2 - a_1} & a_1 \le x \le a_2 \\ \dfrac{a_3 - x}{a_3 - a_2} & a_2 \le x \le a_3 \\ 0 & x > a_3 \end{cases} \qquad (2.2)$$

A *trapezoidal* membership function is defined as [Kaufmann *et al.*, 1988]:

$$\mu_A(x) = \begin{cases} 0 & x < a_1 \\ \dfrac{x - a_1}{a_2 - a_1} & a_1 \le x \le a_2 \\ 1 & a_2 \le x \le a_3 \\ \dfrac{a_4 - x}{a_4 - a_3} & a_3 \le x \le a_4 \\ 0 & x > a_4 \end{cases} \qquad (2.3)$$

Note that a *triangular* function is a special case of the *trapezoidal* function with $a_2 = a_3$.

Pedrycz [1994] contends that not only are triangular fuzzy referential sets easy to work with, they also satisfy entropy equalization criteria, which means that on average the membership functions are activated to the same extent.

(5) *Open-Ended*

For continuous, triangular and trapezoidal shapes, the shape of the membership function can also be open-ended at either end of an infinite range. So for the case of the triangular or trapezoidal shapes, $\mu_A(x) = 1.0 \quad x \geq a_2$, as shown in Figure 2.7.

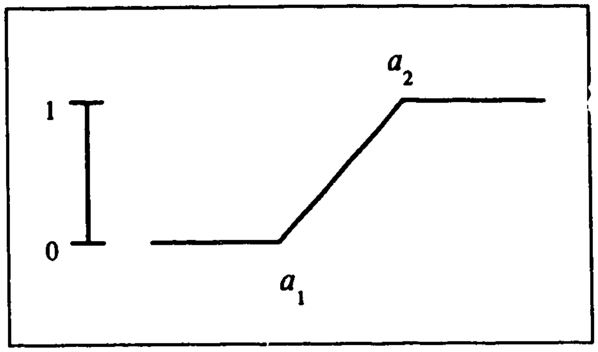


Figure 2.7: Open-Ended Membership Function

Li *et al.* [1989] concluded from their work that the amount of overlap of the fuzzy membership functions affected the efficiency of the fuzzy controller. Too much overlap results in many rules being applied to a single input. To little overlap resulted in difficulty deriving the lookup table.

Once the shape of the membership function has been decided, the size must be specified. As outlined in Braae *et al.* [1979] the range of the membership must be large enough so that measurement error and process disturbance will not effect process performance. These authors suggest that:

> *"the fuzzy sets be sufficiently wide to avoid undue noise transmission from the base variable to the linguistic variable"*

If the measurement noise of the inputs to the fuzzy logic controller can be characterized by a probability density function then the fuzzy sets should be selected such that:

$$w_f > 5\sigma_n \qquad (2.4)$$

where w_f is the width of the fuzzy membership function

σ_n is the standard deviation of the noise

This concept is illustrated in Figure 2.8.

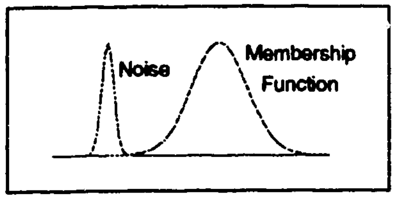


Figure 2.8: Fuzzy Noise [Braae *et al.*, 1979]

Additionally, the membership functions should be small enough to be within the process tolerance of the setpoint or quality specifications. The shape of the membership function should reflect the resolution required. High resolution (i.e. a narrow fuzzy set) is often required near zero to give a narrow dead-band in that region. Whereas low resolution (i.e. a wide fuzzy set) may be sufficient as the distance from zero increases. This concept is illustrated in Figure 2.9.

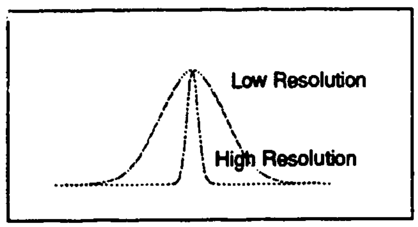


Figure 2.9: Fuzzy Resolution [Braae *et al.*, 1979]

There are clustering algorithms, such as FUZZY ISODATA [Bezdek, 1981] available for constructing the reference fuzzy sets from the available input-output data. However the usual drawback of using these algorithms is that the predictions based on clustering are always below the actual maximum values and above the actual minimum values [Valente de Oliveira, 1993].

2.2.3 The *Fuzzy Rule Model*

For the situation where a linguistic model does not exist data acquisition and model development must be undertaken. However, correlation of the actual input/output data is more applicable to relational-based systems. In many cases fuzzy *rule-based* controllers are applied to control processes which have already proven controllable at the hand of an operator. Since these process are already being controlled by linguistic *rules of thumb*, these rule can easily be translated into a *fuzzy rule base*.

Usually, the fuzzy control rules are generated by [Qiao *et al.*, 1992]:

(1) translating operator experience into a fuzzy linguistic form verbally
(2) monitoring and capturing the control behaviour of the operators
(3) modeling the process to be controlled, using fuzzy set theory
(4) self organizing or on-line rule learning during the running of the control system

Rule-Based Systems develop a series of rules that equate the linguistic value of the fuzzy input membership functions to the linguistic value of the fuzzy output membership functions. The rule base can be formulated as:

(1) *if ... then ...* rules typical of expert systems

(2) look-up tables [Rutherford *et al.*, 1976]

(1) *If ... then ... Rules*

Rule-based fuzzy systems deduce *fuzzy conclusions* from *fuzzy information* by inferencing that involves *if...then* rules. And, although the *if ... then ...* rules are typical of expert systems, the inferencing mechanism used for *fuzzy systems* differs from the typical expert system. For example, during the decision making process, the human brain often makes inferences in which fuzzy membership are involved. Such as the quality *hot* can be *somewhat hot* or *very hot*. Thus inferencing uses grades of belonging, so it can not be modeled using classical two-valued logic. For *fuzzy systems* two methods of inferencing are used, generalized modus ponens and generalized modus tollens.

The *generalized modus ponens fuzzy inference*:

Implication: If x is A, then y is B
Premise: x is A'

Conclusion: y is B'

In this example the x and y are the variables, such as *temperature*, *pressure*, or *error*, A, B, A' and B' are the linguistic labels over the universes of discourse of $\mathcal{X}$, and $\mathcal{Y}$, such as *small*, *medium*, and *large*. This inference reduces to *classical modus ponens* for $A' = A$ and $B' = B$.

The *generalized modus tollens* inference is:

Implication: If x is A, then y is B
Premise: y is B

Conclusion: x is A'

The *classical* inference for this case is also when $A' = A$ and $B' = B$.

The *rule-based system* is the most common application for fuzzy logic controllers. The *if...then...* rules are also called production, premise-action or antecedent-consequent rules. The rules describe in qualitative terms how the controller output will behave when subjected to various inputs. The consequent part of the rule assigns a quantitative value to the output set, based on the conditional part of the statement. The degree of this assignment modifies the output membership function by *capping* it with the degree of truth of the conditional expression. Each rule will produce a fuzzy output set, the union of which will be the overall output.

The biggest problem with rule based systems is obtaining the appropriate rules and the appropriate membership functions. Adaptive techniques are available which allow the rule-base to learn and self-modify [Graham *et al.*, 1988, 1989]. As well, genetic algorithm techniques allow the process to self-generate the rules [Karr, 1991 (a) & (b)].

A weighting for each rule can be determined for those cases when some rules may be more precise than others. Weighting the rules serves to enhance the impact of precise rule and dilute the effect of suspect rules. Additionally, weighting may be used to change the scale of the resolution of the fuzzy rules. As the system approaches the setpoint a finer control action may be implemented. Baldwin *et al.* [1980] discuss the dilution effect of a vague rule firing with a precise rule and present a method to overcome this problem.

The *min* ($\wedge$) and *max* ($\vee$) operators are used extensively in fuzzy logic theory when multiple premises and/or conclusions are specified, with the *min* corresponding to *and* and the *max* corresponding to *or*. The *or* is used in the case when for example either a large positive error or a medium positive error will result in the same control action. Fuzzy rule bases can be structured in a forward chaining manner for process monitoring or in a backward chaining manner to overcome time delay or for process prediction.

(2) ***Look-up Tables***

Rutherford *et al.* [1976] present a simple method of presenting the *fuzzy rules* called *look-up tables*, and shown in Figure 2.10. Many of the most successful fuzzy applications on the market today are simple look-up tables [Kosko, 1991].

Δe_t \ e_t (output u_t)	L	M	H
H	M	H	H
M	L	M	H
L	L	L	M

Figure 2.10: Look-up Table

Verifying rule completeness, for rule-base systems, is often accomplished through the use of look-up-tables [Rutherford *et al.*, 1976]. Additionally, Braae *et al.* [1976] discuss some issues on rule verification using graphical interpretation. Rule consistency can be determined during controller tuning and will be discussed later.

Composition of the fuzzy input subsets and rules to yield the fuzzy output universe can be obtained by {Kosko, 1991]:

(1) correlation-product inference

(2) correlation-min inference

as shown in Figures 2.11 and 2.12 below:

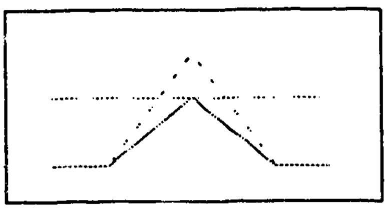


Figure 2.11: Correlation-Product Inference
[Kosko, 1991]

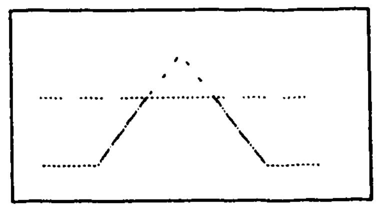


Figure 2.12: Correlation-Min Inference
[Kosko, 1991]

The *correlation-product* inference uses the membership value of the rule antecedent as a multiplier and *scales* the whole fuzzy output set maintaining the original shape. The formulation is as follows:

$$\mu_{ij} \quad = \quad \mu_A(x_i) \cdot \mu_B(x_j) \tag{2.5}$$

The *correlation-min* inference *clips* the resulting fuzzy output set at the membership value of the rule antecedent. The result of this method is that the *peaks* of the output fuzzy sets are eliminated and a flat surface results at the membership value of the antecedent. The formulation of this method is:

$$\mu_{ij} \quad = \quad \min\,(\,\mu_A(x_i),\, \mu_B(x_j)\,) \tag{2.6}$$

Comparing the two methods, the correlation-product preserves more information than correlation minimum, and so the correlation-product method is selected more often in fuzzy applications. These composition procedures apply to either *rule-based* or *relational-based* systems.

2.3 Defuzzification

The output, from both the *rule-based* and *relational-based* fuzzy systems, is itself a fuzzy set which may have to be converted to a *discrete* value for use as controller output. The procedure for this is called defuzzification or more recently *output interface* [Pedrycz, 1994].

The methods of defuzzifying the results of a fuzzy composition are discussed following [Mizumoto, 1989]. The first four (4) methods of defuzzification deal with the consolidated output fuzzy set of the inference. That is the resultant *j* output membership functions, either *clipped* by correlation-min inference or *scaled* by correlation-product inference, are overlaid on the output axis to form an overall output membership function, as illustrated in Figure 2.13.

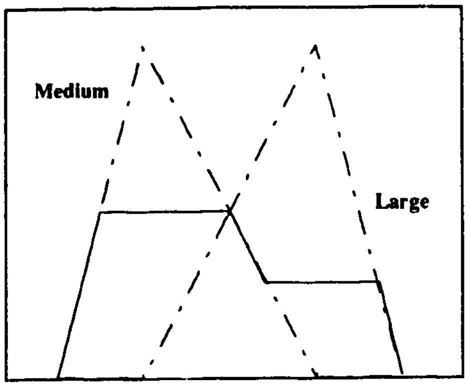


Figure 2.13: Resultant Output Membership Function [Postlethwaite, 1990]
(Overall Membership Function; ——)

(1) Average of Maxima

With the average of maxima method, if there is a single maximum membership value in the output membership function, the value of the output, y_i, at this point is taken as the defuzzified output. If the overall membership function is not unimodal and there are several points at which the membership function is at a maximum, then the defuzzified value is the average of all the output values where the membership function is at its maximum.

$$y = \frac{\sum_{i}^{m} y_i}{m} \tag{2.7}$$

where y_i is the output value where the membership value is a maximum

m is the number of maximal elements

(2) **Midpoint of Maxima**

This method is a simplified version of the average of maxima method. Instead of taking all the points, y_i, which give maximal membership value, the smallest element, y', and the largest element, y'', are averaged to calculate a midpoint.

$$y = \frac{(y' + y'')}{2} \tag{2.8}$$

(3) **Median or Center of Area**

The median or center of area method of defuzzification involves finding the value of output which defines the centre of the area under the resultant output membership function profile. This method allows for smoother changes in the output value over the center of gravity method, however, it is computationally more intensive.

Not only is this method computationally expensive but it is prone to problems with *edge-effects*. *Edge-effect* occur when a reference fuzzy set, at the end of the output range, extends to very high values or even infinity. In these cases, when the reference fuzzy set is fired, even with very low grades of truth, it pulls the center of the area towards itself. Since relation matrix identification often uses output reference fuzzy sets with these characteristics, the center of area method is not suited for relational matrix systems [Postlethwaite, 1990].

The next four (5) defuzzification methods deal with each output membership function separately. For these next definitions the value y_j is the output value of the j-th membership function where the membership value is a maximum.

(4) **Center of Gravity**

The center of gravity method is widely used in *fuzzy control*. Each output reference fuzzy sets is assigned a characteristic output value, y_i. For triangular reference fuzzy sets this value is usually taken to be the centre point. With other reference fuzzy shapes it is usually the value where the membership function is the largest. The defuzzification output is a weighted mean of the characteristic output values with the degree of membership in the relevant reference fuzzy set..

$$y = \frac{\sum_i y_i \cdot \mu(y_i)}{\sum_i \mu(y_i)} \tag{2.9}$$

where y is the defuzzified output

y_i is the characteristic output value of the i-th reference fuzzy set

μ is the degree of membership

(5) **Height**

This method obtains the *discrete* value as a weighted average of the resultant height, h_j, of the j-th membership function.

$$y = \frac{\sum_j y_j \cdot h_j}{\sum_j h_j} \tag{2.10}$$

where $h_j = \mu(y_j)$

(6) **Maximal Height**

The output value which corresponds to the maximum height, h_j, from among all the membership functions is considered the *discrete* output value.

$$y = h_j \tag{2.11}$$

where h_j is the maximal height

(7) **Area Method**

The area method obtains the output as the weighted average resultant area, S_j, of the j-th membership function.

$$y = \frac{\sum_j y_j \cdot S_j}{\sum_j S_j} \tag{2.12}$$

(8) **Maximal Area**

The representative output value, y_j, is considered the discrete output value, where j is the output membership function with the maximal area.

$$y = y_j \tag{2.13}$$

(9) Heuristic Rules

And finally, in some instances, defuzzification can be through additional heuristic rules. In the case where the membership function is convex and unimodal a crisp maximizing decision can be made. However, if the membership functions of the control statements are not unimodal a heuristic rule such as "*Take the action which is midway between two peaks or at the centre of a plateau*" can be used for defuzzification [Zimmermann, 1985].

All the methods above produce a continuous *discrete* output variable. In some cases the results may be required in a *fuzzy* form (i.e.. *large, medium, small, etc.*) in which case the continuous *discrete* output variable would be compared against a fuzzy range to produce a single fuzzy variable.

According to Tong [1980] there is no published evidence to suggest that any method is superior. In other instances, [Xu *et al.*, 1987] defuzzification can be consider as part of the minimization criteria for the model structure of the specific application.

2.4 Tuning

A number of difficulties are encountered by the designer of a fuzzy logic controller. First of all, for an even moderately complex process the number of rules required to obtain adequate control in all operating regions may be extremely large. The task of obtaining all of these rules from the process operator can itself be a formidable task. Secondly, once the initial rule set has been established the designer must ensure that the rules are consistent and complete. And finally, the rules must be tuned so that the overall performance is optimal (or at least satisfactory).

Consistency means that no two rules are in conflict such that they have the same antecedents but a different consequent. A rule set is complete when every possible input state has a membership function value greater than an arbitrary cutoff level (e.g. 0.1) in at least one rule.

Tuning challenges include [Li *et al.*, 1989]:

- completeness of the rule base
- shape of the fuzzy linguistic functions
- overlapping of subsets
- practical methods for controller calibration

Tuning can be accomplished by [Li *et al.*, 1989]:

- changing the rules
- changing the range of the membership functions
- changing the shape of the membership functions
- changing the gains on the fuzzy input
- adjusting the deadband about the setpoint (if coarse and fine control is used)

or a combination of the above.

Insight into rule consistency can be gained by observing the profile of the output fuzzy membership functioned obtained after the rules have been fired for a given condition. The shape of this curve can be used to assess the quality of the control rules used. As illustrated in Figure 2.14, curve (a) shows a single strong peak indicating one dominant control rule. The two peaks of curve (b) show that two strong but contradictory rules are present. The flatness of curve (c) indicates an absence of a strong rule set for the condition. For both of cases (b) and (c) some modification of the control rules may be necessary to obtain good control [King *et al.*, 1977].

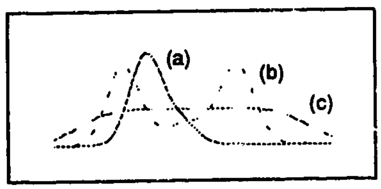


Figure 2.14: Rule Consistency [King *et al.*, 1977]

2.5 Advantages

The big advantage of fuzzy logic controllers is that they are simple in structure and are relatively easy to construct. They do not require the modeller to have an in-depth mathematical knowledge of the process, only intuition or experience with the process.

Many complex chemical processes are too complicated to be fully understood in terms of an exact mathematical model and/or cannot be control in a precise way using classical control techniques. However, these same processes may be successfully controlled by operators using *rules of thumb* and are the main candidates for *fuzzy logic control applications.*

2.6 Disadvantages

Some of the strengths of rule-based fuzzy logic controllers can also their greatest weaknesses. Because the process is not modeled in a conventional sense it is difficult to prove the stability of the system since stability analysis relies on the existence of a mathematical model. But if a discrete mathematical model were available the controller could have been designed using classical or modern control techniques.

Proving the stability of a fuzzy logic controller is one of the main concerns with fuzzy applications. Some developers feel that it is not a question of whether the controller is stable but if the process is stable under control. In most cases it is felt that if a process can be controlled successfully by a human operator it should be possible to implement fuzzy control.

Although fuzzy rule-based controllers are simple to construct, they are difficult to tune because of the large number of tunable features and the lack of quantitative performance measures. As much as 90% of the development time of a fuzzy logic controller is spent tuning the control rules [Karr, 1991(b)]. There also seems to be a lack of systematic methodology to optimize rule-based solutions other than trial-and-error or adopting search techniques such as genetic algorithms [Karr, 1991(a) & (b)].

2.7 Summary

Chapter 2 has presented a literature review of *fuzzy rule-based systems theory* as a starting point for the development of the *relational-based control system* presented in this thesis. The literature review consolidates the rule-based development theory from several sources. The format of this development is consistent with the relational-based theory, which is the focus of the thesis. Relational-based theory will be discussed in Chapter 3.

REFERENCES

[1] Baldwin, J.F., Guild, N.C.F., "Modeling Controllers using Fuzzy Relations", *Kybernetics*, Vol. 9, pp. 223-229, 1980.

[2] Bezdek, J.C., *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.

[3] Braae, M., Rutherford, D.A., "Selection of Parameters for a Fuzzy Logic Controller", *Fuzzy Sets and Systems*, Vol. 2, pp. 185-199, 1979.

[4] Braae, M., Rutherford, D.A., "Fuzzy Relations in a Control Setting", *Kybernetes*, Vol. 7, pp. 185-188, 1978.

[5] Manual for: *CubiCalc*, HyperLogic Corp., Version 1.1, 1991.

[6] Czogala, E., Pedrycz, W., "Some Problems Concerning the Construction of Algorithms of Decision-Making in Fuzzy Systems", *International Journal of Man-Machine Studies*, Vol. 15, pp. 201-211, 1981.

[7] Graham, B.P., Newell, R.B., "Fuzzy Identification and Control of a Liquid Level Rig", *Fuzzy Sets and Systems*, Vol. 26, pp. 255-273, 1988.

[8] Graham, B.P., Newell, R.B., "Fuzzy Adaptive Control of a First-Order Process", *Fuzzy Sets and Systems*, Vol. 31, pp. 47-65, 1989.

[9] Gupta, M.M., Qi, J., "Theory of T-Norms and Fuzzy Inference Methods", *Fuzzy Sets and Systems*, Vol. 40, pp. 431-450, 1991a.

[10] Karr, C., "Genetic Algorithms for Fuzzy Controllers", *AI Expert*, pp. 26-33, February, 1991(a).

[11] Karr, C., "Applying Genetics to Fuzzy Logic", *AI Expert*, pp. 38-43, March, 1991(b).

[12] Kaufmann, A., Gupta, M.M., *Fuzzy Mathematical Models in Engineering and Management Science*, Elsevier Science Publishers Co. Inc., 1988.

[13] King, P.J., Mamdani, E.H., "The Application of Fuzzy Control Systems to Industrial Processes", *Automatica*, Vol. 13, pp. 235-242, 1977.

[14] Kosko, B., *Neural Networks and Fuzzy Systems*, Prentice-Hall Canada, Inc., Toronto, 1992.

[15] Lee, C.C., "Fuzzy Logic in Control Systems: Fuzzy Logic Controller", *IEEE Transactions on Systems, Man and Cybernetics*, Vol 20, Part I: pp. 404-418, Part II: pp. 419-435, 1990.

[16] Li, Y.F., Lau, C.C., "Development of Fuzzy Algorithms for Servo Systems", *IEEE Control Systems Magazine*, pp. 65, 72, April, 1989.

[17] Mizumoto, M., "Improvement Methods of Fuzzy Controls", *Proceedings of 3rd IFSA*, Seattle, WA, Aug 6-11, pp. 60-62, 1989.

[18] Pedrycz, W., "Some Applicational Aspects of Fuzzy Relational Equations in Systems Analysis", *International Journal of General Systems*, Vol. 9, pp 125-132, 1983.

[19] Pedrycz, W., "Why Triangular Membership Functions?", *Fuzzy Sets and Systems*, Vol. 64, pp. 21-30. 1994.

[20] Postlethwaite, B., "Basic Theory and Algorithms for Fuzzy Sets and Logic", J. McGhee *et al.* (Editors), *Knowledge-Based Systems for Industrial Control*, Peter Peregrinus Ltd., pp. 34-46, 1990.

[21] Qiao, W.Z.,. Zhuang, W.P., Heng, T.H., "A Rule Self-Regulating Fuzzy Controller", *Fuzzy Sets and Systems*, Vol. 47, pp. 13-21, 1992.

[22] Ren, P., "Generalized Fuzzy Sets and Representation of Incomplete Knowledge", *Fuzzy Sets and Systems*, Vol. 36, pp. 91-96, 1990.

[23] Rutherford, D.A., Bloore, G.C., "Implementation of Fuzzy Algorithms for Control", *Proceedings of the IEEE*, Vol. 76, pp. 572-573, April 1976.

[24] Self, K., "Designing with Fuzzy Logic", *IEEE Spectrum*, pp. 42-44, 105, November, 1990.

[25] Tong, R.M., "The Evaluation of Fuzzy Models Derived from Experimental Data", *Fuzzy Sets and Systems*, Vol. 4, pp. 1-12, 1980.

[26] Valente de Oliveira, J., "Neuron Inspired learning Rules for Fuzzy Relational Structures", *Fuzzy Sets and Systems*, Vol. 57, pp. 41-53, 1993.

[27] Walpole, R.E., Myers, R.H., *Probability and Statistics for Engineers and Scientist*, Macmillan Publishing Co. Inc., New York, pp. 110, 1978.

[28] Xu, C.W., Lu, Y.-Z., "Fuzzy Model Identification and Self-Learning for Dynamic Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 17, pp. 683-689, 1987.

[29] Zadeh, L.A., "Outline of a New Approach to the Analysis of Complex Systems and Decision Processes', *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC-3, No. 1, pp. 28-44, 1973.

[30] Zadeh, L.A., "The Calculus of Fuzzy If/Then Rules", *AI Expert*, pp. 23-27, March 1992.

[31] Zimmermann, H.-J., *Fuzzy Set Theory-and Its Applications*, Kluwer Academic Publishers, Hingham, MA, 1985.

CHAPTER 3 *FUZZY* RELATIONAL-BASED SYSTEMS

Fuzzy relational equations play a significant role as a platform for a uniform development of techniques in fuzzy sets

[Pedrycz, 1991]

3.1 Introduction

Fuzzy logic can be applied to control problems through the use of *fuzzy relational equations*. *Relational-based fuzzy systems* use a *fuzzy relational equation* which qualitatively relates the input and output states. The relational model is a matrix composed of values which represent the degree of truth of each possible cause-and-effect relationship that can exist between the inputs and outputs.

Relational equations date back to the middle of the 1970's when they were studied by Rudeanu and were devoted exclusively to processing Boolean variables in relational structures. In the area of fuzzy research, relational equations were introduced by Sanchez [1976] as part of his work of the application of *relational equations* to medical diagnosis [Pedrycz, 1991(a)].

The advantage of the *relational equation* approach is that several techniques exists which allow the relational matrix to be identified directly from input-output data which is a more straight forward approach and is less subjective than developing a rule set based on interviews with process operators. *Relational-based* fuzzy equations exploit the numerical framework inherent in fuzzy logic. The information available in the *fuzzy relational matrix*, can be expressed in a fuzzy *state-space* design $y_k = y_{k-1} \circ u_{k-\tau-1} \circ R$, where $\circ$ is the fuzzy compositional operator. This design facilitates numerical analysis of the *fuzzy system* which is a major advantage of *relational-based systems* over *rule-based systems*.

The mathematical basis of *fuzzy relational equations* is reviewed in this section and then it is shown how *fuzzy relational systems* can be translated to *rule-based fuzzy systems* and visa versa.

3.2 Fuzzy Definitions and Notation

A brief overview of fuzzy definitions and notation is required prior to the investigation of fuzzy relational matrices. This is by no means an exhaustive coverage of the area. It is included as background information and a reference for terminology that will be used later in the thesis development.

3.2.1 Fuzzy Sets

Fuzzy sets, in which membership is gradual, are actually *generalizations* of *classical sets*, where membership function can take on only two values {0,1}, 1 being *full* membership and 0 being *no* membership. The term *fuzzy* is used for convenience.

Let $\mathcal{U}$ be a collection of objects or variables, either discrete $\{u_i\}$ or continuous $\{u\}$. $\mathcal{U}$ is called the universe of discourse and includes the range of all the elements u of $\mathcal{U}$. The universe is never fuzzy.

A ***fuzzy subset***, $\boldsymbol{F}$, of a universe of discourse $\mathcal{U}$ can be represented by a membership function:

$$\mu_F: \mathcal{U} \rightarrow [0,1]. \tag{3.1}$$

This function assigns to each element u of $\mathcal{U}$ a number $\mu_F(u)$ in the interval [0,1] which specifies the grade of membership.

When $\mathcal{U}$ is ***continuous***, the *fuzzy subset*, $\boldsymbol{F}$, can be written:

$$\boldsymbol{F} = \int_{\mathcal{U}} (\mu_F(u)/u). \tag{3.2}$$

When $\mathcal{U}$ is ***discrete***, the *fuzzy subset*, $\boldsymbol{F}$, can be written:

$$\boldsymbol{F} = \sum_{i=1}^{n} (\mu_F(u_i)/u_i). \tag{3.3}$$

In both of these definitions the integral sign and the summation sign are interpreted as *union* rather than an arithmetic sum.

More commonly, for the case when $\mathcal{U}$ is discrete, $\boldsymbol{F}$ can be written as a set of ordered pairs of the grade of membership $\mu_F(u_i)$ and the elements u_i:

$$\boldsymbol{F} = \{(\mu_F(u_i), u_i) \mid u_i \in \mathcal{U}\}. \tag{3.4}$$

If the elements of the universe of discourse are well known, the notation for the fuzzy set $\boldsymbol{F}$ can be reduced to a vector notation:

$$\boldsymbol{F} = \{\mu_F(u_i) \mid u_i \in \mathcal{U}\}. \tag{3.5}$$

The ***support*** of the *fuzzy subset*, $\boldsymbol{F}$, is the set of all points u in $\mathcal{U}$ where $\mu_F(u)>0$, (i.e. $\mu_F(u)\neq 0$).

The ***crossover point*** in $\boldsymbol{F}$ is an element u of $\mathcal{U}$ whose grade of membership in $\boldsymbol{F}$ is $\mu_F(u) = 0.5$.

A ***fuzzy singleton*** is a *fuzzy set* whose support is a single point in $\mathcal{U}$. So if F is a fuzzy singleton whose support is the point u, then:

$$F = \mu/u \tag{3.6}$$

where μ is the grade of membership of u in F. For this notation to be consistent, a *non fuzzy singleton* is denoted by $1/u$, where the single point has full membership which equates to *non fuzzy*.

The ***height*** of F is:

$$\mathrm{hgt}(F) = \max_{u \in \mathcal{U}} \mu_F(u). \tag{3.7}$$

A *fuzzy set*, F, is said to be ***normalized*** iff:

$$\exists \ u \in \mathcal{U}, \ \mu_F(u) = 1. \tag{3.8}$$

This definition implies that the hgt(F) = 1.

A set F is said to be ***included*** in G (i.e. $F \subseteq G$) iff:

$$\forall u \in \mathcal{U}, \qquad \mu_F(u) \leq \mu_G(u). \tag{3.9}$$

To determine whether an element $u \in \mathcal{U}$ typically belongs to a fuzzy set, F, it may be required that the membership value be greater than some threshold value $\alpha \in \,]0,1]$. The set of these elements is called the ***α-cut***, F_α, of F:

$$F_\alpha = \{u \in \mathcal{U}, \ \mu_F(u) \geq \alpha\}. \tag{3.10}$$

A *fuzzy set*, F, is ***convex*** iff its α-cuts are convex. Or, $\forall u_1 \in \mathcal{U}, \ \forall u_2 \in \mathcal{U}, \ \forall \lambda \in [0, 1]$:

$$\mu_F(\lambda u_1 + (1-\lambda)u_2) \geq \min(\ \mu_F(u_1), \ \mu_F(u_2)). \tag{3.11}$$

If F and G are convex, then so is $F \cap G$.

3.2.2 Fuzzy Set Operations

Let A B and C be fuzzy subsets in $\mathcal{U}$ with membership functions μ_A, μ_B and μ_C, respectively. The ***fuzzy set theoretic operations*** of *union*, *intersection*, and *complement* are defined by their membership functions as follows:

The membership function of the ***union*** $A \cup B$ is defined for all $u \in \mathcal{U}$ by:

$$\mu_{A \cup B}(u) = \mu_A(u) \cup \mu_B(u) = \max\{\mu_A(u), \mu_B(u)\}. \qquad (3.12)$$

The ***union*** corresponds to the ***disjunction*** connective ***or***.

The membership function of the ***intersection*** $A \cap B$ is defined for all $u \in \mathcal{U}$ by:

$$\mu_{A \cap B}(u) = \mu_A(u) \cap \mu_B(u) = \min\{\mu_A(u), \mu_B(u)\}. \qquad (3.13)$$

The ***intersection*** corresponds to the ***conjunction*** connective ***and***.

The membership function of the ***complement*** of $\dot{A}$ is defined for all $u \in \mathcal{U}$ by:

$$\mu_{\bar{A}}(u) = 1 - \mu_A(u). \qquad (3.14)$$

The ***complement*** corresponds to the negation ***not***.

The above definitions of *union* (*or*) and *intersection* (*and*) were given by Zadeh [1965] in his landmark paper "Fuzzy Sets". These definitions correspond to *non-interactive disjunction* and *non-interactive conjunction*, respectively. The condition of *non-interactive* requires that for either $A \cup B$ or $A \cap B$ an increase in the first argument cannot be compensated by or traded off with a decrease in the second argument, or vice-versa. More formally this can be stated:

Let $\mathcal{U}=\{u\}$, $\mathcal{V}=\{v\}$ and c be a mapping from $\mathcal{U} \times \mathcal{V}$ to the unit interval [0,1]. Then for all $u \in [0,1]$, there does not exist an $\alpha, \beta \in [0,1]$ such that $\alpha > u$, $\beta < u$ (or $\alpha < u$, $\beta > u$) and $c(\alpha, \beta) = c(u, u)$. For this to be true c must be of the form:

$$c = \min(u, v) = u \wedge v \qquad (3.15)$$

On the contrary, *interactive* allows an increase in one argument to be compensated for by a decrease on the other [Bellman *et al.*, 1977].

As given in the example, the *non-interactive conjunction* is defined uniquely by the *min* connection, while the *interactive conjunction* is strongly application specific and has no universally valid definition.

The above definition for *union*, *intersection* and *complementation* adhere to the basic identities which hold for ordinary sets, such as:

(1) ***Commutativity***: $A \cup (B \cup C) = (A \cup B) \cup C$

$A \cap (B \cap C) = (A \cap B) \cap C$

(2) ***Associativity***: $A \cup B = B \cup A$

$A \cap B = B \cap A$

(3) ***Idempotency***: $A \cup A = A$

$A \cap A = A$

(4) ***Distributivity***: $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$

(5) ***Boundary***: $A \cup U = U$

$A \cap \varnothing = \varnothing$

(6) ***Identity***: $A \cup \varnothing = A$

$A \cap \mathbf{U} = U$

(7) ***Absorption***: $A \cap (A \cup B) = A$

$A \cup (A \cap B) = A$

(8) ***De Morgan's Laws***: $\overline{(A \cup B)} = \bar{A} \cap \bar{B}$

$\overline{(A \cap B)} = \bar{A} \cup \bar{B}$

(9) ***Involution***: $\bar{\bar{A}} = A$

(10) ***Equivalence Formula***: $(\bar{A} \cup B) \cap (A \cup \bar{B}) = (\bar{A} \cap \bar{B}) \cup (A \cap B)$

(11) ***Symmetrical Difference Formula***:

$(\bar{A} \cap B) \cup (A \cap \bar{B}) = (\bar{A} \cup \bar{B}) \cap (A \cup B)$

The only law of ordinary set theory that is not longer valid is:

(12) ***The Law of excluded middle***: $A \cap \bar{A} \neq \varnothing$

$A \cup \bar{A} \neq U$

When computing composite terms the precedence rules governing the evaluation of Boolean expressions is observed.

Precedence	Operation
First	*not*
Second	*and*
Third	*or*

The definition presented for *union* and *intersection* are by no means the only definitions for these fuzzy set operations, but they are the most popular. Two other equally important definitions, one for *union* and the other for *intersection* are:

The membership function of an ***alternate union*** $A \cup B$ is defined for all $u \in \mathcal{U}$ by:

$$\mu_{A \cup B}(u) = \mu_A(u) \cup \mu_B(u) = \mu_A(u) \oplus \mu_B(u) = \mu_A(u) + \mu_B(u) - \mu_A(u) \cdot \mu_B(u) \quad (3.16)$$

The operation for the ***alternate union***, in this case, is the ***algebraic sum***. This definition of *union* is *associative, commutative* and *non-interactive.*

The membership function of an ***alternate intersection*** $A \cap B$ is defined for all $u \in \mathcal{U}$ by:

$$\mu_{A \cap B}(u) = \mu_A(u) \cap \mu_B(u) = \mu_A(u) \cdot \mu_B(u) \quad (3.17)$$

The operation for this ***alternate intersection*** is the ***algebraic product***. This definition of *intersection* is also *associative* and *commutative*, but, it is *interactive*.

The ***complement*** for these two ***alternate*** definitions of ***union*** and ***intersection*** is the same as given above. These alternate definitions along with the *complement* follow *De Morgan's Laws*, but they do not follow the *Laws of Distribution, Idempotency, Excluded Middle, Absorption*, the *Boundary* and *Identity* condition, nor the *Formulae of Equivalency* or *Symmetrical Difference*. The condition of *Involution* is unchanged under these new definitions.

Figure 3.1 gives a graphic comparison of the common and alternate definition for *union, intersection* and *complementation*. This comparison is important for this thesis as the *product* operator is substituted for the more commonly used *minimum* operator. The basis for this substitution is discussed in detail in Chapter 4.

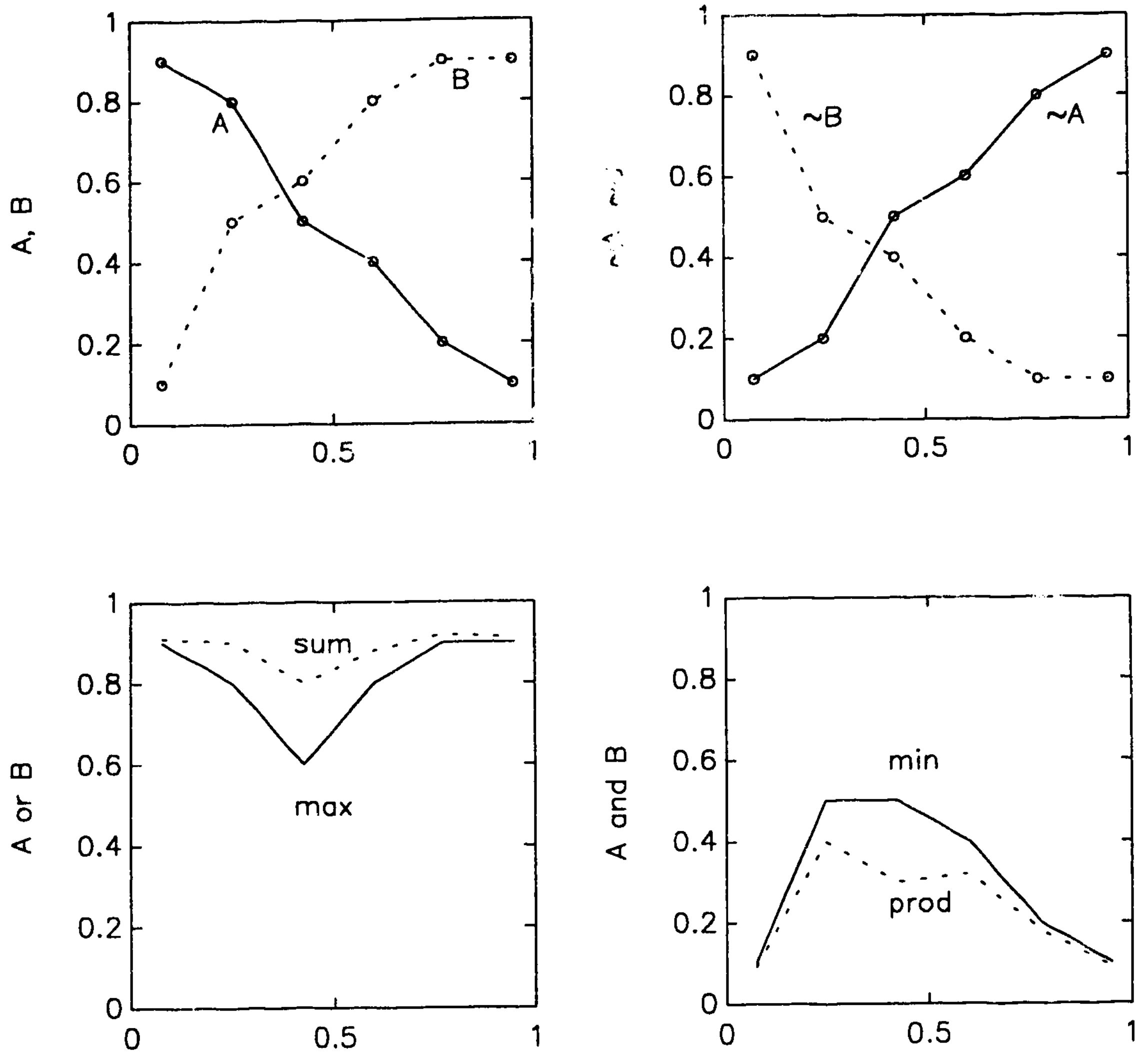


Figure 3.1: Operations on Fuzzy Sets [Pedrycz, 1989]

$(\sim A \equiv \bar{A})$

3.2.3 Other Fuzzy Sets

Occasionally in the literature, *fuzzy sets* are referred to as ***L-fuzzy sets***, where the ***L*** refers to a *lattice* structure, and the set is defined with infinite bounds. Under this structure, intersection and union are defined:

$$\forall \; u \in \mathcal{U}, \quad \mu_{F \cap G}(u) = \inf(\mu_F(u), \mu_G(u))$$
$$\forall \; u \in \mathcal{U}, \quad \mu_{F \cup G}(u) = \sup(\mu_F(u), \mu_G(u)) \tag{3.18}$$

where *inf* and *sup* denote the greatest lower bound and the least upper bound, respectively. Another lattice structure considered is ***Brouwerian lattice***, $\boldsymbol{L}$, such that:

$$\forall \ a \in \boldsymbol{L}, \ \forall \ b \in \boldsymbol{L}, \ \{x \in \boldsymbol{L}, \inf(a, x) \leq b\}. \tag{3.19}$$

In other words there is a least upper bound, x denoted $a \alpha b$. The ***dual Brouwerian lattice***, $\boldsymbol{L}$, is such that:

$$\forall \ a \in \boldsymbol{L}, \ \forall \ b \in \boldsymbol{L}, \ \{x \in \boldsymbol{L}, \sup(a, x) \geq b\}. \tag{3.20}$$

In this case there is a greatest lower bound, x denoted $a \varepsilon b$.

In cases where the bounds are finite, as is often the case with *fuzzy sets*, the *sup* is replaced with *max* or the symbol $\vee$ and the *inf* is replace with *min* or the symbol $\wedge$.

3.2.4 Fuzzy Relations

If $F_1, \ldots, F_n$ are *fuzzy sets* in $\mathcal{U}_1, \ldots, \mathcal{U}_n$, respectively, the ***Cartesian Product*** of $F_1, \ldots, F_n$ is a *fuzzy set* on the product space $\mathcal{U}_1 \times \ldots \times \mathcal{U}_n$ with membership function:

$$\mu_{F_1 \times \ldots \times F_n}(u_1, u_2, \ldots, u_n) = \min\{\mu_{F_1}(u_1), \ldots, \mu_{F_n}(u_n)\} \tag{3.21}$$

if the *intersection* is *min*, or

$$\mu_{F_1 \times \ldots \times F_n}(u_1, u_2, \ldots, u_n) = \mu_{F_1}(u_1) \cdot \mu_{F_2}(u_2) \ldots \mu_{F_n}(u_n) \tag{3.22}$$

if the *intersection* is *algebraic product*.

A ***fuzzy relation***, $\boldsymbol{R}$, from a fuzzy set on the universe of discourse $\mathcal{U}$ to a fuzzy set on the universe of discourse $\mathcal{V}$, is a fuzzy set on the cartesian product $\mathcal{U} \times \mathcal{V}$. This fuzzy set or relation $\boldsymbol{R}$ on $\mathcal{U} \times \mathcal{V}$ can be represented by the membership function:

$$\mu_{\boldsymbol{R}}: \mathcal{U} \times \mathcal{V} \rightarrow [0,1]. \tag{3.23}$$

This function assigns to the pairing of each element u of $\mathcal{U}$ with each element of v of $\mathcal{V}$ a number $\mu_{\boldsymbol{R}}(u, v)$ in the interval [0,1] which specifies the grade of membership of the ordered pair (u, v). This ***fuzzy relation*** can be expressed:

$$\boldsymbol{R} = \{(\mu_{\boldsymbol{R}}(u, v), (u, v)) | (u, v) \in \mathcal{U} \times \mathcal{V}\} \tag{3.24}$$

Often this relation may be represented as a ***relational matrix***. For example, if $u_1, u_2 \in \mathcal{U}$, $v_1, v_2 \in \mathcal{V}$ and $\boldsymbol{R}$ is the relation:

$$\boldsymbol{R} = \{(0.8, (u_1, v_1)), (0.6, (u_1, v_2)), (0.2, (u_2, v_1)), (0.9, (u_2, v_2))\} \tag{3.25}$$

then $\boldsymbol{R}$ can be expressed in ***relational matrix form*** as:

$$\boldsymbol{R} = \begin{matrix} & v_1 & v_2 \\ u_1 & \multicolumn{2}{c}{\multirow{2}{*}{}} \end{matrix} \quad \begin{matrix} \\ u_1 \\ u_2 \end{matrix} \begin{matrix} v_1 \quad v_2 \\ \begin{pmatrix} 0.8 & 0.6 \\ 0.2 & 0.9 \end{pmatrix} \end{matrix} \tag{3.26}$$

More generally, an ***n-ary fuzzy relation***, $\boldsymbol{R}$, is a fuzzy set in $\mathcal{U}_1 \times \ldots \times \mathcal{U}_n$ and is expressed as:

$$\boldsymbol{R}_{\mathcal{U}_1 \times \ldots \times \mathcal{U}_n} = \{(\mu_{\boldsymbol{R}}(u_1, u_2, \ldots, u_n), (u_1, u_2, \ldots, u_n)) \mid (u_1, u_2, \ldots, u_n) \in \mathcal{U}_1 \times \ldots \times \mathcal{U}_n\} \tag{3.27}$$

A ***fuzzy relation***, $\boldsymbol{R}$ can also be known as a ***fuzzy restriction***. Let $u = (u_1, u_s, \ldots u_n)$ be a variable on $\mathcal{U} = \mathcal{U}_1 \times \mathcal{U}_2 \times \ldots \times \mathcal{U}_n$. Then a ***fuzzy restriction***, denoted $\boldsymbol{R}(u)$, is a *fuzzy relation* that acts as a *constraint* on the values of $\mathcal{U}$ that may be assigned to the variable u.

The ***projection*** of a *fuzzy relation*, $\boldsymbol{R}$, means the *projection* of the relation on its various subspaces. For example:

$$\mu_{\boldsymbol{R}}(u) = \max_{v} [\mu_{\boldsymbol{R}}(u, v)] \tag{3.28}$$

$$\mu_{\boldsymbol{R}}(v) = \max_{u} [\mu_{\boldsymbol{R}}(u, v)] \tag{3.29}$$

Note that for this 2-dimensional system, the projection of a projection is the *height*.

$$\text{hgt}(\boldsymbol{R}) = \max_{v} \max_{u} [\mu_{\boldsymbol{R}}(u, v)] \tag{3.30}$$

The result of the projection can be interpreted as a shadow appearing on the remaining axis.

Interactivity of a *fuzzy relation*, $\boldsymbol{R}$, has been described by Zadeh [1975] as follows:

An *n-ary fuzzy relation* $\boldsymbol{R}(u_1, u_2, \ldots, u_n)$ as said to be *separable* iff

$$\boldsymbol{R}(u_1, u_2, \ldots, u_n) = \boldsymbol{R}(u_1) \times \boldsymbol{R}(u_2) \times \ldots \times \boldsymbol{R}(u_n) \tag{3.31}$$

where × denotes the cartesian product and $R(u_i)$ denote the projection of R on u_i.
The variables $u_1, u_2, \ldots, u_n$ are said to be *non-interactive* iff the *relation*, $R(u_1, u_2, \ldots, u_n)$ is separable.

Figure 3.2 illustrates *interactive* and *non interactive* for *fuzzy relations*.

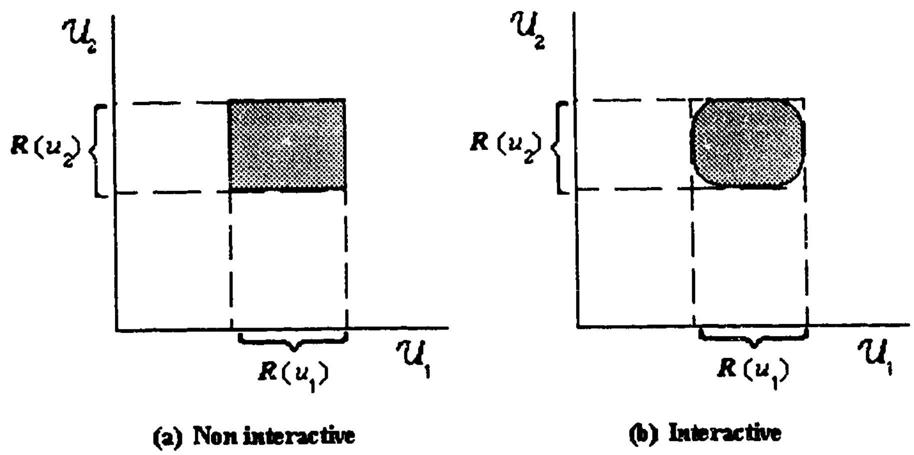


Figure 3.2: Interactive vs. Non interactive *Fuzzy* Relations [Dubois and Prade, 1980]

Various properties of ***fuzzy relation*** have been defined so that the *relational matrix* can be categorized according to its agreement with the definitions. Several of these definitions will be given below, however, this method of definition and categorization is more appropriate to *fuzzy relations* involved in circuitry.

A ***similarity relation***, the fuzzy generalization of an ***equivalence relation***, is a *fuzzy relation*, ***R***, which is ***reflexive***, ***symmetrical*** and ***max-min transitive***.

$$R \text{ is } \textbf{\textit{reflexive}} \text{ iff } \forall u \in \mathcal{U},\ \mu_R(u, u) = 1 \tag{3.32}$$

$$R \text{ is } \textbf{\textit{symmetric}} \text{ iff } \forall u \in \mathcal{U}, \forall v \in \mathcal{V},\ \mu_R(u, v) = \mu_R(v, u). \tag{3.33}$$

The ***transitivity*** of a relational matrix can be described as the strength of the link between the elements. That is the strength of the link between any two elements must be greater than or equal to the strength of any indirect linkage, which might involve other elements.

Let $\boldsymbol{R}$ be a *fuzzy relation* of $\mathcal{U}\times\mathcal{V}\times\mathcal{W}$. $\boldsymbol{R}$ is considered ***max-min transitive*** iff $\forall(u, v, w) \in \mathcal{U}\times\mathcal{V}\times\mathcal{W}$:

$$\mu_R(u, w) \geq \min(\mu_R(u, v), \mu_R(v, w)) \tag{3.34}$$

Zadeh [1971] and Bezdek *et al.* [1978] have pointed out that *max-min transitivity* is too strong a property to impose on a *similarity* relation. Bezdek *et al.* [1978] compared the strength of the *max-min* transitivity with several other binary operators.

$$\boldsymbol{R}_{\hat{+}} \subseteq \boldsymbol{R}_{\max} \subseteq \boldsymbol{R}_{\nabla} \subseteq \boldsymbol{R}_{\min} \subseteq \boldsymbol{R}_{\cdot} \subseteq \boldsymbol{R}_{\wedge} \tag{3.35}$$

where	$\hat{+}$	$a + b - ab$	(probabilistic sum)
	max	$\max(a, b)$	(union)
	∇	$1/2(a + b)$	(arithmetic sum)
	min	$\min(a, b)$	(intersection)
	$\cdot$	ab	(algebraic product)
	$\wedge$	$\max(0, a+b-1)$	(bold intersection)

They found that the *max*-$\wedge$ and the *max-product* are the weakest of the transitivities and are therefore described by Dubois *et al.* [1980] to be intuitively more appealing for fuzzy *similarity* relations than the *max-min* transitivity.

A ***transitive closure*** of a binary fuzzy relation $\boldsymbol{R}$, denoted $\mathrm{cl}(\boldsymbol{R})$, is a fuzzy relation constructed from the union of powers of $\boldsymbol{R}$,

$$\mathrm{cl}(\boldsymbol{R}) = \boldsymbol{R} \cup \boldsymbol{R}^2 \cup \ldots \cup \ldots \tag{3.36}$$

where the powers of $\boldsymbol{R}$ are defined recursively,

$$\begin{aligned} \boldsymbol{R}^2 &= \boldsymbol{R} \cap \boldsymbol{R} \\ &\vdots \\ \boldsymbol{R}^{k+1} &= \boldsymbol{R}^k \cap \boldsymbol{R} \end{aligned} \tag{3.37}$$

The Extension Principle, introduced by Zadeh [1975] is basic to the idea of fuzzy set theory. It provides a method of extending non fuzzy mathematical concepts to the fuzzy domain.

Let $\mathcal{U}$ be a cartesian product of universes, $\mathcal{U} = \mathcal{U}_1 \times \mathcal{U}_2 \times \ldots \times \mathcal{U}_n$, and $A_1, A_2, \ldots A_r$ be n fuzzy sets in $\mathcal{U}_1, \mathcal{U}_2, \ldots \mathcal{U}_n$, respectively. Let f be a mapping from $\mathcal{U}_1 \times \mathcal{U}_2 \times \ldots \times \mathcal{U}_n$ to a universe $\mathcal{V}$ such that $v = f(u_1, u_2, \ldots u_n)$. The extension principle allows us to induce from the n fuzzy sets of A_i a fuzzy set B on $\mathcal{V}$ through f such that:

$$\mu_B(v) = \max_{\substack{u_1, \ldots, u_n \\ v = f(u_1, \ldots, u_n)}} [\min(\mu_{A_1}(u_1), \ldots, \mu_{A_n}(u_n))] \tag{3.38}$$

$$\mu_B(v) = 0 \qquad \text{if } f^1(v) = \varnothing \tag{3.39}$$

where $f^1(v)$ is the inverse image of v.

A binary operation $*$ in $\mathcal{U} = \mathcal{V}$ is said to be ***increasing*** iff:

$$\begin{aligned} &\text{for } u_1 > v_1 \text{ and } u_2 > v_2, \\ &\quad \text{then } u_1 * u_2 > v_1 * v_2 \end{aligned} \tag{3.40}$$

In the same way, $*$ is said to be ***decreasing*** iff

$$\begin{aligned} &\text{for } u_1 < v_1 \text{ and } u_2 < v_2, \\ &\quad \text{then } u_1 * v_2 < u_1 * v_2 \end{aligned} \tag{3.41}$$

Using the extension principle, an arbitrary operator, $*$, can be extended into the *fuzzy* domain, so that two fuzzy sets, F and G, can be combined. The combination is defined in the following manner:

$$\mu_{F \circledast G}(w) = \max_{w = u * v} (\mu_F(u) * \mu_G(v)) \tag{3.42}$$

Now consider a graphical interpretation of a ***fuzzy composition***. Let A be a *fuzzy set* in $\mathcal{U}$, B be a *fuzzy set* in $\mathcal{V}$ and R a *fuzzy relation* in $\mathcal{U} \times \mathcal{V}$. Then

$$\mu_B(v) = \mu_{A \circ R}(v) = \max_x \min(\mu_A(u), \mu_R(u, v)) \tag{3.43}$$

can be interpreted as $B = A \circ R$, where B is the fuzzy set induced from A through R. This is shown graphically in Figure 3.3.

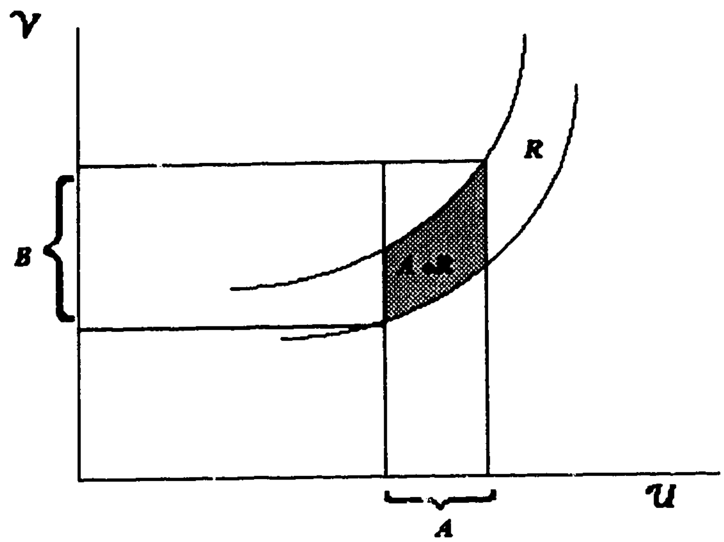


Figure 3.3: Graphical Interpretation of Fuzzy Relational Equation [Dubois and Prade, 1980]

The composition of two *fuzzy sets* can be performed by any operation that satisfies the properties of a *triangular-norm* (*t-norm*) or a *triangular-conorm* (*s-norm*). The properties of a *t-norm* are:

For x, $y \in [0,1]$

(1) **Commutativity:**		$x \mathbf{t} y = y \mathbf{t} x$
(2) **Associativity:**		$(x \mathbf{t} y) \mathbf{t} z = x \mathbf{t} (y \mathbf{t} z)$
(3) **Boundary Conditions:**		$0 \mathbf{t} x = x \mathbf{t} 0 = 0$
		$1 \mathbf{t} x = x \mathbf{t} 1 = x$
(4) **Monotonicity:**		If $x \geq v$ and $y \geq w$ then
		$x \mathbf{t} y \geq v \mathbf{t} w$
(5) **Continuity:**		For any fixed $x \in [0,1]$,
		$x \mathbf{t} y$ is continuous on $[0,1]$ for all $y \in [0,1]$.

The properties of an *s-norm* are:

For $x, y \in [0,1]$

(1)	**Commutativity:**	$x \text{ s } y = y \text{ s } x$
(2)	**Associativity:**	$(x \text{ s } y) \text{ s } z = x \text{ s } (y \text{ s } z)$
(3)	**Boundary Conditions:**	$0 \text{ s } x = x \text{ s } 0 = x$ $1 \text{ s } x = x \text{ s } 1 = 1$
(4)	**Monotonicity:**	If $x \geq v$ and $y \geq w$ then $x \text{ s } y \geq v \text{ s } w$
(5)	**Continuity:**	For any fixed $x \in \mathbf{I}$, $x \text{ s } y$ is continuous on $\mathbf{I}$ for all $y \in \mathbf{I}$.

Triangular norms and *co-norms* will be discussed in more detail in Chapter 4.

If R and S are fuzzy relations in $\mathcal{U} \times \mathcal{V}$ and $\mathcal{V} \times \mathcal{W}$, respectively, then the *max-t* composition of R and S is a fuzzy relation given by:

$$R \textcircled{t} S = \left\{ \left[\max_{v} (\mu_R(u, v) \ t \ \mu_S(v, w)), (u, w) \right] \mid u \in \mathcal{U}, v \in \mathcal{V}, w \in \mathcal{W} \right\} \qquad (3.44)$$

In this example the t could be any *triangular norm* connector, such as, *min*, *algebraic product*, etc.

3.2.5 Fuzzy Logic

In fuzzy logic there are two important fuzzy inference rules, as briefly outlined in Chapter 2. These are ***generalized modus ponens*** (GMP) and the ***generalized modus tollens*** (GMT).

Let x and y be fuzzy sets and A, A', B, B' be fuzzy linguistic variables. The ***generalized modus ponens*** for a rule-based fuzzy systems can be stated:

Premise 1: x is A'

Premise 2: if x is A then y is B

Consequence: y is B'

The GMP reduces to the *modus ponens* of traditional logic when $A' = A$ and $B' = B$. This form of implication is a *forward data-driven inference* and can be formulated as the ***compositional rule of inference*** for fuzzy relations as follows:

If R is a fuzzy relation from $\mathcal{U}$ to $\mathcal{V}$, and x is a fuzzy subset of $\mathcal{U}$, then the fuzzy subset y of $\mathcal{V}$ which is induced by x is given by the composition of R and x:

$$y = x \circ R \tag{3.45}$$

When $\circ \equiv$ *max-min* the *compositional rule of inference* is the one suggested by Zadeh [1973].

The ***generalized modus tollens*** for a rule-based fuzzy systems can be stated:

Premise 1:	y is B'
Premise 2:	if x is A then y is B
--	
Consequence:	x is A'

The GMT reduces to the *modus tollens* of traditional logic when $A' =$ ***not*** A and $B' =$ ***not*** B. This form of implication is a *backward goal-driven inference* and can be formulated as the ***compositional rule of inference*** for fuzzy relations as follows:

If R is a fuzzy relation from $\mathcal{U}$ to $\mathcal{V}$, and y is a fuzzy subset of $\mathcal{V}$, then the greatest fuzzy subset x of $\mathcal{U}$ which causes y is given by the composition of R and y:

$$x = R \circleda y \tag{3.46}$$

This definition of an inverse to the fuzzy compositional equation was first suggested by Sanchez [1976].

The concepts of *composition* and *resolution* can be extended to fuzzy relational calculations in both a system causal and system identification context. these concepts will be discussed further in Chapter 5.

3.3 Discrete State Space Fuzzy Relational Equations

In order to permit system analysis, fuzzy relational equations can be described by a *state space* formulation, similar to conventional discrete state space equations. The *fuzzy* state space model is described as follows.

Let $u=\{u_i \mid i = \{1,2, ..., m\}\} \in \mathcal{U}$, $x=\{x_j \mid j = \{1,2, ..., n\}\} \in \mathcal{X}$, and $y=\{y_l \mid l = \{1,2, ..., n\}\} \in \mathcal{Y}$ be the *fuzzy* spaces of input, state and output, respectively, all defined on the finite fuzzy universes of discourses indicated. Then for a series of N state, output and control data points, the fuzzy state space relationship is written:

$$x_k = u_{k-\tau-1} \circ u_{k-\tau-2} \circ \ldots \circ u_{k-\tau-p} \circ x_{k-1} \circ x_{k-2} \circ \ldots \circ x_{k-p} \circ R \tag{3.47}$$

$$y_k = x_k \circ S \tag{3.48}$$

where $x_k, x_{k-1}, \ldots, x_{k-p}$ are the fuzzy states at the time instances indicated,

$u_{k-\tau-1}, u_{k-\tau-2}, u_{k-\tau-p}$ are the fuzzy control instances,

y_k is the fuzzy output at time k.

τ is the system delay

p is the order of the system

and $\circ \in O$, where O stands for a family of composition operators (i.e. *max-min*, *max-product*, etc.)

The first equation, equation (3.47), is a state equation which relates the state at the k-th time instant with; the states occurring at the previous instants, $k-1, \ldots, k-p$; and the fuzzy control at the previous time instants plus a delay, $k-\tau-1, \ldots, k-\tau-p$. Equation (3.48) transforms the state x_k into the fuzzy output, y_k. The fuzzy relations R and S model the system dynamics and are defined on the cartesian product spaces:

$$R \in \underbrace{U \times U \times \ldots \times U}_{p \text{ times}} \times \underbrace{X \times X \times X \ldots \times X}_{(p+1) \text{ times}}; \text{and} \qquad S \in X \times Y \quad . \tag{3.49}$$

Assuming the fuzzy states and the fuzzy outputs are the same, (i.e. $x_k = y_k$ for all k), then S is an identity matrix, with $S(x_j, y_l) = 1$ if $x_j = y_l$ and 0 otherwise. For this case the equation (3.48) can be ignored and only the state equation is considered, as follows:

$$y_k = u_{k-\tau-1} \circ u_{k-\tau-2} \circ \ldots \circ u_{k-\tau-p} \circ y_{k-1} \circ y_{k-2} \circ \ldots \circ y_{k-p} \circ R \tag{3.50}$$

Note that the fuzzy relation R expresses the relationship between the reference fuzzy sets not the relationship between the actual input-output data.

The following proposition shows that a p-th order model can be transformed into a first order model ($p = 1$) without lost of generality [Pedrycz, 1989, 1993]

Proposition 1: The fuzzy p-th order model given in (3.50) can be reduced to a first order model (p = 1), as follows:

$$y_{k+1} = u_{k-\tau-1} \circ y_{k-1} \circ K \tag{3.51}$$

where $$K = u_{k-\tau-2} \circ \ldots \circ u_{k-\tau-p} \circ y_{k-2} \circ \ldots \circ y_{k-p} \circ R \tag{3.52}$$

Inserting equation (3.52) into (3.51) results in equation (3.50), the original p-th order equation model.

$$y_k = u_{k-\tau-1} \circ u_{k-\tau-2} \circ \ldots \circ u_{k-\tau-p} \circ y_{k-1} \circ y_{k-2} \circ \ldots \circ y_{k-p} \circ R \tag{3.50}$$

This analysis by Pedrycz [1989, 1993] is particularly important with regard to the dimensionality problem often suffered by fuzzy relational models. This dimensionality deficiency can be alleviated by exploiting this analysis.

Based on the generality of the first order fuzzy model, further discussion of the relational equations will be restricted to a first order time delay model.

3.4 Fuzzy Discretization

For situations when the input-output data is discrete but the process is too complicated to be modeled deterministically, the discrete data can be transform into a fuzzy format through fuzzy discretization. The idea of fuzzy discretization is explained as following. Let $X_1, X_2, \ldots, X_\rho$ be referential fuzzy sets defined in the space or universe of discourse, $\mathcal{X}$, such that:

$$X_i : \mathcal{X} \rightarrow [0, 1] \qquad \text{for } i = \{1, 2, \ldots, \rho\} \tag{3.53}$$

and the condition of completeness is satisfied:

$$\forall\, x \in \mathcal{X}:\ \exists\, i \in \{1, 2, \ldots, \rho\} \ni X_i(x) = x_i > 0 \tag{3.54}$$

That is, the referential fuzzy sets *cover* the complete range of the input space $\mathcal{X}$. As well, every fuzzy set defined in $\mathcal{X}$ can be expressed in terms of the family of referential fuzzy sets and can be represented by a ρ-tuple of numbers in the interval [0, 1] as will be shown.

The possibility of $\boldsymbol{Z}$ belonging to $\boldsymbol{X}_i$ $\{i = 1, 2, \ldots, \rho\}$ is defined [Pedrycz, 1984] as:

$$p_i = \max_{x \in \mathcal{X}} (\boldsymbol{X}_i(x) \ \mathbf{t} \ \boldsymbol{Z}(x)) \qquad \{i = 1, 2, \ldots, \rho\} \tag{3.55}$$

where $\quad p_i \in [0, 1]$ $\{i = 1, 2, \ldots, \rho\}$

and $\quad \mathbf{t}$ is any triangular norm.

If $\boldsymbol{Z}$ is a fuzzy singleton where the membership function is equal to 1 in exactly one point of $\mathcal{X}$, say x_o, then:

$$\boldsymbol{Z}(x) = \begin{cases} 1 & x = x_o \\ 0 & x \neq x_o \end{cases} \tag{3.56}$$

and

$$p_i = \boldsymbol{X}_i(x_o) \qquad \{i = 1, 2, \ldots, \rho\} \tag{3.57}$$

The following example from Chen *et al.* [1994] clearly illustrates this concept.

Example 1

Assume $\mathcal{X} = \{100, 200, 300, 400, 500\}$ is the discrete temperature universe of discourse. Let $\{\boldsymbol{X}_1, \boldsymbol{X}_2, \boldsymbol{X}_3\}$ be the set of reference fuzzy sets defined in $\mathcal{X}$, such that $\boldsymbol{X}_1 = \{1, 0.5, 0, 0, 0\}$, $\boldsymbol{X}_2 = \{0, 0.5, 1, 0.5, 0\}$, and $\boldsymbol{X}_3 = \{0, 0, 0, 0.5, 1\}$ represent the linguistic values of *low*, *medium* and *high temperature*, respectively. Therefore any temperature within the universe of discourse, $\mathcal{X}$, can be expressed by reference fuzzy sets $\{\boldsymbol{X}_1, \boldsymbol{X}_2, \boldsymbol{X}_3\}$. Thus a *slightly low temperature* denoted by the fuzzy set {0.8, 0.9, 0.7, 0, 0} can be expressed as {0.8, 0.7 0} through application of equation (3.55). A non-fuzzy temperature, $x = 200$, can be expressed through equation (3.57) as {0.5, 0.5, 0}, which is the same result obtained through equation (3.55) when the accurate temperature ($x = 200$) is expressed as the fuzzy set {0, 1, 0, 0, 0}.

The approach of approximating fuzzy sets by a family of referential fuzzy sets has two main advantages [Pedrycz, 1984]:

(1) a reduction of the memory load in a computer implementation of the algorithms dealing with the fuzzy data

(2) a unified treatment of fuzzy and nonfuzzy forms of information

The choice of the shape and number of the referential fuzzy sets is still relatively subjective, however, there are several methodologies available in the literature to aid in their computation. Fuzzy clustering methods by Bezdek [1981], and in particular the FUZZY ISODATA algorithm, is often sited in the literature [Valente de Oliveira, 1994, Pedrycz, 1984]. One of the drawbacks of using clustering algorithms for the construction of the referential fuzzy sets is that predictions are always below the maximum values of the state and above the minimum values, due to their interpolating character [Valente de Oliveira, 1994].

Recently, Pedrycz [1994] pointed out that triangular fuzzy membership functions are frequently used for the development of fuzzy controllers and classifications schemes mainly due to their simplicity and limited information about the actual linguistic terms. However, in his paper he validates the use of these membership functions for fuzzification in that they lead to entropy equalization, based on assumptions for the underlying probability density function. Similarly for defuzzification, triangular membership functions, with a 1/2 overlap level produce a zero value reconstruction error. Pedrycz [1994] qualifies this work by pointing out that other membership function shapes may produce the same results as these, but at the expense of the simplicity.

Shaw *et al.* [1992] provide the following detailed algorithm for the calculations of *fuzzification* and *defuzzification*.

Definition 1: A given space Z can be partitioned into ρ referential fuzzy sets consisting of isosceles triangles, which satisfy the requirements of being normal convex and completely covering the space Z [Zadeh, 1965]. The referential fuzzy sets also overlap, as shown in Figure 3.4. So every real value of z will generate a fuzzy variable $\tilde{z}$ which is a set of ρ-terms. So

$$\tilde{z} = \{z_1, z_2, \ldots, z_\rho\} \tag{3.58}$$

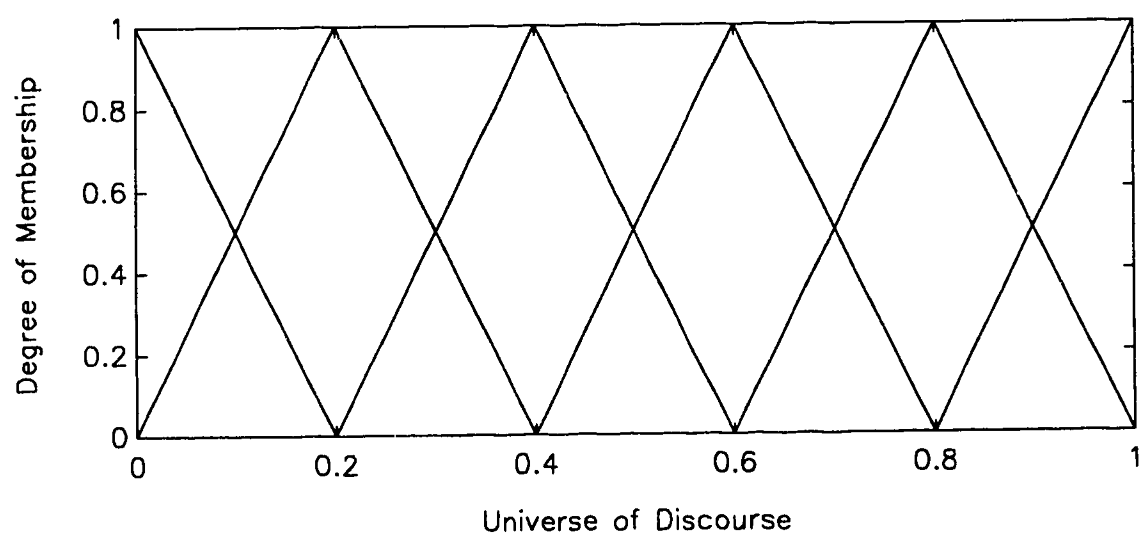


Figure 3.4: Fuzzy Referential Sets

Definition 2: Partitioning the real space Z into ρ referential fuzzy sets, *fuzzification* is defined as the 1-to-ρ mapping from the single real value to the ρ fuzzy values. The *fuzzifier operator* ϕ_ρ defined over ρ referential fuzzy sets is:

$$\tilde{z} = \phi_\rho (z) \qquad (3.59)$$

Definition 3: Partitioning the real space Z into ρ referential fuzzy sets, *defuzzification* is defined as the ρ-to-1 mapping where the fuzzy ρ-term is converted into a single real value. The *defuzzifier operator* ϕ_ρ^{-1} defined over ρ referential fuzzy sets is:

$$z = \phi_\rho^{-1} (\tilde{z}) \qquad (3.60)$$

Combining equations (3.59) and (3.60):

$$z_i = \phi_\rho [\phi_\rho^{-1} (z_i)] \qquad (3.61)$$

it would appear that theoretically the same referential sets can be used for both fuzzification and defuzzification since they are inverses of the other. This is not necessarily the case as illustrated by the following discussion on the problem of *leakage*.

One of the benefits of fuzzification and defuzzification is that the systems itself tends to filter out the excitation of certain states [Braae *et al.*, 1979]. However, Shaw *et al.* [1992] performed experiments using uniformly distributed white noise as an input and showed that excessive noise tends to distort the fuzzy relation causing *leakage* effects.

Leakage results when, using the earlier definitions of *fuzzification* and *defuzzification*, the following inequality occurs:

Given $\quad z_2 = z_1 \circ \boldsymbol{R} \qquad (3.62)$

calculate $\quad z_2' = \phi_\rho[\phi_\rho^{-1} (z_1)] \circ \boldsymbol{R} \qquad (3.63)$

then $\quad z_2' \neq z_2 \qquad (3.64)$

In general, the *defuzzifier operator*, ϕ_ρ^{-1} yields the inverse of the *fuzzifier operator*, ϕ_ρ only for certain values of $\boldsymbol{R}$. More normally, as a result of the process by which $\boldsymbol{R}$ has been identified, the components of $\boldsymbol{R}$ have values such that defuzzification is not the inverse of fuzzification. More specifically, operations on $\boldsymbol{R}$ introduce additional components to the fuzzy ρ-term set which are not present originally and so the defuzzified value of z_2' contains errors. This phenomenon, called *leakage*, prevents the use of the fuzzifier for defuzzification and results in the use of an approximation or defuzzification techniques. Therefore defuzzification methodologies, such as *mean of maximum* or *center of area* discussed earlier, yield the *most likely* rather that the exact real value corresponding to the fuzzy ρ-term set variable.

Since the defuzzifier and fuzzifier no longer function as inverse operators, distortion or *leakage* results when calculated output states are fed back recursively to the fuzzy model. To alleviate this problem, the estimated or calculated fuzzy states must first be defuzzified and then fuzzified again before being applied recursively to the identification algorithm. This process acts to *restore* the fuzzy state to the correct number of non-zero and zero values which conform to the fuzzy states generated by the real variables. It does not, however, and can not eliminate the error due to the defuzzifier and fuzzifier not being exact inverses.

3.5 Translation between Relational-Based and Rule-Based Fuzzy Models

Rule-based fuzzy models are generally more recognizable due to there similarity to rule-based expert systems. The structure is also more comprehensible for those systems consisting of two inputs affecting one output with the construction of the relatively simplistic *look-up* table. Most of the introductory articles dealing with fuzzy systems illustrate fuzzy rule-based systems. Additional, almost every successful commercial or industrial design has been a rule-based fuzzy system [Kosko, 1992].

This section will show that relational-based fuzzy models and rule-based fuzzy models can actually represent the same information. While rule-based fuzzy system have a structure almost universally understood without a strong mathematical background, relational based fuzzy models possess a solid mathematical base that permits mathematical analysis.

3.5.1 Translation from Relational-Based to Rule-Based

Pedrycz [1984] describes the translation from a relational-based fuzzy system to a rule-based fuzzy system as follows. Let $U_1, U_2, \ldots, U_{\rho_u}$ be ρ_u linguistic referential fuzzy sets defined on the input space $\mathcal{U}$, such that for u ∈ $\mathcal{U}$:

$$u : U_i \rightarrow [0,1] \qquad (3.65)$$

and the conditions of completeness are satisfied:

$$\forall\, u \in \mathcal{U}:\ \exists\, i \in \{1, 2, \ldots, \rho_u\} \ni U_i(u) = u_i > 0 \qquad (3.66)$$

That is the referential fuzzy sets *cover* the complete range of the input space $\mathcal{U}$.

Also correspondingly, let $Y_1, Y_2, \ldots, Y_{\rho_y}$ be ρ_y linguistic referential fuzzy sets defined on the input space $\mathcal{Y}$,

$$y : Y_i \rightarrow [0,1] \qquad (3.67)$$

and the same conditions of completeness are satisfied:

$$\forall y \in \mathcal{Y}: \exists j \in \{1, 2, \ldots, \rho_y\} \ni Y_j(y) = y_j > 0 \qquad (3.68)$$

Define the discrete first order state space fuzzy model, with $y = x$, as follows:

$$y_k = u_{k-\tau-1} \circ y_{k-1} \circ R \qquad \text{for } k = 1, 2, \ldots, N \qquad (3.69)$$

where k is a specific time instant

u is the fuzzy input, defined $u = [u_1, u_2, \ldots, u_{\rho_u}]$

y is the fuzzy output, defined $y = [y_1, y_2, \ldots, y_{\rho_y}]$

and R is the fuzzy relational matrix for the first order model.

Since the relational matrix R has $\rho_u\rho_y^2$ elements, this relational fuzzy model can be translated into a set of $\rho_u\rho_y^2$ fuzzy implication statements of the type:

If input is u_i *and output is* y_j *(at the (k-1)-th time instant)*
then output is y_l *(at the (k)-th time instant) with possibility measure* λ_{ijl}.

The possibility measure λ_{ijl} is the (ijl)-th entry of the R matrix. From this format a simplified linguistic table can be constructed which is the same as the look-up table common in many fuzzy controllers. The look-up table, illustrated in Table 3.1, has linguistic entries of system output at the k-th time instant calculated from the formula:

$$\lambda_{ijl}' = y_p \qquad \text{where } p = l; \text{ the index from the calculation } \max_{1 \le l \le \rho_y} \lambda_{ijl} \qquad (3.70)$$

Thus the $\rho_u\rho_y^2$ fuzzy implication statements above are reduced by a factor of ρ_y to a *look-up* table format.

And, if a *rule-based system* is sought, information regarding the system can be learned via *relational matrix* techniques and then translated to a *rule-based* format

	Output at k-th Time Instant
Input at k-th Time Instant	$y_1 \ y_2 \ \cdots \ y_j \ \cdots \ y_{\rho_y}$
u_1 u_2 $\cdots$ u_i $\cdots$ u_{ρ_u}	λ_{ijl}'

Table 3.1: Linguistic Table of Translated Relational Matrix [Pedrycz, 1984]

3.5.2 Translation from Rule-Based to Relational-Based

Now consider a fuzzy *rule-based systems* generated from knowledgeable process personnel. A rule-based system in the form of a look-up table, can be translated to the relational matrix for stability analysis or to check the consistency of the rule-based model against a relational based model learned from the I/O data. Although a *rule-based fuzzy logic system* may appear to resemble a typical *rule-based expert system*, which has a symbolic framework, the *rule-based fuzzy logic system* is actually numerically based. The numerical framework of fuzzy system is consistent between the rule-based and relational-based systems. And just as a relational-based system can be translated to a rule-based system, the fuzzy logic rule-based system can also be represented by a relational matrix.

The translation from rule-based to relational-based, or a relational matrix will be demonstrated by the following example. Given a *fuzzy* control scenario:

$$\textbf{\textit{If error}} = e \ \textbf{\textit{and change of error}} = \Delta e$$
$$\textbf{\textit{then input}} = u$$

where the values or e, Δe and u are represented linguistically by *low* (L), *medium* (M) and *high* (H). This scenario can be easily represented the a *look-up table* format shown in Figure 3.5.

u

Δe				
	H	**M**	**H**	**H**
Δe	**M**	**L**	**M**	**H**
	L	**L**	**L**	**M**
		L	**M**	**H**
			e	

Figure 3.5: Example Look-Up Table

The *look-up table* can be translated to a relational matrix by equating the value of *low* to the *fuzzy vector* [1 0 0], the value of *medium* to [0 1 0] and *high* to [0 0 1]. With these vectors, the *look-up table* is expanded along an *input* axis, or, from the current 2-dimensional table to a 3-dimensional relational matrix with an axis dimension for each of *error*, *change of error* and *input*. This translation results in the $\Delta e \times e \times u$ relational matrix shown in Figure 3.6

The ability of *fuzzy rule-based systems* to be represented by relational matrices is the reason *fuzzy systems* can be classified under a numerical framework, as was outlined in the introduction. As well, it provides these system with a basis for quantitative analysis superior to traditional rule-based expert systems.

3.6 Summary

The aim of the literature review of relational-based fuzzy theory is to provide the background terminology and theory required to differentiate *relational-based fuzzy systems* from *rule-based fuzzy systems* and to emphasize the fact that rule-based fuzzy information can be translated into a relational matrix format. This co-representation is the reason that *fuzzy rule-based systems* are considered to have a numerical structure, while conventional expert rule-based systems are considered symbolic.

$$
u(\mathrm{L}) \qquad \Delta e \begin{array}{c} \mathrm{H} \\ \mathrm{M} \\ \mathrm{L} \end{array} \begin{bmatrix} 0 & 0 & 0 \\ 1.0 & 0 & 0 \\ 1.0 & 1.0 & 0 \end{bmatrix}
\quad \begin{array}{ccc} \mathrm{L} & \mathrm{M} & \mathrm{H} \\ & e & \end{array}
$$

$$
u(\mathrm{M}) \qquad \Delta e \begin{array}{c} \mathrm{H} \\ \mathrm{M} \\ \mathrm{L} \end{array} \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & 1.0 & 0 \\ 0 & 0 & 1.0 \end{bmatrix}
\quad \begin{array}{ccc} \mathrm{L} & \mathrm{M} & \mathrm{H} \\ & e & \end{array}
$$

$$
u(\mathrm{H}) \qquad \Delta e \begin{array}{c} \mathrm{H} \\ \mathrm{M} \\ \mathrm{L} \end{array} \begin{bmatrix} 0 & 1.0 & 1.0 \\ 0 & 0 & 1.0 \\ 0 & 0 & 0 \end{bmatrix}
\quad \begin{array}{ccc} \mathrm{L} & \mathrm{M} & \mathrm{H} \\ & e & \end{array}
$$

Figure 3.6: Translation from Look-up Table to Relational Matrix

REFERENCES

[1] Bellman, R.E., Zadeh, L.A., "Local and Fuzzy Logics", In: *Modern Uses of Multiple Valued Logic*, D. Reidel Publishing Co., Dordrecht-Holland, pp. 103-165, 1977.

[2] Bezdek, J.C., Harris, J.D., "Fuzzy Partitions and Relations: An Axiomatic Basis for Clustering", *International Journal of Fuzzy Sets and Systems*, Vol. 1, pp. 111-127, 1978.

[3] Bezdek, J.C., *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.

[4] Braae, M., Rutherford, D.A., "Selection of Parameters for a Fuzzy Logic Controller", *Fuzzy Sets and Systems*, Vol. 2, pp. 185-199, 1979.

[5] Chen, J.Q., Lu, J.H., Chen, L.J., "An On-Line Identification Algorithm for Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 64, pp. 63-72, 1994.

[6] Dubois, D., Prade, H., *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, Toronto, 1980.

[7] Gupta, M.M., Qi, J., "Theory of T-Norms and Fuzzy Inference Methods", *Fuzzy Sets and Systems"*, Vol. 40, pp. 431-450, 1991.

[8] Kaufmann A., *Introduction to the Theory of Fuzzy Subsets - Vol. 1*, Academic Press Inc., New York, 1975.

[9] Kaufmann, A, Gupta, M.M., *Introduction to Fuzzy Arithmetic*, Van Nostrand Reinhold Company, Inc., New York, 1985.

[10] Kosko, B., *Neural Networks and Fuzzy Systems*, Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1992.

[11] Lee, C.C., "Fuzzy Logic in Control Systems: Fuzzy Logic Controller", *IEEE Transactions on Systems, Man and Cybernetics*, Vol 20, Part I: pp. 404-418, Part II: pp. 419-435, 1990.

[12] Pedrycz, W., "Some Applicational Aspects of Fuzzy Relational Equations in Systems Analysis", *International Journal of General Systems*, Vol. 9, pp. 125-132, 1983.

[13] Pedrycz, W., "An Identification Algorithm in Fuzzy Relational Systems", *Fuzzy Sets and Systems*, Vol. 13, pp. 153-167, 1984.

[14] Pedrycz, W., *Fuzzy Control and Fuzzy Systems*, John Wiley & Sons, Inc., New York, N.Y., 1989.

[15] Pedrycz, W., "Processing in Relational Structures: Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 40, pp. 77-106, 1991(a).

[16] Pedrycz, W., "Fuzzy Modeling: Fundamentals, Construction and Evaluation", *Fuzzy Sets and Systems*, Vol. 41, pp. 1-15, 1991(b).

[17] Pedrycz, W., "Why Triangular Membership Functions?", *Fuzzy Sets and Systems*, Vol. 64, pp. 21-30, 1994.

[18] Sanchez, E., "Resolution of Composite Fuzzy Relation Equations", *Information and Control*, Vol. 30, pp. 38-48, 1976.

[19] Shaw, I.S., Krüger, J.J., "New Fuzzy Learning Model with Recursive Estimation f Dynamic Systems", *Fuzzy Sets and Systems*, Vol. 48, pp. 217-229, 1992.

[20] Zadeh, L.A., "Fuzzy Sets", *Information and Control*, Vol. 8, pp. 338-353, 1965.

[21] Zadeh, L.A., "Outline of a New Approach to the Analysis of Complex Systems and Decision Processes", *IEEE Transactions on Systems, Man and Cybernetics*, Vol., 3, pp. 28-44, 1973.

[22] Zadeh, L.A., "The Concept of a Linguistic Variable", *Information Sciences*, I: Vol. 8, pp. 199-249, II: Vol. 8, pp. 301-357, III: Vol. 9, pp. 43-80, 1975.

[23] Zimmermann, H.-J., *Fuzzy Sets Theory - and Its Applications*, Kluwer-Nijhoff Publishing, Boston, 1985.

CHAPTER 4 COMPOSITION OF FUZZY RELATIONAL EQUATIONS

For those systems that possess an inherently fuzzy structure, a key objective is to choose an ***implication operator*** *or inferencing method that ensures the highest rate of coherence between the knowledge acquisition and reasoning. However, for those systems that are fuzzified, due to lack of an adequate deterministic model, the* ***implication operator*** *is chosen for good performance at the nonfuzzy or numerical level.*

[Pedrycz, 1991]

4.1 Introduction

With the introduction of fuzzy set theory there also came the question of logical connectors. Fuzzy set applications, such as fuzzy decision-making, require the ability to connect the individual fuzzy concepts. For example decision making for the purchase of a car may require the vehicle to be economical *and* roomy *and* have 5-speeds *or* 4-doors. With such applications the question is how to combine the fuzzy sets through the *and* and *or* operators [Yager, 1982].

There is no unique answer to this question since the answer depends on what conditions the *and* and *or* are expected to satisfy (See discussion in Section 3.2.2). The statement of economical *and* roomy implies that both criteria must be satisfied, so *and* represents the degree of simultaneous satisfaction of both criteria. The statement of 5-speed *or* 4-door implies that satisfying either condition is acceptable, so *or* is the degree of collective satisfaction of the conditions [Yager, 1982].

There are a large number of ways to meaningfully define *and* and *or* for use in decision-making with fuzzy logic. Bellman *et al.* [1977] maintain that the selection of the fuzzy connectives should be a choice that depends on the situation. The following sections will outline some of the basic properties that the *and* and *or* connectors should possess, then detail some additional properties that may be required for specific applications.

4.2 Basic Properties of the *Intersection* or *And* Connector

The development of the criteria for the definition and selection of *and* are such that there should be a strong connection between the real and mathematical characteristics of the *and*. Thus there are certain basic properties that every *and* should satisfy. Before describing these basic properties of the *and* connector the following definition is required.

Definition 1: For the interval $\mathbf{I} \in [0,1]$, let *and* be:

(*i*) a binary operation $\mathbf{t}$ on $\mathbf{I}$, and

(*ii*) a mapping $\mathbf{I}\times\mathbf{I}\rightarrow\mathbf{I}$,

such that $x \mathbf{t} y = z$, for $x, y, z \in \mathbf{I}$

The basic properties for the *and* connector are listed below along with a short interpretation of the property.

(1) Commutativity: $x \mathbf{t} y = y \mathbf{t} x$

This property implies that when two expressions are connected by *and* the order is not important.

(2) Associativity: $(x \mathbf{t} y) \mathbf{t} z = x \mathbf{t} (y \mathbf{t} z)$

This property implies that when a finite number of expressions are connected by *and* then the expressions can be combined in any order.

(3) Boundary Conditions:

$1 \mathbf{t} 0 = 0 \mathbf{t} 1 = 0$

$1 \mathbf{t} 1 = 1$

$0 \mathbf{t} 0 = 0$

This property requires that the *and* connector collapse to the *and* operation in conventional two-valued logic.

(4) Monotonicity: If $x \geq v$ and $y \geq w$

then $x \mathbf{t} y \geq v \mathbf{t} w$

This condition insures that the *and* operation is positively associated with the elements in its argument. Any binary operation, (an operation on two variables) satisfying these four condition is called a triangular norm (*t-norm*).

(5) Continuity: For any fixed $x \in \mathbf{I}$,

$x \mathbf{t} y$ is continuous on $\mathbf{I}$ for all $y \in \mathbf{I}$.

This property insures against a situation in which a very small increase in one of the arguments produces a large change in the resulting combination.

It is easy to see that if all five conditions are satisfied, then the following two additional properties hold for any $x \in \mathbf{I}$,

$$\textbf{\textit{And} Property 1:} \qquad 0 \; t \; x = x \; t \; 0 = 0$$

$$\textbf{\textit{And} Property 2:} \qquad 1 \; t \; x = x$$

There are a number of definition which satisfy the five basic properties defined above, two of them are:

	$x \; t \; y \; = \; \min(x, y)$	*min* operation
and	$x \; t \; y \; = \; x \cdot y$	*product* operation

Based on these definitions of *and*, the set of *and* connectors on the interval [0,1] is non-empty and non-unique.

4.3 Basic Properties of the *Union* or *Or* Connector

Having defined an *and*, a complementary *or* can be obtained by defining a negation and then requiring the *and* and the *or* satisfy De Morgan's laws. The negation operator is defined and denoted as follows:

$$\overline{x} = 1 - x \qquad \text{for } x \in \mathbf{I}$$

De Morgan's Laws require:

$$\overline{(x \cap y)} = \overline{x} \cup \overline{y}$$

$$\overline{(x \cup y)} = \overline{x} \cap \overline{y}$$

Definition 2: For the interval $\mathbf{I} \in [0,1]$, let *or* be:

(*i*) a binary operation s on $\mathbf{I}$, and

(*ii*) a mapping $\mathbf{I} \times \mathbf{I} \rightarrow \mathbf{I}$,

such that $x \; s \; y = z$, for $x, y, z \in \mathbf{I}$

The basic properties for the *or* are outlined below:

(1) **Commutativity:** $x \text{ s } y = y \text{ s } x$

(2) **Associativity:** $(x \text{ s } y) \text{ s } z = x \text{ s } (y \text{ s } z)$

(3) **Boundary Conditions:** $1 \text{ s } 0 = 0 \text{ s } 1 = 1$

$1 \text{ s } 1 = 1$

$0 \text{ s } 0 = 0$

Note that the boundary conditions for the *or* differ from the *and*.

(4) **Monotonicity:** If $x \geq v$ and $y \geq w$

then $x \text{ s } y \geq v \text{ s } w$

Any binary operation satisfying these four condition is called a triangular conorm (*t-conorm* or *s-norm*).

(5) **Continuity:** For any fixed $x \in \mathbf{I}$,

$x \text{ s } y$ is continuous on $\mathbf{I}$ for all $y \in \mathbf{I}$.

Again it is easy to see that if all five conditions are satisfied the following two additional properties hold for any $x \in \mathbf{I}$,

***Or* Property 1:** $0 \text{ s } x = x \text{ s } 0 = x$

***Or* Property 2:** $1 \text{ s } x = 1$

There are a number of definitions which satisfy the five basic properties for the *or* operation, two of which are:

$x \text{ s } y = \max(x, y)$ *max* operation

and $x \text{ s } y = x + y - x \cdot y$ $\oplus$-*sum* operation

Based on these definitions of *or*, it is clear that the set of *or* on the interval [0,1] is also non-empty and non-unique.

4.4 Specific Properties of the *Product* and *Min* Operations

Based on De Morgan's Laws, it is clear that the *max* and ⊕-*sum* operations, used to define the *and* connector, are the *conorms* for the *min* and *product* operations, respectively, which define the *or* connector. This section will present the properties specific to each of these operations and their associated conorms and negations.

1. The *min* t-norm and its associated t-conorm, *max*, and the negation were suggested by Zadeh [1965] and are the most popular and widely used the in literature. They are defined and denoted as:

$$\mathbf{t}_1(x, y) = \min(x, y)$$

$$\mathbf{s}_1(x, y) = \max(x, y)$$

$$\mathbf{n}_1(x) = 1-x$$

2. The *product* t-norm and its associated t-conorm, ⊕-*sum*, and the negation, called probabilistic operators, are defined and denoted as:

$$\mathbf{t}_2(x, y) = x \cdot y$$

$$\mathbf{s}_2(x, y) = x+y-x \cdot y$$

$$\mathbf{n}_2(x) = 1-x$$

Additional conditions that can be imposed on the *and* operation which result in a uniquely specified *and* are listed.

(1) Idempotency: $x \,\mathbf{t}\, x = x$ for $\forall\, x \in \mathbf{I}$

The significance of the idempotency condition is that two proposition having the same degree of truth combine to form a value having that same degree of truth. This condition is satisfied only by the *min* operation.

(2) Non-Compensatory: If $v > w$, then $v \,\mathbf{t}\, v > w \,\mathbf{t}\, x$ for $\forall\, x, v, w \in \mathbf{I}$

The unique solution to the non-compensatory property is the *min* operation.

(3) Conservativeness: $a \,\mathbf{t}\, \mathrm{b} = (a - \mathrm{K}) \,\mathbf{t}\, (\mathrm{b} + \mathrm{K})$ for $\mathrm{K} \ni a - \mathrm{K}$ and $b + \mathrm{K} \in \mathbf{I}$

The conservative property of adding a fixed amount to one argum[illegible] [illegible] su[illegible] the same amount from the other argument with the same *and* result is not satisfied by [illegible] or *product*.

(4) Proportionate Interaction: $x \mathbf{t} y = (x/k) \mathbf{t} (y \cdot k)$

for x/k and $y \cdot k \in \mathbf{I}$

The proportionate interaction property is uniquely defined by the *product* operation.

(5) Addition in Both Variables: $(x_1 + x_2) \mathbf{t} y = x_1 \mathbf{t} y + x_2 \mathbf{t} y$

$x \mathbf{t} (y_1 + y_2) = x \mathbf{t} y_1 + x \mathbf{t} y_2$

for $\forall x_n, y_n \in \mathbf{I}$

Addition in both variables is defined only for the *product* operation.

(6) Law of Contradiction: If *not* x is defined as *not* $x = 1 - x$, then

$x \mathbf{t} (1 - x) = 0$ for $\forall x \in \mathbf{I}$

The law of contradiction is satisfied by both *min* or *product*, for $x \neq 0$. This law is also called the *law of excluded middle*.

Properties specific to a *t-norm* and its *t-conorm* combinations are as follows:

(7) Distributivity: $x \mathbf{t} (y \mathbf{s} z) = (x \mathbf{t} y) \mathbf{s} (x \mathbf{t} z)$

$x \mathbf{s} (y \mathbf{t} z) = (x \mathbf{s} y) \mathbf{t} (x \mathbf{s} z)$

for all $x, y, z \in \mathbf{I}$

This property is satisfied only by the t-norm *min* and its associated t-conorm *max* [Bellman *et al.*, 1973 in Yager, 1982].

(8) Absorption: $x \mathbf{t} (x \mathbf{s} y) = x$

$x \mathbf{s} (x \mathbf{t} y) = x$

for all $x, y \in \mathbf{I}$

This property is satisfied only by the t-norm *min* and its associated t-conorm *max* [Gupta *et al.*, 1991(a)].

The properties reviewed above have been given for the *and* or *t-norms* only [Yager, 1982], but if a *t-norm* possess a particular property or condition then its corresponding *t-conorm* also possess that same trait [Gupta *et al.*, 1991(a)].

Now consider the *max-product* composition, which is the focus of this thesis, and the properties of *distributivity* and *absorption*. The combination of *t-norm product* with *t-conorm max* are satisfied only for **t** over **s** for *distributivity* and **s** over **t** for *absorption*, as illustrated below.

(7) Distributivity: $x \mathbf{t} (y \mathbf{s} z) = (x \mathbf{t} y) \mathbf{s} (x \mathbf{t} z)$ (4.1)

for all $x, y, z \in \mathbf{I}$

Proof:

$x \mathbf{t} (y \mathbf{s} z) = x \cdot (max(y, z))$

$= max(x \cdot y, x \cdot z)$

$= (x \mathbf{t} y) \mathbf{s} (x \mathbf{t} z)$

$x \mathbf{s} (y \mathbf{t} z) = max(x, y \cdot z)$

$\neq max(x, y) \cdot max(x, z)$

$\neq (x \mathbf{s} y) \mathbf{t} (x \mathbf{s} z)$

(8) Absorption: $x \mathbf{s} (x \mathbf{t} y) = x$ (4.2)

for all $x, y \in \mathbf{I}$

Proof:

$x \mathbf{t} (x \mathbf{s} y) = x \cdot (max(x, y)$

$= max(x \cdot x, x \cdot y)$

$\neq x$

$x \mathbf{s} (x \mathbf{t} y) = max(x, x \cdot y)$

$= x$

4.5 Selection of an *And* Connector

Selection of the *and* connector depends of the type of scale that is being used. The scale reflects the kind of relationship that exists between the elements or the amount of information available when comparing the elements. There are three types of scales, *ordinal*, *cardinal* and *absolute*. An *ordinal* scale is used when the only information available to relate the elements to one another is larger or smaller. This type of scale effectively orders the elements according to size or magnitude. A *cardinal* scale is used when, in addition to the order of the elements, the relative intensity of the elements to each other is available. The *absolute* scale results when enough information is available to assign a unique value to each element. As the amount of information increases from the *ordinal* scale to the *absolute* scale so does the number of operations to define the *and* connector. Thus the absolute scale has the most operations for the definition of *and*.

Yager [1979] has shown that if x and y are elements from the same *ordinal* scale, the only way to formulate the *and* operation, to satisfy the basic properties, is through the *min* operation, which was suggest by Zadeh [1965]. Additionally, if the x and y are elements from two different *ordinal* scales there is no meaningful way to formulate the intersection.

Yager [1982] has shown that if x and y are elements of two different *cardinal* scales then the unique form of *and* for this situation is the *product* of the two arguments. So if only *ordinal* or *cardinal* information is available, the form of the *and* connector is uniquely specified.

Information available from the absolute scale does not lead to a unique specification of the *and* operation, as expected, since it relates the most information.

Di Nola *et al.* [1989] describe the concept of modeling different types of *and* through the following statements:

"*red* and *new* car"

"*large* and *expensive* house"

In the first statement the properties of the car are quite obviously unrelated to one another. One can have a *new* car without it being *red* or visa versa. In the second statement the properties of the house are, to a certain extent, related to each other. One would expect a *large* house to be *expensive*. Modeling the *and* in these examples one might suggest *min* for the first statement and *product* for the second.

Yager [1980] has proposed the following general *intersection* operator:

Definition 3: Assume A and B are fuzzy subsets of X with grades of membership in the unit interval. Then define:

$$C_p(x) = A(x) \cap_p B(x) \tag{4.3}$$

where $$C_p(x) = 1 - \min[1, ((1 - A(x))^p + (1 - B(x))^p)^{1/p}] \tag{4.4}$$

for $p \geq 1$

Yager showed that,

$$\text{if } p = 1, \quad \text{then} \quad C_1(x) = \max[0, A(x) + B(x) - 1]$$

$$\text{if } p = \infty, \quad \text{then} \quad C_\infty(x) = \min[A(x), B(x)]$$

which is the original definition of fuzzy intersection by Zadeh [1965]. And for all $p \geq 1$ the definition collapses to the ordinary definition of intersection when the grades of membership lie in the set [0,1]. Yager also showed that this operator possessed all the properties of a *t-norm.*

The significance of the selection of the p parameter is interpreted as the *strength* of the *and* or *intersection.* In other words p is a measure of how strong the demand is for simultaneous satisfaction. So when $p = 1$, we have the strongest and most demanding *and* and when $p = \infty$, we have the least demanding *and* which corresponds to the suggestion by Bellman *et al.* [1977] that when faced with no information the default selection should be *min* for *intersection.*

It can be shown that:

$$x + y - 1 \;\leq\; x \cdot y \;\leq\; \min(x, y)$$

The relationship of the *product* and *min* operators versus Yager's [1980] intersection operator is illustrated in Figure 4.1. Clearly from the graph of Figure 4.1 the *product* ≤ *min*, with the *product* operator offering an *averaged* or less severe solution, similar to the Yager operator.

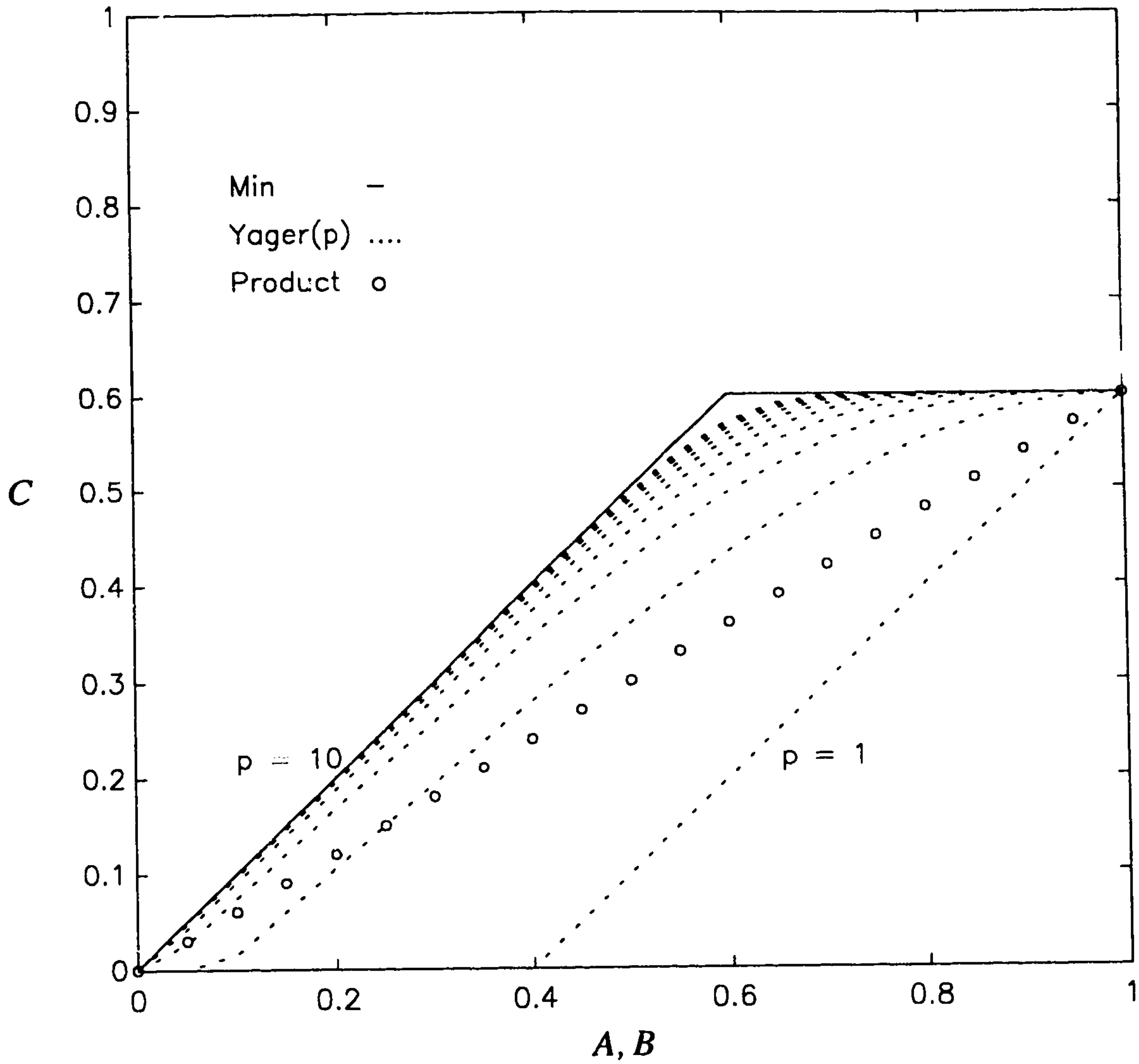


Figure 4.1: Relation of Product Operator to Minimum Operator
(A = [0, 1]; B = 0.6)
(Minimum —; Yager $C_p(x)$ ---; Product o)

4.6 Selection of the *Or* Connector

For the *or* connector, Yager [1980] proposed the following general *union* operator:

Definition 4: Assume A and B are fuzzy subsets of X with grades of membership in the unit interval. Then define:

$$D_p(x) \quad A(x) \cup_p B(x) \tag{4.5}$$

where $$D_p(x) = \min[1, (A(x)^p + B(x)^p)^{1/p}] \tag{4.6}$$

for $p \geq 1$

Yager showed that,

if $p = 1$, then $D_1(x) = Min[1, A(x) + B(x)]$

if $p = \infty$, then $D_\infty(x) = Max[A(x), B(x)]$

which is the original definition of fuzzy union by Zadeh [1965]. And for all $p \geq 1$ the definition collapses to the ordinary definition of intersection when the grades of membership lie in the set [0,1]. Yager also showed that this operator possessed all the properties of a *t-conorm.*

The significance of the selection of the p parameter is interpreted as a measure of the degree of interchangeability of the *or*. When $p = 1$, we have the least interchangeable *or* or *exclusive or* (*i.e. either... or*) and when $p = \infty$, when have the most interchangeable *or*. The situation of most interchangeability corresponds to the suggestion by Bellman *et al.* [1977] that faced with no information the default selection should be *max* for *union.*

It can be shown that:

$$x + y \geq x+y-x \cdot y \geq \max(x, y)$$

The relationship of the *algebraic sum* and *max* operators versus Yager's [1980] union operator is illustrated in Figure 4.2. Again from the graph of Figure 4.2 the *max* ≤ *algebraic sum*, with the *algebraic sum* operator offering a less severe solution, similar to the Yager operator.

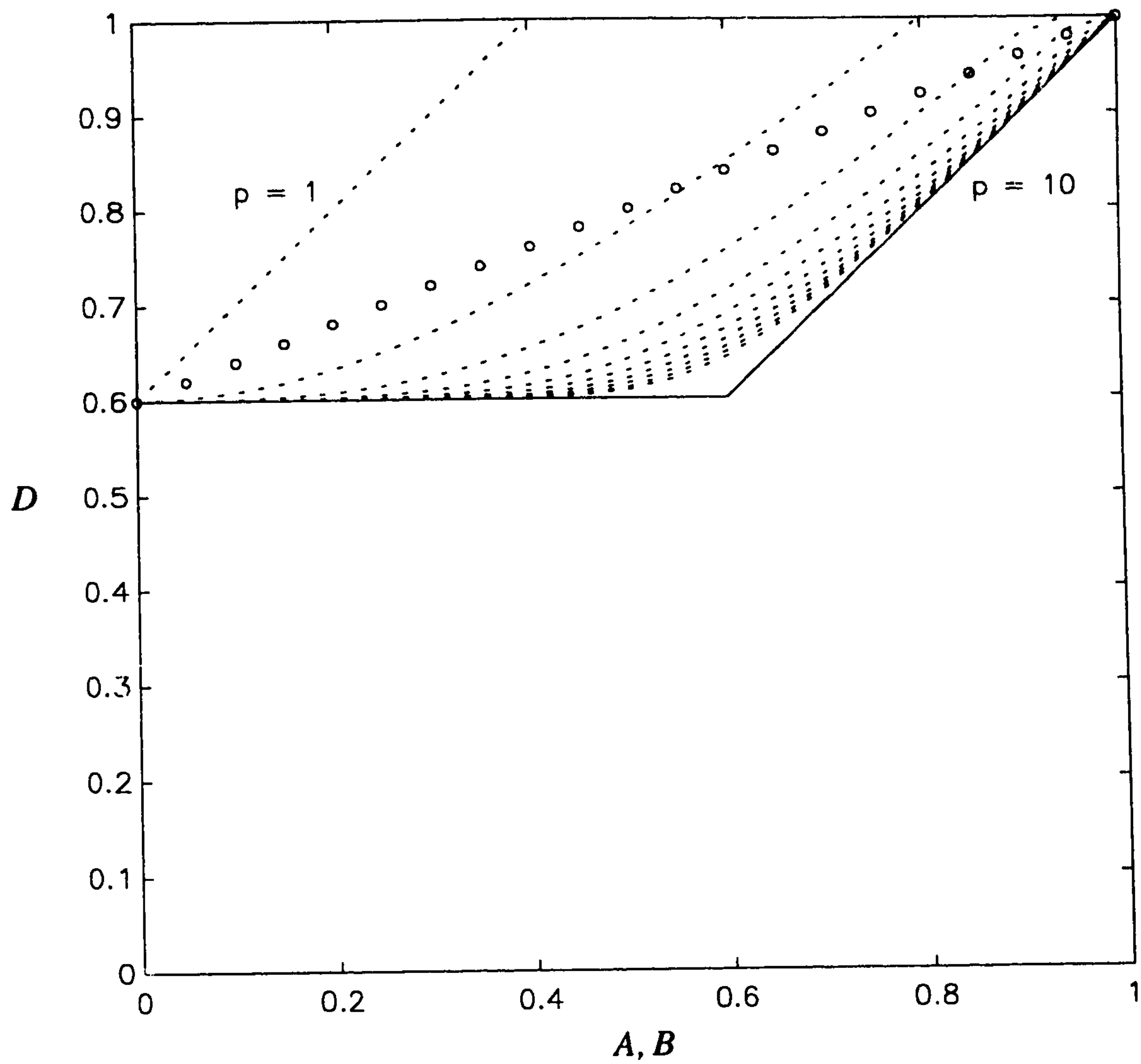


Figure 4.2: Relation of ⊕-Sum Operator to Maximum Operator
(A = [0, 1]; B = 0.6)
(Maximum —; Yager $D_p(x)$ ---; ⊕-Sum o)

4.7 *Max-Product* Composition versus *Max-Min* Composition

The *min* and *max* *t*-operators, as suggested by Zadeh [1965], have been used in almost every design of fuzzy logic controllers [Gupta *et al.*, 1991(a)]. There is an increasing amount of work that shows the other types of *t*-operators may work better in some situations, particularly for decision making processes [Gupta *et al.*, 1991(b)]. Dubois *et al.* [1986] also suggest that the *product* operator may be preferred to the *min* operator in some situations.

Which ever operator is chosen to aggregate the fuzzy sets, the following eight important criteria, suggested by Zimmermann [1985], should be considered in the selection.

(*i*) Axiomatic Strength

The axioms or properties which the operator must satisfy should be carefully considered. Of course, the less limiting are the axioms which must be satisfied the more flexible the operator.

(*ii*) Empirical Fit

Fuzzy sets are used to model real situations or systems, so the operator chosen must be appropriate to model this behaviour. The quality of the choice is normally proven by empirical testing.

(*iii*) Adaptability

If it is required to use a small number of operators to model a number of different situations, then the operators must be adaptable to the specific context of the application. The *max*, *min* and *product* operators cannot be adapted at all. They are only acceptable in the situations in which they fit. On the other hand, Yager's operators can be adapted to varying applications by setting the *p*'s appropriately

(*iv*) Numerical Efficiency

Where Yager's operators may be adaptable, they are not numerically efficient. They require considerable more computations than the *max*, *min* and *product* operators. This is an important consideration when a large problem is involved.

(*v*) Compensation

Compensation, also known as interactivity, for the data operators of fuzzy sets requires that an increase in one argument to the aggregation can be compensated by or traded off with a decrease in the second argument, or vice-versa, to achieve the same result. The *product* operator is compensatory, while the *min* operator is not.

(*vi*) Range of Compensation

If a convex combination is used with the *min* and *max* operators, a compensation could occur in the range between *min* and *max*. The *product* operator allows compensation in the interval [0,1]. Generally the larger the range of compensation the better the compensatory operator.

(*vii*) Aggregating Behaviour

When combining fuzzy sets the degree of membership of the aggregated set depends on the frequency of a particular set and the number of sets combined. If a *product* operator is used, each additional fuzzy set will decrease the degree of membership of the resulting aggregate. This may or may not be a desirable feature.

(*viii*) Required Scale Level of Membership Functions

The scale level (*ordinal*, *cardinal*, or *absolute*) on which the membership information can be obtained depends on the operator. In general, the operator which requires the lowest scale level is the most preferable for information gathering.

According to Yager's generalize connectives [1980], the *max-min* composition corresponds to the situation of $p = \infty$, which is the *least* interchangeable or *exclusive or* and the *least* demanding *and*. The *max-product* composition corresponds to the situation of a *least* interchangeable or *exclusive or* with a variably demanding *and*.

Thole *et al.* [1979] compared the *min* operator and the *product* operator for the formulation of the fuzzy intersection. In this experiment the authors asked their subject to rate fifty preselected objects to determine the belonging of each object in three given class. The results were then combined using each operator and compared against a linearized standard. The results showed that neither connective operator provided a suitable model for the intersection of the subjectively chosen categories. However, the *min* operator appeared to be slightly better for this experiment.

However similar work carried out by Oden [1977] resulted in the *product* operator being clearly superior in describing the cognitive conjunctic . *and*. In this study the author had his subjects judge the truthfulness of the actual conjunction of two statements.

These two works illustrate the different psychological meanings for the connective *and* and confirm the need for different mathematical operators to simulate the various processes.

Gupta *et al.* [1991(b)] compared various combinations of *conorms* and *norms* through applications in a fuzzy logic controller. Performance was assessed based on the speed of response. The *t*-operators of particular interest in this study are:

	t-conorm	*t*-norm
1.	$\max(x, y)$	$\min(x, y)$
2.	$x + y - x \cdot y$	$x \cdot y$

The results of their study show that

max-product **is faster than** ***algebraic sum-product*** **is faster than** ***max-min***

Gupta *et al.* [1991(b)] generalized that the response is faster for the couples with a *smaller t-norm* (i.e. ***t*-norm**$_2$ ≤ ***t*-norm**$_1$) and a *larger* t-conorm (i.e. ***t*-conorm**$_2$ ≥ ***t*-conorm**$_1$). This result provides a basis for the superiority of the *max-product* composition over the *max-min* composition in some solutions.

In relational work, Pedrycz [1984(a)] used a Cartesian product formulation for a batch identification of ***R*** using the Box-Jenkins data [1970]. Then 1-step ahead predictions were calculated using the model obtained for both *max-min* and *max-product* compositions. The results show that the *max-product* composition provided *better* results using a Euclidean minimization criteria.

Xu *et al.* [1987] also identified the Box-Jenkins data [1970] using a on-line identification algorithm. The Hamming minimization criteria then determined the best composition. Demonstrating with two different examples these authors showed that the *max-product* composition was superior to the *max-min* composition.

While much of the current research on fuzzy relational equations deals specifically with the *max-min* composition solutions, there is a growing amount of work which generalize using *max-t-norm* composition, DiNola *et al.* [1984]; Pedrycz, [1983]; Pedrycz, [1984(b)]; Pedrycz, [1985]; Miyakoshi *et al.* [1985]; Di Nola *et al.* [1989], to which both the *max-min* and the *max-product* belong.

4.8 Summary

Traditionally, relational-based fuzzy systems theory is based on the *max-min* composition, in spite of research confirming that the *max-product* composition is superior in some instances. The focus of this thesis is to explore the theory and applicability of the *max-product* composition, in a control setting, in light of this research.

Chapter 4 references several source of research confirming the superiority of the *max-product* composition, as well as presenting the reasons why this superiority exists. The literature review of Chapter 4 compares, in several examples, the physical interpretation and ability of the *max-min* and *max-product* compositions.

In the literature review of this Chapter, two properties of the *max-product* composition are expanded.

(1) Proof that the *max-product* composition is *distributive* for only x **t** (y **s** z), not x **s** (y **t** z).

(2) Proof that the *max-product* composition is *absorptive* for only x **s** (x **t** y), not x **t** (x **s** y)

This expansion increases the basis of theory available for the *max-product* composition which is a composition in which the *norm* operator, *product*, is not based on its associate *conorm* operator, ⊕ *sum*, but with the *conorm* operator *max*.

REFERENCES

[1] Bellman, R.E., Zadeh, L.A., "Local and Fuzzy Logic", In: *Modern Uses of Multiple-Valued Logic*, Dunn, J.M., Epstein, G., (Editors), D. Reidel Publishing Company, Dordrecht-Holland, pp. 103-165, 1977.

[2] Bellman, R.E., Giertz, M., "One the Analytic Formalism of the Theory of Fuzzy Sets", *Information Sciences*, Vol. 5, pp. 103-165, 1973.

[3] Box, G.E.P., Jenkins, G.M., *Time Series Analysis, Forecasting and Control*, Holden Day, San Francisco, 1970.

[4] Di Nola, A., Pedrycz, W., Sessa, S., Pei-Zhuang, W., "Fuzzy Relation Equations Under a Class of Triangular Norms: A Survey and New Results", *Stochastica*, Vol.. 8, pp. 99-145, 1984.

[5] Di Nola, A., Sessa, S., Pedrycz, W., Sanchez, E., *Fuzzy Relational Equations and Their Applications to Knowledge Engineering*, Kluwer Academic Publishers, Norwell, MA., 1989.

[6] Dubois, D., Prade, H., "New Results about Properties and Semantics of Fuzzy Set-Theoretic Operators", In: *Fuzzy Sets*, Wang, P.P., Chang, S.K., (Editors), Plenum Press, New York, pp. 59-75, 1986.

[7] Gupta, M.M., Qi, J., "Theory of T-Norms and Fuzzy Inference Methods", *Fuzzy Sets and Systems*, Vol. 40, pp. 431-450, 1991(a).

[8] Gupta, M.M., Qi, J., "Design of Fuzzy Logic Controllers based on Generalized T-Operators", *Fuzzy Sets and Systems*, Vol. 40, pp. 473-489, 1991(b).

[9] Miyakoshi, M., Shimbo. M., "Solutions of Composite Fuzzy Relational Equations with Triangular Norms", *Fuzzy Sets and Systems*, Vol. 16, pp. 53-63, 1985.

[10] Pedrycz, W., "An Identification Algorithm in Fuzzy Relational Systems", *Fuzzy Sets and Systems*, Vol. 13, pp. 153-167, 1984.

[11] Pedrycz, W., "Some Applicational Aspects of Fuzzy Relational Equations in Systems Analysis", *International Journal of General Systems*, Vol. 9, pp. 125-132, 1983.

[12] Pedrycz. W., "An Identification Algorithm in Fuzzy Relation Systems", *Fuzzy Sets and Systems*, Vol. 13, pp. 153-167, 1984.

[13] Pedrycz, W., "Identification in Fuzzy Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 14, pp. 361-366, 1984.

[14] Pedrycz, W., "On Generalized Fuzzy Relational Equations and Their Applications", *Journal of Mathematical Analysis and Applications*, Vol. 107, pp. 520-536, 1985.

[15] Pedrycz, W., "Fuzzy Modeling: Fundamentals, Construction and Evaluation", *Fuzzy Sets and Systems*, Vol. 41, pp. 1-15, 1991.

[16] Oden, G.C., "Integration of Fuzzy Logical Information", *Journal of Experimental Psychology: Human Perception and Performance"*, Vol. 106, pp. 565-575, 1977.

[17] Thole, U., Zimmermann, H.-J., Zysno, P., "On the Suitability of Minimum and Product Operators for the Intersection of Fuzzy Sets", *Fuzzy Sets and Systems*, Vol. 2, pp. 167-180, 1979.

[18] Xu, C.-W., Lu, Y.-Z., "Fuzzy Model Identification and Self-Learning for Dynamic Systems", *EEE Transactions on Systems, Man and Cybernetics*, Vol., 17, pp. 683-689, 1987.

[19] Yager, R.R., "A Measurement-Information Discussion of Fuzzy Union and Intersection", *International Journal of Man-Machine Studies*", Vol. 11, pp. 189-200, 1979.

[20] Yager, R.R., "On a General Class of Fuzzy Connectives", *Fuzzy Sets and Systems*, Vol. 4, pp. 235-242, 1980.

[21] Yager, R.R., "Some Procedures for Selecting Fuzzy Set-Theoretic Operators", *International Journal of General Systems*, Vol. 8, pp. 115-124, 1982.

[22] Zadeh, L.A., "Fuzzy Sets", *Information and Control*, Vol. 8, pp. 338-353, 1965.

[23] Zadeh, L.A., "Outline of a New Approach to the Analysis of Complex Systems and Decision Processes", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 3, pp. 28-44, 1973.

[24] Zimmermann, H.-J., *Fuzzy Sets Theory - and Its Applications*, Kluwer-Nijhoff Publishing, Boston, 1985.

CHAPTER 5 RESOLUTION OF FUZZY RELATIONAL EQUATIONS [1]

The existence of a family of solutions for fuzzy relational equations, instead of a simple, unique solution, leads to a certain kind of robustness of this form of equation, and confirms the fact that precise values (up to a certain limit) of the membership function are not necessary when working with fuzzy quantities.

[Pedrycz, 1990]

5.1 Introduction

The resolution of fuzzy relational equations means:

(1) identification of the fuzzy relation, $\boldsymbol{R}$, of the fuzzy relational equation, $y = x \circ \boldsymbol{R}$, or

Given x and y, find $\boldsymbol{R}$ such that $y = x \circ \boldsymbol{R}$,

(2) determining the causality, x, of the fuzzy relational equation, $y = x \circ \boldsymbol{R}$, or

Given $\boldsymbol{R}$ and y, find x such that $y = x \circ \boldsymbol{R}$.

The concept of inverse calculations for fuzzy relational equations relates directly to:

(1) fuzzy identification, and

(2) control of fuzzy processes.

The fact that fuzzy inverses are non-unique carries the positive aspect of robustness of the relational matrix for identification, as well as the negative aspect of potential chatter of the input signal calculated by the causal inverse. Unfortunately, as will be shown in Chapter 7, inverses are not well suited to systems when the inverse is not exact.

Most of the literature available on inverse calculations deals with the *max-min* composition. However, the calculations are the same, as shown by DiNola *et al.* [1984], for all *max-t-norm* compositions. This Chapter present a literature survey of research into the resolution of fuzzy relational equations with *max-min* and/or *max-t-norm* compositions.

[1] A version of this chapter has been published. Bourke M.M., Fisher D.G., 1994. (*Fuzzy Sets and Systems*, 63: 111-115).

5.2 Notation

The following definitions outline the notation which will be used throughout this chapter.

Definition 1: Let $\mathcal{U}$, $\mathcal{V}$ and $\mathcal{W}$ be fuzzy universes of discourse and let $u=\{u_i \mid i\in \mathbf{I}_m=\{1,2, ..., m\}\}\in \mathcal{U}$, $v=\{v_j \mid j\in \mathbf{J}_n=\{1,2, ..., n\}\}\in \mathcal{V}$ and $w=\{w_k \mid k\in \mathbf{K}_p=\{1,2, ..., p\}\}\in \mathcal{W}$ be the fuzzy sets associated with each universe of discourse.

Definition 2: Let the fuzzy relations $\boldsymbol{Q}\in \mathcal{U}\times\mathcal{V}$, $\boldsymbol{R}\in \mathcal{V}\times\mathcal{W}$ and $\boldsymbol{T}\in \mathcal{U}\times\mathcal{W}$ be defined such that:

$\boldsymbol{Q}=\{((u_i, v_j), q_{ij}) \mid i\in \mathbf{I}_m;\ j\in \mathbf{J}_n\}$ is the mapping $\boldsymbol{Q}$: $\mathcal{U}\times\mathcal{V}\rightarrow [0,1]\times[0,1]$,

$\boldsymbol{R}=\{((v_j, w_k), r_{jk}) \mid j\in \mathbf{J}_n;\ k\in \mathbf{K}_p\}$ is the mapping $\boldsymbol{R}$: $\mathcal{V}\times\mathcal{W}\rightarrow [0,1]\times[0,1]$

$\boldsymbol{T}=\{((u_i, w_k), t_{ik}) \mid i\in \mathbf{I}_m;\ k\in \mathbf{K}_p\}$ is the mapping $\boldsymbol{T}$: $\mathcal{U}\times\mathcal{W}\rightarrow [0,1]\times[0,1]$

where q_{ij}, r_{jk} and $t_{ik}\in [0,1]$ are the grades of membership in (u_i, v_j), (v_j, w_k) and (u_i, w_k), respectively.

Definition 3: Let $\boldsymbol{a}\in \mathcal{V}$ and $\boldsymbol{b}\in \mathcal{W}$ be fuzzy sets defined as:

$\boldsymbol{a}=\{(v_j, a_j) \mid j\in \mathbf{J}_n\}$ is the mapping $\boldsymbol{a}$: $\mathcal{V}\rightarrow [0,1]$

$\boldsymbol{b}=\{(w_k, b_k) \mid k\in \mathbf{K}_p\}$ is the mapping $\boldsymbol{b}$: $\mathcal{W}\rightarrow [0,1]$

where a_j, and $b_k\in [0,1]$ are the grades of membership in v_j, and w_k, respectively

For simplicity of notation, the fuzzy sets and fuzzy relations will usually be identified only over their universe of discourse (i.e. $\boldsymbol{a}(v)$, $\boldsymbol{R}(v, w)$).

5.3 Problem Statements

The original work by Sanchez [1976] considered the resolution of the following fuzzy relational equation with *max-min* composition :

$$\boldsymbol{T}(u_i, w_k) = \boldsymbol{Q}(u_i, v_j)\circ\boldsymbol{R}(v_j, w_k) = \bigvee_j [\boldsymbol{Q}(u_i, v_j)\wedge\boldsymbol{R}(v_j, w_k)] \tag{5.1}$$

A short time later, Sanchez [1977] then considered the simplified version of equation (5.1), where $\mathcal{U}$ is defined over a single point:

$$b(w_k) = a(v_j) \circ R(v_j, w_k) = \bigvee_j [a(v_j) \wedge R(v_j, w_k)] \tag{5.2}$$

In equations (5.1) and (5.2) the symbols $\vee$ and $\wedge$ denote the fuzzy set operators or connectors *max* and *min*, respectively. And the symbol $\circ$ denotes the *max-min* operator.

Di Nola *et al.* [1984] showed that if the *min* operator $\wedge$ in equations (5.1) and (5.2) was replaced with a *t-norm* operator then the following fuzzy relation equations should be considered:

$$T(u_i, w_k) = Q(u_i, v_j) \textcircled{t} R(v_j, w_k) = \bigvee_j [Q(u_i, v_j) \; t \; R(v_j, w_k)] \tag{5.3}$$

$$b(w_k) = a(v_j) \textcircled{t} R(v_j, w_k) = \bigvee_j [a(v_j) \; t \; R(v_j, w_k)] \tag{5.4}$$

In equations (5.3) and (5.4) the symbol $\vee$ still denotes the fuzzy set connector *max* while the symbol t denotes any *t-norm* operator. The symbol $\textcircled{t}$ represent the *max-t-norm* composition operator.

There are two basic inverse problems to be investigated and resolved, that of fuzzy identification and fuzzy cause. When these two problems are applied to each of equations (5.3) and (5.4) the result is the four problem statements listed below:

(1) "Given the fuzzy relations R and b,
find all fuzzy sets a such that $a \textcircled{t} R = b$".

(2) "Given the fuzzy relations R and T,
find all fuzzy relations Q such that $Q \textcircled{t} R = T$".

(3) "Given the fuzzy sets a and b,
find the fuzzy relation R such that $a \textcircled{t} R = b$".

(4) "Given the fuzzy relations Q and T,
find the fuzzy relation R such that $Q \textcircled{t} R = T$".

Problem statements (1) and (2) represent the search for a fuzzy cause, while statements (3) and (4) represent fuzzy identification. All four problem statements are valid when $\textcircled{t}$ represents either *max-min* or *max-product* composition.

5.4 Inverse Structure

An interesting aspect of the inverse solution of fuzzy relational equations is that the inverse solution is most often *non-unique*. As shown by Czogala *et al.* [1982] and Higashi *et al.* [1984], the solution is bounded above by a *unique maximum* solution and is bounded below by *several minimum* solutions. In some instances a *unique* solution does exist and this situation has been investigated by Sessa [1989].

A schematic of the solution relationship is illustrated in Figure 5.1. As shown, the solution is bounded above by the single *maximum* solution, $\boldsymbol{R}_{max}$. The branches of the lower solutions converge to a single point which is the *union* of all the *minimum* solutions, $\bigcup \boldsymbol{R}_{min}$.

Resolution of fuzzy relational equations, composed by *max-min* composition, was first presented by Sanchez [1976]. Since this beginning there has been continued research into the properties and solutions of fuzzy relational equations with the *max-min* and *max-t-norm* compositions. The literature survey that follows reviews the current and relevant theory for resolving the *max-min* composition of the fuzzy relational equations presented by problem statements (1) to (4), since this information has a greater availability. Relevant literature for *max-product* and *max-t-norm* compositions is then cited in the next section and a Tabular Summary of all referenced material is provided.

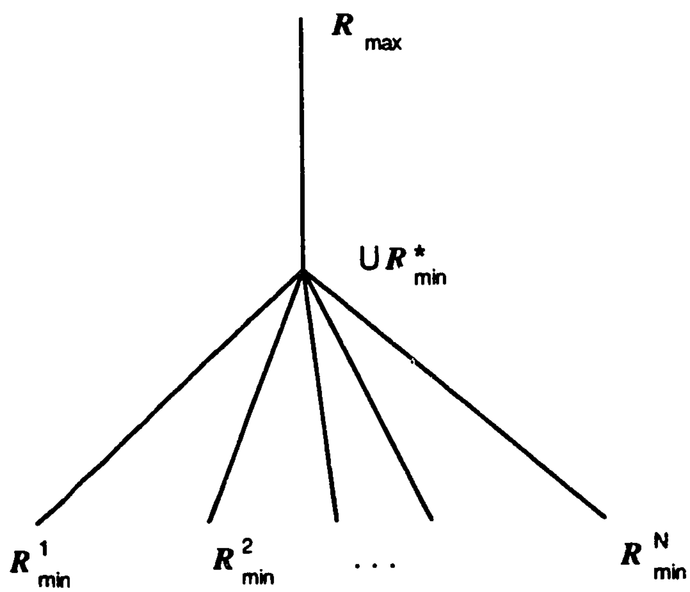


Figure 5.1: Illustration of Solution Relationships

5.5 The *Greatest* Solution

Studying problem statement (2) and (4) for the *max-min* composition Sanchez [1976] established the existence of the inverse solution by determining the greatest solution $\boldsymbol{Q}$, and the greatest solution $\boldsymbol{R}$, for problem statements (2) and (4), respectively. Prior to presenting the two fundamental theorems of Sanchez [1976] the following definitions are required.

Definition 4: For a and $b \in [0,1]$, the α-operator is defined as:

$$a \,\alpha\, b = \begin{cases} 1 & if \quad a \leq b \\ b & if \quad a > b \end{cases} \tag{5.5}$$

The composition $a \,\alpha\, b$ is call the *relative pseudo-complement* of a in b.

Definition 5: Let $\boldsymbol{Q} \in \mathcal{U} \times \mathcal{V}$ and $\boldsymbol{T} \in \mathcal{U} \times \mathcal{W}$ be two fuzzy relations. Define $\boldsymbol{R} = \boldsymbol{Q} \circledast \boldsymbol{T}$, $\boldsymbol{T} \in \mathcal{V} \times \mathcal{W}$, the $\circledast$-composite fuzzy relation of $\boldsymbol{Q}$ and $\boldsymbol{T}$ by:

$$\boldsymbol{R}(v, w) = \boldsymbol{Q} \circledast \boldsymbol{T}(v, w) = \boldsymbol{Q}(u, v) \circledast \boldsymbol{T}(u, w) = \bigwedge_u [\boldsymbol{Q}(u, v) \alpha \boldsymbol{T}(u, w)] \tag{5.6}$$

Note for the special case when $\mathcal{U}$ is a single point, then:

$$\boldsymbol{R}(v, w) = \boldsymbol{a} \circledast \boldsymbol{b}(v, w) = \boldsymbol{a}(v) \circledast \boldsymbol{b}(w) = \boldsymbol{a}(v) \alpha \boldsymbol{b}(w) \tag{5.7}$$

Definition 6: Let $\boldsymbol{R} \in \mathcal{V} \times \mathcal{W}$ be a fuzzy relation, then the fuzzy relation $\boldsymbol{R}^{-1} \in \mathcal{W} \times \mathcal{V}$, the *inverse* or *transpose* of $\boldsymbol{R}$, is defined by:

$$\boldsymbol{R}^{-1}(w, v) = \boldsymbol{R}(v, w) \tag{5.8}$$

for all $(w, v) \in \mathcal{W} \times \mathcal{V}$

The two fundamental theorems for the *greatest* solution of problem statements (2) and (4), respectively, are:

Theorem 1: Let $\boldsymbol{R} \in \mathcal{V} \times \mathcal{W}$ and $\boldsymbol{T} \in \mathcal{U} \times \mathcal{W}$ be two fuzzy relations and let $\mathcal{Q}$ be the set of fuzzy relations $\boldsymbol{Q} \in \mathcal{U} \times \mathcal{V}$ such that $\boldsymbol{Q} \circ \boldsymbol{R} = \boldsymbol{T}$, then

$$\mathcal{Q} = \{\boldsymbol{Q} \in \mathcal{U} \times \mathcal{V} \mid \boldsymbol{Q} \circ \boldsymbol{R} = \boldsymbol{T}\} \neq \varnothing \text{ iff } (\boldsymbol{R} \circledast \boldsymbol{T}^{-1})^{-1} \in \mathcal{Q} \tag{5.9}$$

If $\mathcal{Q} \neq \varnothing$, then $(\boldsymbol{R} \circledast \boldsymbol{T}^{-1})^{-1}$ is the *greatest* element in $\mathcal{Q}$.

Theorem 2: Let $Q \in \mathcal{U} \times \mathcal{V}$ and $T \in \mathcal{U} \times \mathcal{W}$ be two fuzzy relations and let $\mathcal{R}$ be the set of fuzzy relations $R \in \mathcal{V} \times \mathcal{W}$ such the $Q \circ R = T$, then

$$\mathcal{R} = \{R \in \mathcal{V} \times \mathcal{W} \mid Q \circ R = T\} \neq \varnothing \text{ iff } Q^{-1} ⓐ T \in \mathcal{R} \tag{5.10}$$

If $\mathcal{R} \neq \varnothing$, then $Q^{-1} ⓐ T$ is the *greatest* element in $\mathcal{R}$

The proofs for these theorems can be found in Sanchez [1976]. A comment made by Sanchez regarding these two theorems is that either theorem can be chosen as a unique fundamental theorem and the other can be deduced as a corollary.

Sanchez applied the results of this work to medical diagnostics. To illustrate, let S be a set of *symptoms*, $\mathcal{D}$ be a set of *diagnosis*, and $\mathcal{P}$ be a set of *patients*. Then, given $Q \in \mathcal{P} \times S$, the fuzzy relation between the patients and the symptoms, and $T \in \mathcal{P} \times \mathcal{D}$, the fuzzy relation between the same patients and the diagnosis, the problem is to determine the *medical knowledge* provided by Q and T. The information sought is $R \in S \times \mathcal{D}$, the fuzzy relation between symptoms and diagnosis, and is calculated by the formula $Q^{-1} ⓐ T$

Following these results, Sanchez [1977] then established the existence and the *greatest* solution for the simpler cases outlined by problems statements (1) and (3). For these results, Sanchez assumed that the set $\mathcal{U}$ contained only a single point, so that $Q = a \in \mathcal{V}$ and $T = b \in \mathcal{W}$ to form the relational equation $a(v) \circ R(v, w) = b(w)$. The previous results can then be applied directly. These simpler fuzzy relational equations are often called ⓐ-fuzzy relational equations.

Extension of the previous two theorems, for the existence of the *greatest* solution, to the simpler case of problem statements (1) and (3), is as follows [Sanchez, 1977]:

Theorem 3: Let $R \in \mathcal{V} \times \mathcal{W}$ and $b \in \mathcal{W}$ be two fuzzy relations and let $\mathcal{A}$ be the set of fuzzy relations $a \in \mathcal{V}$ such the $a \circ R = b$, then

$$\mathcal{A} = \{a \in \mathcal{V} \mid a \circ R = b\} \neq \varnothing \text{ iff } (R ⓐ b^{-1})^{-1} \in \mathcal{A} \tag{5.11}$$

If $\mathcal{A} \neq \varnothing$, then $(R ⓐ b^{-1})^{-1}$ is the *greatest* element in $\mathcal{A}$

Theorem 4: Let $a \in \mathcal{V}$ and $b \in \mathcal{W}$ be two fuzzy relations and let $\mathcal{R}$ be the set of fuzzy relations $R \in \mathcal{V} \times \mathcal{W}$ such the $a \circ R = b$, then

$$\mathcal{R} = \{R \in \mathcal{V} \times \mathcal{W} \mid a \circ R = b\} \neq \varnothing \text{ iff } a^{-1} ⓐ b \in \mathcal{R} \tag{5.12}$$

An interesting note is that Kiszka *et al.* [1985] undertook an experiment involving the modeling of power systems to determine $\boldsymbol{R}$ given $\boldsymbol{a}$ and $\boldsymbol{b}$ (i.e. problem statement (3)). The results of these experiments showed that the α-operator, as given by Sanchez [1976], proved to be the *best* for their experimental modeling, both in accuracy and computational efficiency.

Pedrycz [1988] has stated that the necessary and sufficient condition for the set $\mathcal{R} \neq \varnothing$ for problem statement (3) is:

$$\exists\, j \in \mathbf{J}_n \ni \boldsymbol{a}(v_j) \geq \boldsymbol{b}(w_k) \quad \forall\, k \in \mathbf{K}_p \tag{5.13}$$

Pedrycz [1988] then states that there is not a simple straightforward conditions which can assure the solvability of an entire system of equations, such as problem statement (4). This is due to the fact that a large number of complex constraints must be satisfied in order to make the system solvable. This theory is extended to the *max-product* composition in Chapter 7.

5.6 The *Minimal* Solutions

A second paper by Sanchez [1977] is significant for the theory establishing the existence of the *least elements* of the inverse solution, which is applied to the simpler case of problem statement (3). The definitions required for these results are as follows.

Definition 7: For a and $b \in [0,1]$, the σ-operator is defined as:

$$a\,\sigma\, b = \begin{cases} 0 & if \quad a < b \\ b & if \quad a \geq b \end{cases} \tag{5.14}$$

The minimization properties of this operation are noted through the following analysis:

if	$0<a<b$	then	$a\,\sigma\, b=0$
		and	$a \wedge b = a$, for $a \neq 0$
therefore			$a\,\sigma\, b \leq a \wedge b$

Definition 8: Let $\boldsymbol{a} \in \mathcal{V}$ and $\boldsymbol{b} \in \mathcal{W}$ be two fuzzy σ-sets, then the fuzzy relation $\boldsymbol{a}^{-1} ⓢ \boldsymbol{b} \in \mathcal{V} \times \mathcal{W}$ is denoted by:

$$\boldsymbol{a}^{-1} ⓢ \boldsymbol{b}(v, w) = \boldsymbol{a}(v)^{-1}\ \sigma\ \boldsymbol{b}(w) \tag{5.15}$$

for all $(v, w) \in \mathcal{V} \times \mathcal{W}$

Definition 9: Let $\boldsymbol{a} \in \mathcal{V}$ and $\boldsymbol{b} \in \mathcal{W}$ be two fuzzy sets, then the fuzzy relation $\boldsymbol{a} \circledast \boldsymbol{b} \in \mathcal{V} \times \mathcal{W}$ is denoted by:

$$\boldsymbol{a}^{-1} \circledast \boldsymbol{b}(v, w) = \boldsymbol{a}^{-1}(v)\ \alpha\ \boldsymbol{b}(w) \tag{5.16}$$

for all $(v, w) \in \mathcal{V} \times \mathcal{W}$

Definition 10: Given the fuzzy sets $\boldsymbol{a} \in \mathcal{V}$ and $\boldsymbol{b} \in \mathcal{W}$, define:

(i) $$\mathcal{R} = \{\boldsymbol{R} \in \mathcal{V} \times \mathcal{W} \mid \boldsymbol{a}(v) \circ \boldsymbol{R}(v, w) = \boldsymbol{b}(w)\} \tag{5.17}$$

(ii) $$\Gamma_k = \{v_j \in \mathcal{V} \mid \boldsymbol{a}(v_j) \geq \boldsymbol{b}(w_k)\} \text{ for all } k \in \mathbf{K}_p \tag{5.18}$$

$|\Gamma|$ denotes the *cardinality*, or number of elements of the set Γ.

A summary of the theory developed by Sanchez [1977], for problem statement (3), is best presented by Sessa [1984] in the following propositions:

Proposition 1: If $\mathcal{R} \neq \varnothing$, then $\Gamma_k \neq \varnothing$ for all $k \in \mathbf{K}_p$

Proposition 2: If $\Gamma_k \neq \varnothing$ for all $k \in \mathbf{K}_p$, then $\boldsymbol{a} \circledcirc \boldsymbol{b} \in \mathcal{R}$ and $\boldsymbol{a} \circledast \boldsymbol{b} \in \mathcal{R}$, and in particular

$$\boldsymbol{a} \circledcirc \boldsymbol{b} \subseteq \boldsymbol{R} \subseteq \boldsymbol{a} \circledast \boldsymbol{b} \tag{5.19}$$

Proposition 3: $\boldsymbol{a} \circledast \boldsymbol{b}$ is the *greatest* element of $\mathcal{R}$

Proposition 4: $\boldsymbol{a} \circledcirc \boldsymbol{b}$ is the *minimum* of $\mathcal{R}$ iff, for all $k \in \mathbf{K}_p$:

$$\text{either } |\Gamma_k| = 1, \text{ or } \quad \boldsymbol{b}(w_k) = 0 \tag{5.20}$$

Proposition 5: If $\mathcal{R} \neq \varnothing$, $\mathcal{R}$ has minimal elements $\boldsymbol{M}_q$. In order to determine these elements a non-zero element in each column of $\boldsymbol{a} \circledcirc \boldsymbol{b}$ must equal to $\boldsymbol{b}(w_k)$ for some $w_k \in \Gamma_k$.

Proof of these propositions can be found in Sanchez [1977] and Di Nola [1985]. An important corollary, to these propositions, made by Sanchez [1977], is:

Corollary 1: If $\mathcal{R} \neq \varnothing$, then the union of all *minimal* solutions of $\mathcal{R}$ is equal to $a \circledcirc b$.

The following definition from Di Nola *et al.* [1983] determines the number of permutations of the *minimal* solution union.

Definition 11: Let J be the set defined as:

$$J = \{j \in \mathbf{K}_p \text{ such that } b(w_j) = 0\}$$

Then the number of elements $M_q \in \mathcal{R}$ is determined by:

$$q = \prod_{h \in K_p - J} |\Gamma_h| \qquad (5.21)$$

An example detailing with the solution of problem statement (3) follows.

Example 1: Let $m=p=3$ and let $a \in \mathcal{V}$ and $b \in \mathcal{W}$ be defined by:

$$a = \begin{bmatrix} 0.1 & 0.0 & 0.5 \end{bmatrix} \qquad b = \begin{bmatrix} 0.5 & 0.1 & 0.5 \end{bmatrix}$$

Then $\Gamma_1 = \{v_3\}$, $\Gamma_2 = \{v_1, v_3\}$ and $\Gamma_3 = \{v_3\}$,

and $$a \circledast b = \begin{pmatrix} 1.0 & 1.0 & 1.0 \\ 1.0 & 1.0 & 1.0 \\ 1.0 & 0.1 & 1.0 \end{pmatrix}$$

$$a \circledcirc b = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.5 & 0.1 & 0.5 \end{pmatrix}$$

The number of lower solutions $q = |\Gamma_1| \cdot |\Gamma_2| \cdot |\Gamma_3| = 2$, which are:

$$M_1 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.5 & 0.0 & 0.5 \end{pmatrix} \qquad M_2 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.5 & 0.1 & 0.5 \end{pmatrix}$$

The two papers by Sanchez [1976,1977] provided the ground breaking work and created research interest in fuzzy relational equations. Since then there has been considerable contributions

Miyakoshi *et al.* [1986] provide lower solutions for both problems (3) and (4). Sessa [1984], DiNola *et al.* [1984] and Di Nola [1985] provide excellent summaries for the theory necessary for the complete resolution of problem statement (4) by showing that it is equivalent to n equations of problem statement (3).

Definition 12: Let the fuzzy relations $\boldsymbol{Q} \in \mathcal{U} \times \mathcal{V}$ and $\boldsymbol{T} \in \mathcal{U} \times \mathcal{W}$ be given and $\boldsymbol{R} \in \mathcal{V} \times \mathcal{W}$ be the unknown to be determined, as defined in problem statement (4). The *max-min* fuzzy relation is defined:

$$\boldsymbol{Q} \circ \boldsymbol{R} = \boldsymbol{T} \tag{5.22}$$

Let the fuzzy sets $\boldsymbol{Q}_i$ and $\boldsymbol{T}_i$ be defined as $\boldsymbol{Q}_i \in u_i \times \mathcal{V}$ and $\boldsymbol{T}_i \in u_i \times \mathcal{W}$, respectively, for $i \in \mathbf{I}_n$. Then the *max-min* fuzzy relation can be defined:

$$\boldsymbol{Q}_i \circ \boldsymbol{R} = \boldsymbol{T}_i \tag{5.23}$$

Let $\mathcal{R}$ be the set of solutions for problem statement (3), and, for any $i \in \mathbf{I}_n$ let $\mathcal{R}_i$ be the set of all solution for problem statement (4), as defined by equation (5.23). Then:

$$\mathcal{R} = \bigcap_{i=1}^{n} \mathcal{R}_i \tag{5.24}$$

So the study of fuzzy equation (5.22) is equivalent to the study of a fuzzy system of n equations described by equation (5.23).

The Sessa [1984] presents the following propositional summaries for the solution of problem statement (4):

Proposition 6: If $\mathcal{R}_i \neq \varnothing$, then there exists a lower solution $\boldsymbol{M}_i \leq \boldsymbol{R}$ for any $\boldsymbol{R} \in \mathcal{R}_i$ where the $\boldsymbol{M}_i$ matrices are calculated as stated in Proposition 5.

Proposition 7: If $\mathcal{R} \neq \varnothing$, the set $\Lambda_i = \{\boldsymbol{M}_i \in \mathcal{R}_i \mid \boldsymbol{M}_i \leq \boldsymbol{S}\}$ is non empty for any $i \in \mathbf{I}_n$ where $\boldsymbol{S}(v, w) = \bigwedge_u [\boldsymbol{Q}^{-1}(u, v) \; \alpha \; \boldsymbol{T}(u, w)]$.

Proposition 8: If $\mathcal{R} \neq \varnothing$, the set $\Lambda = \{\boldsymbol{M} \in \mathcal{V} \times \mathcal{W} \mid \boldsymbol{M} = \bigvee_{i=1}^{n} \boldsymbol{M}_i, \boldsymbol{M}_i \in \Lambda_i\}$ is a finite subset of $\mathcal{R}$. The minimal elements of Λ are the minimal elements of $\mathcal{R}$ and *vice versa*.

Proposition 9: If $\mathcal{R} \neq \varnothing$, for any $\boldsymbol{R} \in \mathcal{R}$ there exists an element $\boldsymbol{M} \in \Lambda$ such that $\boldsymbol{M} \leq \boldsymbol{R}$.

Proposition 10: The fuzzy relation $\Sigma = (\bigvee_{i=1}^{n} W_i) \wedge S$ belongs to $\mathcal{R}$ where $W_i = Q_i \circledcirc T_i$ for any $i \in \mathbf{I}_n$ and $R \wedge \Sigma$ lies in $\mathcal{R}$ for any $R \in \mathcal{R}$

The proof of these propositions can be found in Di Nola [1985]. Example 2 demonstrates the applications of Propositions 6 - 10 to provide the complete lower solution.

Example 2: Let $n = m = p = 3$ and $Q \in \mathcal{U} \times \mathcal{V}$ and $T \in \mathcal{U} \times \mathcal{W}$ be defined by:

$$Q = \begin{pmatrix} 0.1 & 0.0 & 0.5 \\ 0.2 & 0.7 & 0.3 \\ 0.4 & 0.9 & 1.0 \end{pmatrix} \qquad T = \begin{pmatrix} 0.5 & 0.1 & 0.5 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.5 & 0.9 \end{pmatrix}$$

Then S, W_i and Σ for $i \in \mathbf{I}_3$ are given by:

$$S = \begin{pmatrix} 1.0 & 1.0 & 1.0 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.1 & 0.9 \end{pmatrix}$$

$$W_1 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.5 & 0.1 & 0.5 \end{pmatrix} \quad W_2 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.0 & 0.0 & 0.0 \end{pmatrix} \quad W_3 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.9^{*} \\ 0.6 & 0.5^{*} & 0.9 \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.1 & 0.9 \end{pmatrix}$$

$$\therefore \quad M_1 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.5 & 0.1 & 0.5 \end{pmatrix} \quad M_2 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.0 & 0.0 & 0.0 \end{pmatrix} \quad M_3 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6^{*} \\ 0.6 & 0.1^{*} & 0.9 \end{pmatrix}$$

Note that the two entries in the W_3 matrix, marked with the asterisk, are greater than the corresponding entries in the S matrix, and so will not contribute to a *minimal* solution. These entries are adjusted with the Σ calculation.

From M_1 two lower solutions are contributed:

$$M_1^1 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.5 & 0.0 & 0.5 \end{pmatrix} \qquad M_1^2 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.5 & 0.1 & 0.5 \end{pmatrix}$$

From M_2 only one lower solutions is contributed:

$$M_2^1 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.0 & 0.0 & 0.0 \end{pmatrix}$$

From M_3 eight lower solutions are contributed:

$$M_3^1 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.0 \\ 0.0 & 0.0 & 0.9 \end{pmatrix} \qquad M_3^2 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.5 & 0.0 \\ 0.6 & 0.0 & 0.9 \end{pmatrix}$$

$$M_3^3 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.0 & 0.0 \\ 0.0 & 0.1 & 0.9 \end{pmatrix} \qquad M_3^4 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.6 & 0.1 & 0.9 \end{pmatrix}$$

$$M_3^5 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.0 & 0.0 & 0.0 \end{pmatrix} \qquad M_3^6 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.5 & 0.6 \\ 0.6 & 0.0 & 0.0 \end{pmatrix}$$

$$M_3^7 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.0 & 0.6 \\ 0.0 & 0.1 & 0.0 \end{pmatrix} \qquad M_3^8 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.6 \\ 0.6 & 0.1 & 0.0 \end{pmatrix}$$

The total number of lower solutions is equal to: $|M| = |M_1| \cdot |M_2| \cdot |M_3| = 16$

These M solutions are:

$$M_a = M_1^1 \vee M_2^1 \vee M_3^1 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.5 & 0.0 & 0.9 \end{pmatrix}$$

$$M_b = M_1^1 \vee M_2^1 \vee M_3^2 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.0 & 0.9 \end{pmatrix}$$

$$M_c = M_1^1 \vee M_2^1 \vee M_3^3 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.5 & 0.1 & 0.9 \end{pmatrix}$$

$$M_d = M_1^1 \vee M_2^1 \vee M_3^4 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.1 & 0.9 \end{pmatrix}$$

$$M_e = M_1^1 \vee M_2^1 \vee M_3^5 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.5 & 0.0 & 0.5 \end{pmatrix}$$

$$M_f = M_1^1 \vee M_2^1 \vee M_3^6 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.0 & 0.5 \end{pmatrix}$$

$$M_g = M_1^1 \vee M_2^1 \vee M_3^7 = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.5 & 0.1 & 0.5 \end{pmatrix}$$

$$M_h = M_1^{1} \vee M_2^{1} \vee M_3^{8} = \begin{pmatrix} 0.0 & 0.1 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.1 & 0.5 \end{pmatrix}$$

$$M_i = M_1^{2} \vee M_2^{1} \vee M_3^{1} = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.5 & 0.1 & 0.9 \end{pmatrix}$$

$$M_j = M_1^{2} \vee M_2^{1} \vee M_3^{2} = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.1 & 0.9 \end{pmatrix}$$

$$M_k = M_1^{2} \vee M_2^{1} \vee M_3^{3} = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.5 & 0.1 & 0.9 \end{pmatrix}$$

$$M_l = M_1^{2} \vee M_2^{1} \vee M_3^{4} = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.1 & 0.9 \end{pmatrix}$$

$$M_m = M_1^{2} \vee M_2^{1} \vee M_3^{5} = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.5 & 0.1 & 0.5 \end{pmatrix}$$

$$M_n = M_1^{2} \vee M_2^{1} \vee M_3^{6} = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.5 & 0.1 & 0.5 \end{pmatrix}$$

$$M_o = M_1^2 \vee M_2^1 \vee M_3^7 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.5 & 0.1 & 0.5 \end{pmatrix}$$

$$M_p = M_1^2 \vee M_2^1 \vee M_3^8 = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.5 & 0.6 \\ 0.6 & 0.1 & 0.5 \end{pmatrix}$$

Consider $M_a - M_h$, clearly $M_a < M_b$, $M_a < M_c$, $M_a < M_d$ and $M_e < M_f$, $M_e < M_g$, $M_e < M_h$. But $M_a < M_e$ so M_a is a *minimal* element of Λ. Because of the symmetry of the problem the same analysis is made for $M_i - M_p$, that is $M_i < M_j$, $M_i < M_k$, $M_i < M_l$ and $M_m < M_n$, $M_m < M_o$, $M_m < M_p$. And $M_i < M_m$ so M_i is also a *minimal* element of Λ. Thus there are only 2 non-redundant *minimal* solutions to this problem.

Many authors have contributed numerous algorithms to determine the *complete* family of inverse solutions to the various problem types. Prévot [1981] presented an algorithm for the solutions of fuzzy relational equations of problem statement (2). Czogala *et al.* [1982] have contributed an algorithm to determine the lower solutions for problem statement (1), however, the algorithm may also produce some non-solutions. Additionally, these authors have provided an estimate of the number of lower solutions that exist. Xu *et al.* [1982] provide a solution algorithm for problem statement (1).

In response to the work by Prévot [1981] and Czogala *et al.* [1982], Higashi *et al.* [1984] clarified and extended these works by completing the theory and generalizing the algorithms. Higashi *et al.* [1984] solved problem statement (2) using the decomposition technique outlined in Definition 12.

Pappis *et al.* [1985] have provided the background and theory for a clearly defined analytical inverse solution for problem statement (1) and provided a methodology to eliminate some, but not all, of the redundant solutions from the calculation of the lower elements. Pappis *et al.* [1991] then developed a computer algorithm based on this theory. Additionally, Pappis [1987] extended this work to a two-input, one output system, and then more generally to a multi-input, multi-output systems, or to the dimension of problem statement (2) and higher. In his most recent work Pappis *et al.* [1992] extend the computer algorithm of [1991] to the multi-input, multi-output system of problem statement (2).

The analytical inverse solution by Pappis *et al.* [1985] for problem statement (1) is given in the following theorem:

Theorem 5: Given $R \in \mathcal{V} \times \mathcal{W}$ and $b \in \mathcal{W}$, assume that there exists an $a \in \mathcal{V}$ such that $a \circ R = b$. Then:

(i) $\forall\, a \ni a \circ R = b$ then $\exists\, \phi(R\ \delta\ b) \in \Phi(R\ \delta\ b) \ni \vee(\phi(R\ \delta\ b))^t \leq a \leq \wedge(R\ \alpha\ b)^t$ (5.25)

(ii) $\forall\, a,\ \forall \phi(R\ \delta\ b) \in \Phi(R\ \delta\ b) \ni \vee(\phi(R\ \delta\ b))^t \leq a \leq \wedge(R\ \alpha\ b)^t$ then $a \circ R = b$ (5.26)

The δ operator is defined as:

Definition 13: Given an $n \times p$ matrix $R = [r_{jk}]$ and a row vector $b = (b_1, b_2, \ldots, b_p)$, the δ-operator of R with b is defined:

$$R\ \delta\ b = S \equiv [s_{jk}] \tag{5.27}$$

$$s_{ij} = \left(\bigwedge_{i=1}^{m} (r_{ii}\ \alpha\ b_i) \right) \sigma\ (r_{jk}\ \sigma\ b_k) \tag{5.28}$$

for $j \in \mathbf{J}_n$ and $k \in \mathbf{K}_p$

or
$$S = \wedge(R\ \alpha\ b)^t\ \sigma\ (R\ \sigma\ b) \tag{5.29}$$

The Φ sets are described as:

Definition 14: Given an $m \times n$ matrix $R = [r_{ij}]$, let r_j be the j th column vector of R. The set $\Phi(R)$ of the matrices $\phi(R)$ is defined as follows:

$$\Phi(R) \equiv \{\phi(R)\} \tag{5.30}$$

where
$$\phi(R) = [\phi(r_1), \phi(r_2), \ldots, \phi(r_n)]$$

$$\phi(r_j) = (\phi_1, \phi_2, \ldots, \phi_m)^t$$

$$\phi_i = 0 \text{ or } \hat{r}_j, \qquad \text{for } i \in \mathbf{I}_m$$

$$\hat{r}_j = \bigvee_i (r_i)$$

so
$$\sum_{i=1}^{m} \phi_i = \hat{r}_j$$

An example of Φ-sets follows.

Example 3: Let $\boldsymbol{R} = \begin{pmatrix} 0 & 0.5 & 0.3 \\ 0.1 & 0 & 0.3 \\ 0.1 & 0 & 0 \end{pmatrix}$ then

$$\Phi(\boldsymbol{R}) = \left\{ \begin{pmatrix} 0 & 0.5 & 0.3 \\ 0.1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0.5 & 0 \\ 0.1 & 0 & 0.3 \\ 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0.5 & 0.3 \\ 0 & 0 & 0 \\ 0.1 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0.5 & 0 \\ 0 & 0 & 0.3 \\ 0.1 & 0 & 0 \end{pmatrix} \right\}$$

Definition 15: The cardinality or number of matrices in $\Phi(\boldsymbol{R})$ is given by:

$$z = \prod_{j=1}^{n} z_j \tag{5.31}$$

where $$z_j = \begin{cases} number \quad of \quad non \quad zero \quad elements \quad in \quad r_j & if \quad \Sigma r_j \neq 0 \\ 1 & if \quad \Sigma r_j = 0 \end{cases}$$

Definition 16: Let $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ be two solution vectors of $\boldsymbol{a} \circ \boldsymbol{R} = \boldsymbol{b}$, then solution vector $\boldsymbol{a}_1$ is said to be *redundant* if $\boldsymbol{a}_1 \geq \boldsymbol{a}_2$.

Pappis and Sugeno [1985] improve their inverse solution by analytically determining and deleting redundant column vectors from the solution matrix. The basis for determining the redundant column vectors is defined as follows:

Definition 17: Let s_k and s_l be the k-th and l-th column vectors of the matrix $\boldsymbol{S} = \boldsymbol{R} \; \delta \; \boldsymbol{b}$, respectively. A column vector s_k is said to be *dominated* by a column vector s_l:

$$\text{if for } s_{il} \neq 0; \qquad s_{ik} \neq 0 \text{ and } s_{ik} \leq s_{il}, \tag{5.32}$$

$$\text{or, if for } s_{il} = 0; \qquad s_{ik} \text{ is arbitrary} \tag{5.33}$$

The improvement to the solution $\boldsymbol{S} = \boldsymbol{R} \; \delta \; \boldsymbol{b}$ is made as follows:

1. Let $\boldsymbol{S}_o$ be the matrix obtained from $\boldsymbol{S}$ by deleting all its zero column vectors. It is easy to see that:

$$\vee \phi(\boldsymbol{S})^t = \vee \phi(\boldsymbol{S}_o)^t \tag{5.34}$$

2. Let S_k be the matrix obtained from S_o be deleting a column vector s_k such that:

$$\exists\ \phi(S_k) \in \Phi(S_k) \ni \vee(\phi(S))^t \geq \vee(\phi(S_k))^t,\ \forall \phi(S) \in \Phi(S) \tag{5.35}$$

3. Let S^* be the matrix obtained from S_o be deleting all dominant column vectors. Then the improved solution of the inverse problem is given by:

$$\vee(\phi(S^*))^t \leq a \leq \wedge(R\ \alpha\ b)^t,\ \forall \phi(S^*) \in \Phi(S^*) \tag{5.36}$$

The authors state that any further reduction of the dimensions of S^* would result in some non-redundant vectors of $\vee(\Phi(S))^t$ being eliminated. The following example illustrates the improved inverse solution algorithm.

Example 4: Let $m = p = 3$ and let $R \in \mathcal{V} \times \mathcal{W}$ and $b \in \mathcal{V}$ be defined by:

$$R = \begin{pmatrix} 0.4 & 0.8 & 0.9 \\ 0.7 & 0.8 & 0.3 \\ 0.6 & 0.4 & 0.3 \end{pmatrix} \qquad b = \begin{bmatrix} 0.6 & 0.8 & 0.9 \end{bmatrix}$$

Then

$$R\ \alpha\ b = \begin{pmatrix} 1.0 & 1.0 & 1.0 \\ 0.6 & 1.0 & 1.0 \\ 1.0 & 1.0 & 1.0 \end{pmatrix}$$

$$\wedge(R\ \alpha\ b)^t = \begin{bmatrix} 1.0 & 0.6 & 1.0 \end{bmatrix}$$

$$R\ \sigma\ b = \begin{pmatrix} 0.0 & 0.8 & 0.9 \\ 0.6 & 0.8 & 0.0 \\ 0.6 & 0.0 & 0.0 \end{pmatrix}$$

$$R\ \delta\ b = \begin{pmatrix} 0.0 & 0.8 & 0.9 \\ 0.6 & 0.0 & 0.0 \\ 0.6 & 0.0 & 0.0 \end{pmatrix} = S$$

$$S^* = \begin{pmatrix} 0.0 & 0.9 \\ 0.6 & 0.0 \\ 0.6 & 0.0 \end{pmatrix}$$

From S^* the number of *minimal* solutions is: $z = 2\times1 = 2$

$$\phi_1(S^*) = \begin{pmatrix} 0.0 & 0.9 \\ 0.6 & 0.0 \\ 0.0 & 0.0 \end{pmatrix} \qquad \vee\phi_1(S^*)^t = [0.9 \quad 0.6 \quad 0.0]$$

$$\phi_2(S^*) = \begin{pmatrix} 0.0 & 0.9 \\ 0.0 & 0.0 \\ 0.6 & 0.0 \end{pmatrix} \qquad \vee\phi_2(S^*)^t = [0.9 \quad 0.0 \quad 0.6]$$

In most cases the lower solution of the fuzzy inverse problem is non-unique, the question then arises as to how many lower inverse solutions exist. Pei-Zhuang *et al.* [1984] have developed an algorithm to determine this value for problem statement (1). Guo *et al.* [1988] have given the sufficient, but not necessary, condition of the existence of a unique minimal solution to problem (1), with Sessa [1989] providing both necessary and sufficient conditions for problem (2) by decomposing it to *i* vector problems of the form of problem (1). More recently, Li [1990] has presented the necessary and sufficient condition for the existence of a single *minimal* solution for problem (1). Sessa [1984] provides this same theory for a single *minimum* for problem statement (4).

Under certain conditions, however, the *greatest* and the *least* solutions are the same and a unique inverse solution exists. Lettieri *et al.* [1984],[1985] have provided the necessary and sufficient conditions to guarantee the uniqueness of the solution for problem statement (4), while Sessa [1984] has presented similar results for both problem statements (3) and (4). Li [1990] has provided these results for problem statement (1)

There are cases when the inverse of a fuzzy relational equation does not exist analytically. Several authors {Pedrycz, [1983(b)], [1990], [1991(b)]; Ikoma *et al.*, [1993]; Valente de Oliveira [1993]} present some algorithms which permit the study of fuzzy relational equations when the solution set is empty. These algorithms will be examined in Chapter 8.

5.7 Literature Survey Summary

The theory for the resolution of the *max-t-norm* composition is virtually identical to the *max-min* composition. Di Nola *et al.* [1984] provide a extensive literary survey of the resolution of *t-norms* for the years 1976-1984, concentrating specifically on the complete resolution (*maximum* and *minimum* solutions) of problem statements (3) and (4). Di Nola *et al.* [1989] provided the theory, with examples, for the complete resolution of all the problem statements, except for the *minimum* solutions for problems (1) and (2).

According to Di Nola *et al.* [1984], Pedrycz [1982] defined problem statements (3) and (4) for the *max-t-norm* composition, and then provided the proof for the *maximum* solution of problem (3), while Di Nola *et al.* [1982] summarized the lower solutions. Pedrycz [1984(b)] provides a proof for the *maximum* solution of problem statement (3) specifically for the *max-product* operations and then in [1985] gives the proof for the *maximum* solutions of all the problem statements for the general *max-t-norm* composition. Miyakoshi *et al.* [1985] have shown that the results of Sanchez [1976] (proofs for the *maximum* solution of problem statements (2) and (4) for the *max-min* composition) can be generalized to the *max-t-norm* compositions

Pedrycz [1983(a)] provides the proof for the *maximum* and *minimum* solutions from problems (3) and (4) for the general class of fuzzy connectives defined by Yager [1980], in which both the *max-product* and *max-min* compositions are embedded. The existence condition for problem (3) has been documented by Pedrycz [1991] for the *max-t-norm* composition.

The reference summary presented in Table 5.1 summarizes the literature coverage of the proofs of the theorem presented. It is not an exhaustive coverage but indicates some of the research available.

	Max-min	*Max-t-norm*	*Max-product*
Problem 1 - Max Solution	[24],[25],[27],[44] [54]	[9],[35]	
1 - Min Solutions	[3],[11],[24],[25], [26],[41],[54]		
Problem 2 - Max Solution	[3],[14],[21],[27], [28],[42],[43]	[9],[21],[35]	
2 - Min Solutions	[3],[14],[25],[26], [39],[44]		
Problem 3 - Max Solution	[3],[6],[7],[8],[14], [30],[36],[44],[46], [47]	[7],[9],[29],[35]	[30],[34]
3 - Min Solutions	[3],[6],[7],[8],[14], [22],[30],[36],[44], [46],[47]	[5],[7],[9],[35]	[30]
Problem 4 - Max Solution	[6],[7],[9],[22],[31], [44],[47]	[8],[10],[22],[36]	[31]
4 - Min Solutions	[6],[7],[8],[22],[30], [46]	[7],[9],[35]	[30]

Table 5.1: Fuzzy Relational Equation Resolution References

5.8 Cartesian Product of Fuzzy Sets

On-line identification of a process input/output relationship is one possible application for the combination of problem (1) and (3) into a single problem statement. Pappis [1988] combines problem (1) and (3) as follows:

For $\boldsymbol{a}, \boldsymbol{a}' \in \mathcal{V}$ and $\boldsymbol{b}, \boldsymbol{b}' \in \mathcal{W}$, given $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{b}'$ find all $\boldsymbol{a}'$ such that $\boldsymbol{a}' \circ (\boldsymbol{a} \times \boldsymbol{b}) = \boldsymbol{b}'$

The condition placed on this work is that $\boldsymbol{a}$ and $\boldsymbol{b}$ must be *normal*, with *normal* is defined as:

Definition 18: Denote $\boldsymbol{a} = \{a_1, a_2, \dots, a_n\}$ and $\boldsymbol{b} = \{b_1, b_2, \dots, b_p\}$, if $\max(a_j) = \hat{\boldsymbol{a}} = 1$ and $\max(b_k) = \hat{\boldsymbol{b}} = 1$, for $j \in \mathbf{J}_n$ and $k \in \mathbf{K}_p$, respectively, then $\boldsymbol{a}$ and $\boldsymbol{b}$ are said to be *normal.*

Along with the solution for this combined problem statement, Pappis [1988] provides the necessary and sufficient conditions for the existence of the solution.

Kim *et al.* [1991] extend this work for the more usual case when $\boldsymbol{a}$ and $\boldsymbol{b}$ may not be normal. The necessary and sufficient conditions for the existence of a solution for this case are:

Theorem 6: Let $\boldsymbol{a} = \{a_1, a_2, \dots, a_n\} \in \mathcal{V}$, $\boldsymbol{b} = \{b_1, b_2, \dots, b_k\} \in \mathcal{W}$ and $\boldsymbol{b}' = \{b'_1, b'_2, \dots, b'_k\} \in \mathcal{W}$. Then there exists an $\boldsymbol{a}' = \{a'_1, a'_2, \dots, a'_n\} \in \mathcal{V}$ such that:

$$\boldsymbol{a}' \circ (\boldsymbol{a} \times \boldsymbol{b}) = \boldsymbol{b}' \tag{5.37}$$

iff (i) $b_k \geq b_k'$, for all $k \in \mathbf{K}_p$

(ii) if $b_k > b'_k$, then $b'_k = (\boldsymbol{a}' \circ \boldsymbol{a}^t)$,
else $b_k = b'_k$, for all $k \in \mathbf{K}_p$

(iii) $b'_k \leq (\boldsymbol{a}' \circ \boldsymbol{a}^t)$ for all $k \in \mathbf{K}_p$

(iv) there exists a $t \in \mathbf{J}_n$ such that:

$a_k = \hat{\boldsymbol{b}}'$ and $a'_k \geq \hat{\boldsymbol{b}}'$

or $a_k \geq \hat{\boldsymbol{b}}'$ and $a'_k = \hat{\boldsymbol{b}}'$

The combined problem is formulated in the following manner:

$$\boldsymbol{a}'\circ(\boldsymbol{a}\times\boldsymbol{b}) = \boldsymbol{b}' \quad \Rightarrow \quad b_k' = \bigvee_j (a_j'\wedge(a_j\wedge b_k)) \tag{5.38}$$

$$b_k' = (\bigvee_j (a_j'\wedge a_j))\wedge b_k \tag{5.39}$$

$$\therefore \quad b_k' = (\boldsymbol{a}'\circ\boldsymbol{a}^t)\wedge b_k \tag{5.40}$$

Taking the maximum of both sides results in:

$$\Rightarrow \quad \bigvee_k (b_k') = \bigvee_k ((\boldsymbol{a}'\circ\boldsymbol{a}^t)\wedge b_j) \tag{5.41}$$

$$(\bigvee_k b_k') = (\boldsymbol{a}'\circ\boldsymbol{a}^t)\wedge(\bigvee_k b_k) \tag{5.42}$$

$$\therefore \quad \hat{\boldsymbol{b}}' = (\boldsymbol{a}'\circ\boldsymbol{a}^t)\wedge\hat{\boldsymbol{b}} \tag{5.43}$$

The condition that $\boldsymbol{b}$ is *normal* (i.e. $\hat{\boldsymbol{b}} = 1$) reduces equation (5.43) to:

$$\hat{\boldsymbol{b}}' = \boldsymbol{a}'\circ\boldsymbol{a}^t \tag{5.44}$$

However, equation (5.43) also reduces to equation (5.44) when $\hat{\boldsymbol{b}} > \hat{\boldsymbol{b}}'$, as specified above in Theorem 6(*i*).

Pappis *et al.* [1985] show that the solution of equation (5.44) is given by:

$$(\phi(\boldsymbol{a}^t\ \sigma\ \hat{\boldsymbol{b}}'))^t \leq \boldsymbol{a}' \leq (\boldsymbol{a}^t\ \alpha\ \hat{\boldsymbol{b}}')^t \tag{5.45}$$

where $\phi(\boldsymbol{a}^t\ \sigma\ \hat{\boldsymbol{b}}')\in\Phi(\boldsymbol{a}^t\ \sigma\ \hat{\boldsymbol{b}}')$.

The development of the solution for the *non-normal* $\boldsymbol{b}$ is the same as for the *normal* $\boldsymbol{b}$ up to equation (5.43).

$$\hat{\boldsymbol{b}}' = (\boldsymbol{a}'\circ\boldsymbol{a}^t)\wedge\hat{\boldsymbol{b}} \tag{5.43}$$

From this point two solution possibi[illegible]ies exist:

$$(i) \quad \hat{\boldsymbol{b}}' < \hat{\boldsymbol{b}} \tag{5.46}$$

$$(ii) \quad \hat{\boldsymbol{b}}' = \hat{\boldsymbol{b}} \tag{5.47}$$

Note that the possibility $\hat{\boldsymbol{b}}' > \hat{\boldsymbol{b}}$ does not exist due to necessary and sufficient condition of Theorem 6(*i*).

The condition $\hat{\boldsymbol{b}}' < \hat{\boldsymbol{b}} \Rightarrow \hat{\boldsymbol{b}}' = (\boldsymbol{a}' \circ \boldsymbol{a}^t)$ from the necessary and sufficient condition of Theorem 6(*i*). This situation can also be solved by the method presented by Pappis [1988].

The condition $\hat{\boldsymbol{b}}' = \hat{\boldsymbol{b}} \Rightarrow \hat{\boldsymbol{b}}' \leq (\boldsymbol{a}' \circ \boldsymbol{a}^t)$ has been partially solved by Kim et.al [1991]. The solution provided by Kim *et al.* [1991] correctly identifies the lower solution of $\boldsymbol{a}$ under both solution conditions, but the upper solution of $\boldsymbol{a}'$ is not correct for either condition.

The solution for the case when $\hat{\boldsymbol{b}}' = \hat{\boldsymbol{b}}$ is addressed by Bourke *et al.* [1994]. From equation (5.43), the condition is that $\hat{\boldsymbol{b}}' = \hat{\boldsymbol{b}} \Rightarrow \hat{\boldsymbol{b}}' \leq (\boldsymbol{a}' \circ \boldsymbol{a}^t)$. So the solution for $\boldsymbol{a}$ must be such that:

$$\hat{\boldsymbol{b}}' \leq (\boldsymbol{a}' \circ \boldsymbol{a}^t) \leq 1 \tag{5.48}$$

For this situation to be valid the necessary and sufficient condition of Theorem 6(*iv*) must be modified as follows:

(*iv*) (a) if $\hat{\boldsymbol{b}}' \neq \hat{\boldsymbol{b}}$

there exist a $t \in \mathbf{J}_n$ such that:

$a_t = \hat{\boldsymbol{b}}'$ and $a'_t \geq \hat{\boldsymbol{b}}'$

or $a_t \geq \hat{\boldsymbol{b}}'$ and $a'_t = \hat{\boldsymbol{b}}'$

(b) if $\hat{\boldsymbol{b}}' = \hat{\boldsymbol{b}}$

there exist a $t \in \mathbf{J}_n$ such that:

$a_t \geq \hat{\boldsymbol{b}}'$ and $a'_t \geq \hat{\boldsymbol{b}}'$

So for the case when $\hat{\boldsymbol{b}}' = \hat{\boldsymbol{b}}$, if the necessary and sufficient condition of Theorem 6(*iv*)(b) is satisfied then the *largest* $\boldsymbol{a}$ which satisfies $\hat{\boldsymbol{b}}' \leq (\boldsymbol{a}' \circ \boldsymbol{a}^t)$ is the vector $\boldsymbol{a} = [1, 1, \ldots, 1]_{1 \times m}$

The complete general solution for the combined problem can now be given as:

$$\Phi(\boldsymbol{a}\ \beta\ \hat{\boldsymbol{b}}') \leq \boldsymbol{a}' \leq (\boldsymbol{a}\ \hat{\alpha}\ \hat{\boldsymbol{b}}') \qquad (5.49)$$

where
$$a_i\ \beta\ \hat{\boldsymbol{b}}' = \begin{cases} 0 & if \quad a_i < \hat{\boldsymbol{b}}' \\ \hat{\boldsymbol{b}}' & if \quad a_i \geq \hat{\boldsymbol{b}}' \end{cases} \qquad (5.50)$$

and
$$a_i\ \hat{\alpha}\ \hat{\boldsymbol{b}}' = \begin{cases} 1 & if \quad \hat{\boldsymbol{b}}' = \hat{\boldsymbol{b}} \\ 1 & if \quad \hat{\boldsymbol{b}}' \neq \hat{\boldsymbol{b}} \quad and \quad a_i \leq \hat{\boldsymbol{b}}' \\ \hat{\boldsymbol{b}}' & if \quad \hat{\boldsymbol{b}}' \neq \hat{\boldsymbol{b}} \quad and \quad a_i > \hat{\boldsymbol{b}}' \end{cases} \qquad (5.51)$$

The concept of the Cartesian Product of Fuzzy Sets can be extended to the *max-product* operator as follows:

For $\boldsymbol{a}, \boldsymbol{a}' \in$ V and $\boldsymbol{b}, \boldsymbol{b}' \in$ W, given $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{b}'$ find all $\boldsymbol{a}'$ such that $\boldsymbol{a}' \circ (\boldsymbol{a} \times \boldsymbol{b}) = \boldsymbol{b}'$

Since the objective of this work is to provide a complete solution, there will be no conditions place on $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{b}'$. The necessary and sufficient conditions for the existence of a solution for this case are:

Theorem 7: Let $\boldsymbol{a} = \{a_1, a_2, \ldots, a_n\} \in \mathcal{V}$, $\boldsymbol{b} = \{b_1, b_2, \ldots, b_k\} \in \mathcal{W}$ and $\boldsymbol{b}' = \{b'_1, b'_2, \ldots, b'_k\} \in \mathcal{W}$. Then there exists an $\boldsymbol{a}' = \{a'_1, a'_2, \ldots, a'_n\} \in \mathcal{V}$ such that:

$$\boldsymbol{a}' \circ (\boldsymbol{a} \times \boldsymbol{b}) = \boldsymbol{b}' \qquad (5.52)$$

iff $\qquad b_k \geq b_k'$, for all $k \in \mathbf{K}_p$

Proof: Formulate the problem as follows:

$$\boldsymbol{a}' \circ (\boldsymbol{a} \times \boldsymbol{b}) = \boldsymbol{b}' \quad \Rightarrow \quad b_k' = \bigvee_j (a_j' \cdot (a_j \cdot b_k)) \qquad (5.53)$$

$$b_k' = (\bigvee_j (a_j' \cdot a_j)) \cdot b_k \qquad (5.54)$$

$$\therefore \quad b_k' = (\boldsymbol{a}' \circ \boldsymbol{a}^t) \cdot b_k \qquad (5.55)$$

Let $(a' \circ a^t) = \alpha$, such that $0 \le \alpha \le 1$ since both a' and a are bounded by [0, 1].

$$\therefore \qquad b_k' = \alpha \cdot b_k \qquad \text{for all } k \in \mathbf{K}_p \tag{5.56}$$

$$\text{and} \qquad b_k' \le b_k \tag{5.57}$$

Corollary 2: Since $(a' \circ a^t) = \alpha$ (a constant), then $b' = \alpha \cdot b$ and $\alpha = b_k'/b_k$ for any k. (Note: from Theorem 7, $b_k \ge b_k'$, for all $k \in \mathbf{K}_p$)

The next step is to determine the a' based on the values of α. The solution possibilities are as follows:

(1) If $b_k' = b_k$ then $\alpha = (a' \circ a^t) = 1.0$

and $\bigvee_j (a_j' \cdot a_j) = 1.0$

iff a' and a are both normal and $PP(a) \subseteq PP(a')$, where PP stands for *peak pattern* [Tong, 1978]

Proof: In order for $\bigvee_j (a_j' \cdot a_j) = 1$ there must be at least one entry 1 in each of a' and a and this entry must occur at the same j in each fuzzy set. So the *peak pattern* of a must be contained in the *peak pattern* of a'.

(2) If $b_k' = 0$ then $\alpha = (a' \circ a^t) = 0$

and $\bigvee_j (a_j' \cdot a_j) = 0$

iff supp(a') $\neq$ supp(a)

Proof: If supp(a') $\neq$ supp(a) then for each j=1, ..., n either

(*i*) $a_j' = 0$

(*ii*) $a_j = 0$

or (*iii*) $a_j' = a_j = 0$

so $\bigvee_j (a_j' \cdot a_j) = 0$

(3) If $0 < b_k' < b_k < 1$ then $0 < \alpha = (\boldsymbol{a}' \circ \boldsymbol{a}^t) < 1.0$

or $0 < \bigvee_j (a_j' \cdot a_j) < 1.0$

iff supp($\boldsymbol{a}'$) overlaps with supp($\boldsymbol{a}$) and

if $\boldsymbol{a}'$ and $\boldsymbol{a}$ are *normal* then the *peak pattern* must be

different.

Proof: For a result > 0 the support of $\boldsymbol{a}'$ and $\boldsymbol{a}$ must overlap. In order that the result $\neq 1$, the *peak pattern* must be different if both are *normal*.

The solution algorithms for each of the cases presented above are as follows:

(1) If $b_k' = b_k$ then $\alpha = 1.0$

(*i*) $\bigcup \boldsymbol{a}'_{\min}$: $a_j' = \begin{cases} 0 & \text{if} \quad a_j \neq 1 \\ 1 & \text{if} \quad a_j = 1 \end{cases}$

(*ii*) $\boldsymbol{a}'_{\max}$: $a_j' = 1$ for all j

(2) If $b_k' = 0$ then $\alpha = 0$

(*i*) $\boldsymbol{a}'_{\min}$: $a_j' = 0$ for all j

(*ii*) $\boldsymbol{a}'_{\max}$: $a_j' = \begin{cases} 0 & \text{if} \quad a_j > 0 \\ 1 & \text{if} \quad a_j = 0 \end{cases}$

(3) If $0 < b_k' < b_k < 1$ then $0 < \alpha < 1.0$

(*i*) $\bigcup \boldsymbol{a}'_{\min}$: $a_j' = \begin{cases} 0 & \text{if} \quad a_j = 0 \quad \text{or} \quad \alpha / a_j > 1 \\ \alpha / a_j & \text{otherwise} \end{cases}$

(*ii*) $\boldsymbol{a}'_{\max}$: $a_j' = \begin{cases} 1 & \text{if} \quad a_j = 0 \quad \text{or} \quad \alpha / a_j > 1 \\ \alpha / a_j & \text{otherwise} \end{cases}$

Fuzzy Cartesian Product calculations have by *completed* for the *max-min* composition and then *extended* to the *max-product* composition. As with the inverse calculations, for application of these results the Cartesian Product calculations must also be exact.

5.9 Summary

Chapter 5 begins with a literature review that consolidates the solutions to the various the inverse problems presented in the literature.

(1) "Given the fuzzy relations $\boldsymbol{R}$ and $\boldsymbol{b}$,
find all fuzzy sets $\boldsymbol{a}$ such that $\boldsymbol{a} \circ \boldsymbol{R} = \boldsymbol{b}$".

(2) "Given the fuzzy relations $\boldsymbol{R}$ and $\boldsymbol{T}$,
find all fuzzy relations $\boldsymbol{Q}$ such that $\boldsymbol{Q} \circ \boldsymbol{R} = \boldsymbol{T}$".

(3) "Given the fuzzy sets $\boldsymbol{a}$ and $\boldsymbol{b}$,
find the fuzzy relation $\boldsymbol{R}$ such that $\boldsymbol{a} \circ \boldsymbol{R} = \boldsymbol{b}$".

(4) "Given the fuzzy relations $\boldsymbol{Q}$ and $\boldsymbol{T}$,
find the fuzzy relation $\boldsymbol{R}$ such that $\boldsymbol{Q} \circ \boldsymbol{R} = \boldsymbol{T}$".

Problem statements (1) and (2) represent the search for a fuzzy cause, while statements (3) and (4) represent fuzzy identification. All four problem statements are valid when $\circ$ represents either *max-min* or *max-product* composition.

A complete program, written in MATLAB®, to determine fuzzy inverses using either the *max-min* or the *max-product* composition for all the problem statements is available and a listing is provided in Appendix 1.

A table is provided in Chapter 5 which details the literature source for all the problem definitions for *max-min*, *max-product* and *max-t-norm*. Additionally, a complete solution guide for these problems for the *max-product* composition is provided in Appendix 2. The consolidated literature review for fuzzy inverse problems, available in this Chapter, provides an important reference itself to the abundant material available in the subject.

Also available in Chapter 5 is the complete solution of cartesian product of fuzzy Sets for the *max-min* composition, published in *Fuzzy Sets and Systems*, [Bourke *et al.*, 1994]. This paper completes the work of two other papers, by different authors, and has potential for control systems from a on-line identification perspective. The complete solution theory of cartesian product is extended to the *max-product* composition for control applications under this composition.

REFERENCES

[1] Bellman, R.E., Zadeh, L.A., "Local and Fuzzy Logic", In: *Modern Uses of Multiple-Valued Logic*, Dunn, J.M., Epstein, G., (Editors), D. Reidel Publishing Company, Dordrecht-Holland, pp. 103-165, 1977.

[2] Bourke, M.M., Fisher, D.G., "The Complete Resolution of Cartesian Products of Fuzzy Sets", *Fuzzy Sets and Systems*, Vol. 63, pp. 111-115, 1994.

[3] Czogala, E., Drewniak, J., Pedrycz, W., "Fuzzy Relation Equations on a Finite Set", *Fuzzy Sets and Systems*, Vol. 7, pp. 89-101, 1982.

[4] Czogala, E., Pedrycz, W., "Fuzzy Rules Generation for Fuzzy Control", *Cybernetics and Systems: An International Journal*, Vol 13, pp. 275-293, 1982.

[5] Di Nola, A., Pedrycz, W., Sessa, S., "On Solutions of Fuzzy Relational Equations and Their Characterization", *BUSEFAL*, Vol. 12, pp. 60-71, 1982.

[6] Di Nola, A., Sessa, S., "On the Fuzziness of Solutions of @-Fuzzy Relation Equations on Finite Spaces", *Fuzzy Sets and Systems*, Vol. 11, pp. 65-77, 1983.

[7] Di Nola, A., Pedrycz, W., Sessa, S., Pei-Zhuang, W., "Fuzzy Relation Equations Under a Class of Triangular Norms: A Survey and New Results", *Stochastica*, Vol. 8, pp. 99-145, 1984.

[8] Di Nola, A., "Relational Equations in Totally Ordered Lattices and Their Complete Resolution", *Journal of Mathematical Analysis and Applications*, Vol. 107, pp. 148-155, 1985.

[9] Di Nola, A., Sessa, S., Pedrycz, W., Sanchez, E., *Fuzzy Relation Equations and their Applications to Knowledge Engineering*, Kluwer Academic Publishers, The Netherlands, 1989.

[10] Dubois, D., Prade, H., "New Results about Properties and Semantics of Fuzzy Set-Theoretic Operators", In: Wang, P.P., Chang, S.K., (Editors), *Fuzzy Sets*, Plenum Press, New York, pp. 59-75, 1986.

[11] Guo, S.Z., Wang, P.Z., Di Nola, A., Sessa, S., "Further Contributions to the Study of Finite Fuzzy Relation Equations", *Fuzzy Sets and Systems*, Vol. 26, pp. 93-104, 1988.

[12] Gupta, M.M., Qi, J., "Theory of T-Norms and Fuzzy Inference Methods", *Fuzzy Sets and Systems*, Vol. 40, pp. 431-451, 1991(a).

[13] Gupta, M.M., Qi, J., "Design of Fuzzy Logic Controllers Based on Generalized T-Operators", *Fuzzy Sets and Systems*, Vol. 40, pp. 473-489, 1991(b).

[14] Higashi, M., Klir, G.J., "Resolution of Finite Fuzzy Relation Equations", *Fuzzy Sets and Systems*, Vol. 13, pp. 65-82, 1984.

[15] Ikoma, N., Pedrycz, W., Hirota, K., "Estimation of Fuzzy Relational Matrix by using Probabilistic Descent Method", *Fuzzy Sets and Systems*, Vol. 57, pp. 335-349, 1993.

[16] Kiszka, J.B., Kochanska, M.E., Slewinska, D.S., "The Influence of Some Fuzzy Implication Operators on the Accuracy of a Fuzzy Model", *Fuzzy Sets and Systems*, Part I: Vol. 15, pp. 111-128, Part II: Vol. 15, pp. 223-240, 1985.

[17] Kim, J.B., Kim, Y.H., Kim, C.B., "Resolution of the Cartesian Product of Fuzzy Sets", *Fuzzy Sets and Systems*, Vol. 41, pp. 293-297, 1991.

[18] Lettieri, A., Liguori, F., "Characterization of Some Fuzzy Relation Equations Provided with One Solution on a Finite Set", *Fuzzy Sets and Systems*, Vol. 13, pp 83-94, 1984.

[19] Lettieri, A., Liguori, F., "Some Results on Fuzzy Relation Equations Provided with One Solution", *Fuzzy Sets and Systems*, Vol. 17, pp. 199-209, 1985.

[20] Li, J.-X., "The Smallest Solution of Max-Min Fuzzy Equations", *Fuzzy Sets and Systems*, Vol. 41, pp. 317-327, 1990.

[21] Miyakoshi, M., Shimbo, M., "Solutions of Composite Fuzzy Relational Equations with Triangular Norms", *Fuzzy Sets and Systems*, Vol. 16, pp. 53-63, 1985.

[22] Miyakoshi, M., Shimbo, M., "Lower Solutions of Systems of Fuzzy Equations", *Fuzzy Sets and Systems*, Vol. 19, pp. 37-46, 1986.

[23] Oden, G.C., "Integration of Fuzzy Logical Information", *Journal of Experimental Psychology (Human Perception and Performance)*, Vol. 106, pp. 565-575, 1977.

[24] Pappis, C.P., Sugeno, M., "Fuzzy Relational Equations and the Inverse Problem", *Fuzzy Sets and Systems*, Vol. 15, pp. 79-90, 1985.

[25] Pappis, C.P., "Multi-Input Multi-Output Fuzzy Systems and the Inverse Problem", *European Journal of Operational Research*, Vol. 28, pp. 228-230, 1987.

[26] Pappis, C.P., "Resolution of Cartesian Products of Fuzzy Sets", *Fuzzy Sets and Systems*, Vol. 26, pp. 387-391, 1988.

[27] Pappis, C.P., Adamopoulos, G.I., "A Computer Algorithm for the Solution of the Inverse Problem of Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 39, pp. 279-290, 1991.

[28] Pappis, C.P., Adamopoulos, G.I., "A Software Routine to Solve the Generalized Inverse Problem of Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 47, pp. 319-322, 1992.

[29] Pedrycz, W., "Fuzzy Relational Equations with Triangular Norms and their Resolution", *BUSEFAL*, Vol. 11, pp. 24-32, 1982.

[30] Pedrycz, W., "Fuzzy Relational Equations with Generalized Connectives and Their Applications", *Fuzzy Sets and Systems*, Vol. 10, pp. 185-201, 1983(a).

[31] Pedrycz, W., "Numerical and Aplicational Aspects of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 11, pp. 1-18, 1983(b).

[32] Pedrycz, W., "Some Applicational Aspects of Fuzzy Relational Equations in Systems Analysis", *International Journal of General Systems*, Vol. 9, pp. 125-132, 1983(c).

[33] Pedrycz. W., "An Identification Algorithm in Fuzzy Relation Systems", *Fuzzy Sets and Systems*, Vol. 13, pp. 153-167, 1984(a).

[34] Pedrycz, W., "Identification in Fuzzy Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 14, pp. 361-366, 1984(b).

[35] Pedrycz, W., "On Generalized Fuzzy Relational Equations and Their Applications", *Journal of Mathematical Analysis and Applications*, Vol. 107, pp. 520-536, 1985.

[36] Pedrycz, W., "Approximate Solutions of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 28, pp. 183-202, 1988.

[37] Pedrycz, W., *Fuzzy Control and Fuzzy Systems*, John Wiley & Sons, Inc., New York, N.Y., 1989.

[38] Pedrycz, W., "Inverse Problem in Fuzzy Relational Equations", *Fuzzy Sets and Systems"*, Vol. 36, pp. [illegible]-291, 1990.

[39] Pedrycz, W., "Processing in Relational Structures: Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 40, pp. 77-106, 1991(a).

[40] Pedrycz, W., "Neurocomputations in Relational Systems", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, pp. 289-297, 1991(b).

[41] Pei-Zhuang, W., Sessa, S., Di Nola, A., Pedrycz, W., "How Many Lower Solutions Does a Fuzzy Relation Equation Have? , *BUSEFAL*, Vol. 18, pp. 67-74, 1984.

[42] Prévot, M., "Algorithm for the Solution of Fuzzy Relations", *Fuzzy Sets and Systems*, Vol. 5, pp. 319-322, 1981.

[43] Sanchez, E., "Resolution of Composite Fuzzy Relation Equations", *Information and Control*, Vol. 30, pp. 38-49, 1976.

[44] Sanchez, E., "Solutions in Composite Fuzzy Relation Equations: Applications to Medical Diagnosis in Brouwerian Logic", In: *Fuzzy Automata and Decision Processes*, Gupta, M.M., Saridis, G.N., Ragade, R.K., North-Holland, Amsterdam, pp. 221-234, 1977.

[45] Sanchez, E., "Resolution of Eigen Fuzzy Sets Equations", *Fuzzy Sets and Systems*, Vol. 1, pp. 69-74, 1978.

[46] Sessa, S., "Some Results in the Setting of Fuzzy Relation Equations Theory", *Fuzzy Se and Systems*, Vol. 14, pp. 281-297, 1984.

[47] Sessa, S., "Finite Fuzzy Relation Equations with Unique Solution in Complete Brouwe Lattices", *Fuzzy Sets and Systems*, Vol. 29, pp. 103-113, 1989.

[48] Thole, U., Zimmermann, H.-J., Zysno, P., On the Suitability of Minimum and Product Operators for Intersection of Fuzzy Sets", *Fuzzy Sets and Systems*, Vol. 2, pp. 167-18(1979.

[49] Tong, R.M., "Analysis and Control of Fuzzy Systems using Finite Discrete Relations", *International Journal of Control*, Vol. 27, pp. 431-440, 1978.

[50] Valente de Oliveira, J., "Neuron Inspired Learning Rules for Fuzzy Relational Structu *Fuzzy Sets and Systems*, Vol. 57, pp. 41-53, 1993.

[51] Vrba, J., "Peak-Pattern Concept and Max-Min Inverse Problem in Fuzzy Control Mod *Fuzzy Sets and Systems*, Vol. 47, pp. 1-11, 1992.

[52] Yager, R.R., On A General Class of Fuzzy Connectives", *Fuzzy Sets and Systems*, Vo pp. 235-242, 1980.

[53] Yager, R.R., "Some Procedures for Selecting Fuzzy Set-Theoretic Operators", *Interna Journal of General Systems*, Vol. 8, pp. 115-124, 1982.

[54] Xu, W.-L., Wu, C.-F., Cheng, W.-M., "An Algorithm to Solve the Max-Min Composi Fuzzy Relational Equations", In: *Approximate Reasoning in Decision Analysis*, Gupt M.M., and Sanchez, E., (Editors), North-Holland Publishing Co., Amsterdam, pp. 47- 1982.

[55] Zadeh, L.A., "Fuzzy Sets", *Information and Control*, Vol. 8, pp. 338-353, 1965.

[56] Zimmermann, H.-J., Zysno, P., "Latent Connectives in Human Decision-Making", *Fu Sets and Systems*, Vol.. 4, pp. 37-51, 1980.

CHAPTER 6 *FUZZY CONTROL* versus CLASSICAL CONTROL [1]

Since fuzzy systems are, by their very nature, somewhat imprecise, it is not unreasonable to ask for fuzzy solutions to relational equations.

[Tong, 1976]

6.1 Introduction

The most straightforward applications of fuzzy sets occur when there is direct agreement between the fuzziness and the systems structure. In general, however, fuzzy control applications tend to artificially fuzzify and defuzzify deterministic data in order to handle complex non-deterministic systems. It has become evident, through the abundant commercial applications developed in Japan, that the most successful implementations of fuzzy controllers are simple *look-up* tables [Kosko, 1992]. This does not mean that fuzzy controllers are only *look-up* tables. Unfortunately this narrow thinking has extremely biased the views of some researchers who feel that extensive mathematical formalism is lacking in these controller developments [Pedrycz, 1991(b)].

This Chapter reviews the most widely demonstrated fuzzy controller, the *Fuzzy PI controller*, and relates this design, as well as, the design analysis (i.e. stability, controllability, etc.) to conventional classical control.

[1] A version of this chapter has been accepted for publication. Bourke M.M., Fisher D.G., 1995. (*Fuzzy Sets and Systems*).

6.2 Conventional PID Control

Conventional PID controllers employ three basic feedback control modes; Proportional (P), integral (I) and derivative (D). For this basic control system, shown in Figure 6.1, the controller compares the measured value to the set point and takes the appropriate corrective action.

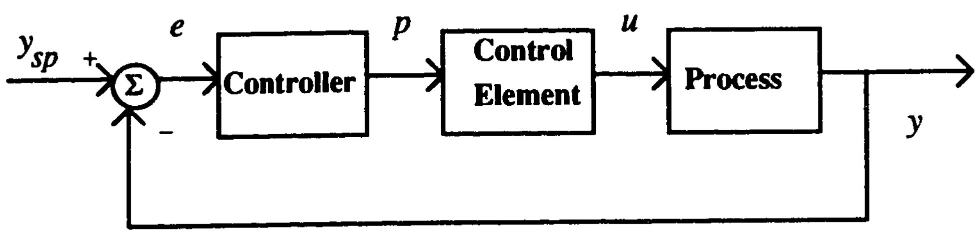


Figure 6.1: Basic Feedback Control System

The original form of the PID controller for systems that are continuous functions of time is:

$$p(t) = \bar{p} + K_c\left[e(t) + \frac{1}{\tau_I}\int_0^t e(t)dt + \tau_D \frac{de(t)}{dt}\right] \tag{6.1}$$

where
- $p(t)$ is the controller output at time t
- $\bar{p}$ is the controller bias
- $e(t)$ is the error $y_{sp}(t)$ - $y(t)$
- K_c is the controller gain
- τ_I is the integral time or reset time
- τ_D is the derivative time

However, with widespread application of digital control systems, the form of the PID controller changed with the introduction of digital control techniques. Basically, the changes involved replacing the integral and derivative terms by their discrete equivalents, a summation and a first order backward difference, respectively. The digital form of the PID controller is:

$$p_n = \bar{p} + K_c\left[e_n + \frac{\Delta t}{\tau_I}\sum_{k=1}^{n} e_k + \frac{\tau_D}{\Delta t}(e_n - e_{n-1})\right] \tag{6.2}$$

where
- Δt is the sampling period
- p_n is the controller output at the nth sampling instant, $n = 1, 2, \ldots$
- e_n is the error at the nth sampling instant, $y_{sp}(n)$ - $y(n)$

Equation (6.2) is known as the *positional form* of the PID control algorithm because it calculates the actual output. An alternate approach is the *velocity form* of the algorithm which calculates the *change* in the controller output. The *velocity form* is:

$$\Delta p = p_n - p_{n-1} = K_c\left[(e_n - e_{n-1}) + \frac{\Delta t}{\tau_I} e_n + \frac{\tau_D}{\Delta t}(e_n - 2e_{n-1} - e_{n-2})\right] \tag{6.3}$$

or

$$p_n = p_{n-1} + K_c\left[(e_n - e_{n-1}) + \frac{\Delta t}{\tau_I} e_n + \frac{\tau_D}{\Delta t}(e_n - 2e_{n-1} - e_{n-2})\right] \tag{6.4}$$

Equations (6.3) and/or (6.4) are the algorithmic basis for the *fuzzy* PID control algorithms which are discussed in the next section.

6.3 Fuzzy PID Control

The *fuzzy PI controller* is the control methodology encountered most often in the literature for fuzzy rule-based controller design. Based on the literature search for this thesis, there are few *fuzzy relational-based* controller designs [Graham *et al.*, 1988], although some self-organizing or on-line learning controller designs involve relational matrix learning techniques [Qiao, *et al.* 1992, Song *et al.* 1993, Shao, 1988]. The most comprehensive paper related to overall rule-based *fuzzy logic controller* development, including *fuzzy PID*, is a 2 part publication from Lee [1990], which references a extensive bibliography. This section provides an overview of *fuzzy PI* and *PID* controller designs and a brief survey of research encountered in literature which compliments the paper by Lee [1990].

The first and most noted fuzzy controller design was by Mamdani [1974] for the control of a highly interactive model steam engine. The control of the speed and the pressure of the engine was obtained by manipulating heat addition and throttle position. The controller was calculated based on the error of each of the pressure and speed and the change of error of each. The results of this MIMO controller, consisting of 4 input and 2 outputs, clearly demonstrated the ability of *fuzzy logic* in the control application area.

The design of the *fuzzy PID controller* is very similar to the conventional PID controller. As with the conventional PID controller, various combinations of proportional (P), integral (I) or derivative (D) may be chosen for the final implementation in order to meet the control objective. The format of the rule-based design is as follows:

$$\textit{If error}_k \textit{ is } e \textit{ and change of error}_k \textit{ is } \Delta e$$
$$\textit{and rate of change of error}_k \textit{ is } \Delta^2 e$$
$$\textit{then change of input is } \Delta u$$

where $input_k = u(k) - u(k-1)$

$error_k = y_{sp}(k) - y(k)$

$change\ of\ error_k = error_k - error_{k-1}$

$rate\ of\ change\ of\ error_k = change\ of\ error_k - change\ of\ error_{k-1}$

and Δu, e, Δe, and $\Delta^2 e$ are linguistic descriptions such a ***big, small***, etc. with quantifiers ***positive*** (P), ***zero*** (Z) and ***negative*** (N)

More commonly, however, it is the ***error*** and ***change of error*** which are used in the formulation of the *fuzzy PI controller*. Implementation is usually in the form of a *look-up* table, as shown in Figure 6.2 and discussed in Chapter 2.

Δu is	Δe is N	Δe is Z	Δe is P
e is N	N	N	Z
e is Z	N	Z	P
e is P	Z	P	P

Figure 6.2: Look-up Table
(N = Negative; Z = Zero; P = Positive)

Peng *et al.* [1987] have published a 2 part paper on the design and development of fuzzy rule-based PI, PD and PID controllers. In these papers, the rule-based fuzzy controller is established and remains constant through the test. The controller is then *tuned* through the use of scaling factors on the values of Δu, e, Δe, and $\Delta^2 e$. Tests by these authors [Peng *et al.*, 1987] to determine the *best* implication operator for the learning of R, for 1st and 2nd order fuzzy systems, showed that the *max-product* operator was superior when control was initiated with their rule-base. A more recent paper [Peng *et al.*, 1988] continues the earlier work and discusses a *self-learning* PID controller. The controller is *self-learning* in that it varies the scaling factors on-line as a function of the error.

Tang *et al.* [1987] relate the parameters of fuzzy controllers to the parameters of a conventional linear PI controller, where the fuzzy controller is defined:

$$K_3 \cdot [\Delta u(k)] = f[K_1 \cdot e(k), K_2 \cdot \Delta e(k)] \tag{6.3}$$

and the conventional linear PI controller is defined:

$$\Delta u(k) = K_P \cdot [K_I \cdot e(k) + \Delta e(k)\ ] \tag{6.4}$$

Thus (K_1, K_2, K_3) are related to (K_P, K_I). The technique described can also be applied to linear PD controllers, linear multiband and multilevel relay controllers.

Maeda *et al.* [1988] designed a fuzzy rule-based PID controller for two examples; control of vehicle speed and stabilization of an inverted pendulum. From their examples, the graphical results clearly show that control action from the fuzzy controller is *significantly* smoother than with manual or operator control of the same problem.

Shao [1988] modified the learning algorithm, which has been presented in detail by Chen *et al.* [1994] and reviewed in Chapter 7, for large relational matrices (i.e. 15×15×15) in order to produce a computationally efficient, rule-based fuzzy controller for on-line implementation. Qiao *et al.* [1992] simplify the method of learning the fuzzy control rules which are represented by an analytical expression with a regulating factor α. The fuzzy control rules for this system are described:

$$\textit{If } A \textit{ is } A_i \textit{ and } B \textit{ is } B_j \textit{ then } C \textit{ is } C_k \qquad (6.5)$$

where i, j and k are quantization levels

and $$k = \varphi(i, j) = \langle \alpha i + (1-\alpha) j \rangle, \qquad \alpha \in [0,1] \qquad (6.6)$$

Jang *et al.* [1992] extended the controller presented by Shao [1988] from the regulation control level to the optimization level. This optimization was achieved by introducing a modified form of the reduced gradient search procedure. The modification in the search procedure is that past experience is *learned* in the form of a linguistic process model which is used to improve future calculations. The results of these authors compare favourably against several well-known heuristic and gradient search algorithms.

Song *et al.* [1993] presented a dynamic learning fuzzy rule-based PI controller which utilizes a variable universe of discourse. That is, the range of the universe of discourse widens or narrows according to the magnitude of the error, thus permitting finer control than would be possible using a fixed universe of discourse. The learning algorithm is again similar to that outlined by Chen *et al.* [1994]. The controller design presented by these authors is compared against other fuzzy logic controller designs with good results.

Based on this review, it is quite evident that there is a large amount of research into the development of *fuzzy logic controllers* that parallels classical control design. Following behind the development of fuzzy control techniques is fuzzy control theory [Xu, 1990]. Because of the imprecise nature of fuzzy systems it is arguable whether precise analysis of fuzzy systems makes sense. But analysis of fuzzy systems may make sense, for two reasons. First, fuzzy models built through the aid of operator knowledge are not likely to be optimal, and secondly, techniques for fuzzy identification from plant data result in a numerical models in a form which lends itself readily to analysis [Xu, 1990].

The next two sections discuss fuzzy control theory as it pertains to systems analysis. The concepts of conventional control theory dominate in this review, however, the concepts are applied to a fuzzy environment.

6.4 Stability

Key to the analysis of any control systems is the determination of *stability*. However, this concept is not easily extended to *fuzzy* systems. The main reason for this is that by describing the state of the system as *fuzzy*, the concept of the system being *unbounded* becomes unclear. Consider the example by Tong [1980], for the statement "***X*** is ***large***" with the fuzzy membership function described in Figure 6.3. Clearly, from the distribution indicated in Figure 6.3, there is a possibility of x taking an infinitely large value. Tong [1980] questions whether this system would be stable. Also, for finite discrete systems described by *fuzzy relations*, would the imposition of finiteness ensure that the system can never become unbounded? It is Tong's [1980] opinion that *stability* should have a *fuzzy* definition. He suggests that the black or white definition of whether a system is *stable* or *unstable* should be replaced by the concepts of a system as having degrees of *stability* (or, *instability*).

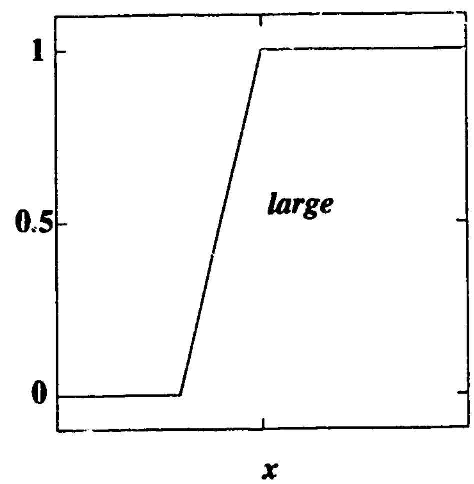


Figure 6.3: Membership Function for *Large* [Tong, 1980]

For fuzzy relational equations composed with *max-min* or *max-product* there exists, in most cases, a non-unique family of solutions. This family of solutions can be represented in two forms:

(1) different *fuzzy relational matrices*, $\boldsymbol{R} = f(\boldsymbol{u}, \boldsymbol{y})$ will yield the same *fuzzy output*, $\boldsymbol{y}$, for a given *fuzzy input*, $\boldsymbol{u}$.

(2) different *fuzzy inputs*, $\boldsymbol{u} = f(\boldsymbol{R}, \boldsymbol{y})$ will yield the same *fuzzy output*, $\boldsymbol{y}$, for a constant *relational matrix*, $\boldsymbol{R}$.

This property of a non-unique solution, as indicated in form (1), in some cases leads to a type of robustness for these forms of equations but in other cases leads to insensitivity to changes in control. However, considering form (2), the non-uniqueness can result in the undesirable feature of controller chattering, even though the output solution remains constant. This non-uniqueness of the solution is also an indication that precision is not essential when working with fuzzy systems.

Proving the stability of a fuzzy logic controller is one of the main concerns for industrial applications. Since the analytical nature of the controller is quite different from conventional controllers, classical stability analysis techniques are not applicable. The key question is often not whether the controller is stable but whether the process is stabilizable. In many cases it is felt that if a process can be controlled successfully by a human operator it should be possible to implement fuzzy control.

Some of the key issues of *fuzzy stability* are:

- ***Completeness*** which is concerned with ensuring that the controller is always able to compute a meaningful control action given the current process conditions. In other words, a controller should never be in a situation in which it does not know what to do [Tong, 1985]. A rule set is complete when every possible input state has a membership function value greater than an arbitrary cutoff level in at least one rule [Graham *et al.*, 1988].

$$\forall \mathbf{u} \in \mathcal{U} \; \exists j \in \{1, 2, \ldots, n\} \ni \mu_j(\mathbf{u}) = u_j > \varepsilon \in [0,1] \qquad (6.7)$$

- ***Consistency*** means that no two rules are in conflict such that they have the same antecedents but different consequences [Graham *et al.*, 1988].

- ***Interaction*** of the control rules occurs if:

$$\exists k \in \{1, 2, \ldots, N\} \ni u_k \not\subset (y_k * \boldsymbol{R}) \circledast y_{k+1} \qquad (6.8)$$

where $*$ is a t-norm operator (e.g. *min*, *product*)

$\circledast$ is the corresponding inverse composition

With ***interaction***, the fuzzy relation, $\boldsymbol{R}$, and the composition operator together modify the original fuzzy input causing a deformation. So the system, in these situations, is irreversible.

- ***Robustness*** of the fuzzy controller is concerned with errors in the closed-loop dynamics of the systems, as well as, the reaction of the controller to input disturbances. The numerical framework of the fuzzy relational equation allows for numerical analysis of the system which enables analysis to determine the influence of noise on the fuzzy controller.

Pedrycz [1989] discusses these issues in detail with recommendations to overcome some of these problems.

Several stability indexes consistent with Tong's [1980] suggestion of degrees of stability have been proposed. Kiszka *et al.*, [1985] proposed an index based on the energy of fuzziness of the fuzzy set. For the fuzzy state x_k, the energy $E(x_k)$ is defined as:

$$E(x_k) = \frac{\sum_{1\le i\le n} w(x_i)\cdot f(\mu_X(x_i))}{n} \tag{6.9}$$

where $w : \mathcal{X} \ni x_i \rightarrow \mathcal{R}$ is the mapping that takes into account the fuzzy set position on the universe of discourse $\mathcal{X}$.

$f : F(\mathcal{X}) \ni \mu_X(x_i) \rightarrow \mathcal{R}$ is the mapping that takes into account the maxima, the shape, the surface, the spread, the contrast and the degree of fuzziness of the fuzzy membership function.

and $\mathcal{R}$ is the set of non-negative real numbers

Stability is determined by the changes in $E(x_k)$ over time. Thus:

(*i*) the system is stable if:

$$E(x_k) - E(x_{k-1}) \le 0 \qquad \text{for } k \rightarrow \infty \tag{6.10}$$

(*ii*) the system is unstable if:

$$E(x_k) - E(x_{k-1}) \ge 0 \qquad \text{for } k \rightarrow \infty \tag{6.11}$$

(*iii*) the system is oscillatory if:

$$|E(x_k) - E(x_{k-1})| = |E(x_{k+\tau}) - E(x_{k-1+\tau})| \qquad \text{for } k \rightarrow \infty \tag{6.12}$$

Based on this definition it is possible to *measure* the energy of a fuzzy relation by considering its important physical properties, such as position of the support set, maxima of the membership function, shape, surface spread, contrast, convexity, degree of fuzziness, cardinality and volume. The inclusion of the physical properties depends on the specific form of the function f. However, the authors [Kiszka *et al.*, 1985] point out that the energy of a fuzzy system, based on the definition of equation (6.9) is more general, and in some cases better, than that of the strict physical definition.

Another stability index, described by Gupta *et al.* [1986], is directly related to the *fuzzy* system itself and is therefore more intuitive. Let Φ be a measure of fuzziness, expressed as:

$$\Phi(x) = \sum_{1 \le i \le n} x_i \tag{6.13}$$

for $x = \{x_i \mid i = \{1, 2, \ldots, n\}\} \in \mathcal{X}$. Then for $x_o \in \mathcal{X}$, an arbitrary initial state, and $x_k = x_o \circ R^k$, the desired state, the index of stability $S_\alpha(x_o)$ is defined as:

$$S_\alpha(x_o) = \min_{k \ge 0} [\Phi((x_o \wedge x_k)_\alpha), \Phi((x_k)_\alpha)] \tag{6.14}$$

where $\quad 0 \le \alpha \le 1$ is the α-cut parameter.

$S_\alpha(\cdot)$ can be considered the normalized measure of the distance between x_o and the farthest element in the sequence $\{x_1, x_2, \ldots, x_k\}$. The parameter α is introduced to filter out the noise caused by the *fuzzifying* property of the relational matrix R. The properties of $S_\alpha(\cdot)$ are such that:

$$(i) \quad 0 \le S_\alpha(x_o) \le 1 \tag{6.15}$$

$$(ii) \quad S_\alpha(x_o) = 1 \text{ iff } (x_k)_\alpha \subset (x_o)_\alpha \text{ for every } k \tag{6.16}$$

So the closer $S_\alpha(x_o)$ is to 1 the more stable is the system. However, the authors [Gupta *et al.*, 1986] point out that $S_\alpha(x_o)$ may be close to 1 and yet the system is unstable. This situation can take place if the relational matrix R is not sufficiently described or learned.

Stabilization of the closed-loop system by proper choice of the control action, u, is a key issue for *relational-based systems*. Consider the first order system without delay.

$$y_k = u_{k-1} \circ y_{k-1} \circ R \tag{6.17}$$

The stabilizing control for this fuzzy system, with a optimal output given in the form of the fuzzy set y_{opt}, refers to determining if there exists a stabilizing control $u = u_{ss}$ such that u_{ss} satisfies the following equation [Czogala *et al.*, 1982]:

$$y_{opt} = u_{ss} \circ y_{opt} \circ R \tag{6.18}$$

The solution of this problem can be computed easily through the inverse calculation presented in Chapter 5:

$$u_{ss} = (y_{opt} * R) \circledast y_{opt} \tag{6.19}$$

Alternatively, consider determining the fuzzy states, y, stabilized by the control action u_{ss}. From the first order non-delay system:

$$y_{k+1} = u_{ss} \circ y_k \circ R \tag{6.20}$$

Combine u_{ss} and R:

$$P = u_{ss} \circ R \tag{6.21}$$

Substituting equation (6.21) into (6.20) results in:

$$y_{k+1} = P \circ y_k \tag{6.22}$$

At steady state $y_{k+1} = y_k$ and the collection of output states stabilized by u_{ss} is the family of all eigen fuzzy sets of P.

The theory to determine the existence of the eigen fuzzy set for equation (6.22), and if it exists, the greatest element of the same was developed by Sanchez [1978] for the *max-min* composition. A brief review of this theory is presented next.

Using the same notation as established in Chapter 5, define $\mathcal{V} \equiv \mathcal{W}$. Then an *eigen* fuzzy set $a \in \mathcal{V}$ of a known fuzzy relation $R \in \mathcal{V} \times \mathcal{V}$ is a fuzzy set on $\mathcal{V}$ such that $a \circ R = a$. The proof of the existence of the greatest eigen fuzzy set (GEFS) is established as follows:

(a) Let $a_1 \in V$ be the fuzzy set such that the grades of membership are equal to the greatest element for each column of R.

$$a_1(v') = \bigvee_{v} R(v, v') \qquad \text{for } \forall v' \in \mathcal{V} \tag{6.23}$$

(b) Let $a_0 \in V$ be a constant fuzzy set of the minimum of these values.

$$a_0(v) = a_1(v') \qquad \text{for } \forall v' \in \mathcal{V} \tag{6.24}$$

This constant fuzzy set is an eigen fuzzy set of R, but not always the GEFS.

(c) Define the sequence of fuzzy sets $\boldsymbol{a}_n$ by:

$$\boldsymbol{a}_2 = \boldsymbol{R}\circ\boldsymbol{a}_1$$
$$\boldsymbol{a}_3 = \boldsymbol{R}\circ\boldsymbol{a}_2 = \boldsymbol{R}^2\circ\boldsymbol{a}_1$$
$$\vdots$$
$$\boldsymbol{a}_{n+1} = \boldsymbol{R}\circ\boldsymbol{a}_n = \boldsymbol{R}^n\circ\boldsymbol{a}_1$$

It is obvious that the sequence $\boldsymbol{a}_n$ is decreasing and bounded by $\boldsymbol{a}_0$ and $\boldsymbol{a}_1$.

$$\boldsymbol{a}_0\subseteq\ldots\subseteq\boldsymbol{a}_{n+1}\subseteq\boldsymbol{a}_n\subseteq\ldots\subseteq\boldsymbol{a}_3\subseteq\boldsymbol{a}_2\subseteq\boldsymbol{a}_1 \tag{6.25}$$

Additionally, Sessa [1984] has provided the conditions necessary for an eigen fuzzy solution to problem statement (4) defined in Chapter 5, and these are presented next.

Definition 1: $\Gamma_{ik} = \{v_j \in \mathcal{V} \mid \boldsymbol{Q}(u_i,v_j) \geq \boldsymbol{T}(u_i,w_k)\}$ for any $i\in \mathbf{I}_n$, $k\in \mathbf{K}_p$

Theorem 1: If $\mathcal{U} = \mathcal{V}$ and $\Gamma_{ik} = \{x_i\}$ for any $i\in \mathbf{I}_n$, $k\in \mathbf{K}_p$ we have $\Sigma = \boldsymbol{T}$.
Moreover, $\boldsymbol{T} = \boldsymbol{Q}\circ\boldsymbol{T}$

Sanchez [1978] also presents three methods of determining the greatest eigen fuzzy set (GEFS) of $\boldsymbol{R}$. The results of these methodologies were used to solve *problems of invariants in therapeutic recommendations*. The knowledge being sought was:

> *Given a fuzzy relation **R** between medical symptoms expressing the action of a drug on patients in a specific therapy, what is the greatest intensity of each symptom on which **R** produces no effect?*

Now consider repeated application of equation (6.22) from an initial point in time, y_o.

$$\begin{aligned} y_1 &= \boldsymbol{P}\circ\boldsymbol{y}_o \\ y_2 &= \boldsymbol{P}\circ\boldsymbol{y}_1 = \boldsymbol{P}^2\circ\boldsymbol{y}_o \\ &\vdots \\ y_k &= \boldsymbol{P}^k\circ\boldsymbol{y}_o \end{aligned} \tag{6.26}$$

The stability of the powers of $\boldsymbol{P}$ for the *max-min* composition has been analyzed by Thomason [1977], where the powers of any matrix $\boldsymbol{R}$ are defined recursively by:

Given $\quad \boldsymbol{R}^0 = \mathbf{I}, \quad$ where $\quad \mathbf{I}$ is the $n \times n$ unit diagonal matrix (6.27)

then $\quad \boldsymbol{R}^k = \boldsymbol{R}^{k-1} \circ \boldsymbol{R}, \quad$ for $k = 1, 2, ,3, \ldots$ (6.28)

The paper by Thomason showed that for the *max-min* composition, the powers of $\boldsymbol{R}$:

(*i*) converge to an idempotent $\boldsymbol{R}^c$

(*ii*) oscillate with a finite period τ

This same stability of powers theory is now extended to the *max-product* composition to ensure stability of the predictive fuzzy logic controller proposed in this thesis. This theory, as applied to the *max-product* composition, is of particular interest and concern due to the fact that for some relational matrices, $\boldsymbol{R}$:

$$\lim_{n \to \infty} \boldsymbol{R}^n = [0] \tag{6.29}$$

When a control application is being considered, this degradation of the relational matrix is unacceptable for stable servo control. Being able to predict or test for this degradation would be a requirement for any control system using the *max-product* composition. And then once stability of the powers is established, determining the family of eigen fuzzy sets of the matrix $\boldsymbol{R}$ or alternatively, determining the range of $\boldsymbol{R}$ required to maintain a desired output must then be considered.

For this discussion a simple measure of the shape of the fuzzy sets of the relation will be utilized. Tong [1978] proposed a measure using the position of the peaks in the membership function which defines the set and found it to be a satisfactory definition for discussing the concepts of stability and controllability.

The following definitions taken from Tong [1978] will formalize the concept of a peak pattern for eigen fuzzy stability discussion.

Definition 2: A *peak pattern*, *PP*, of a fuzzy set X on a finite set $\mathcal{X}$ is a binary map of the discrete membership function of X and μ_x with

$$PP_X(x) = \begin{cases} 0 & if \quad \mu_X \neq \mu_{max} \\ 1 & if \quad \mu_X = \mu_{max} \end{cases} \tag{6.30}$$

where μ_{max} is the maximum value of the membership function (i.e. $\mu_{max} = \mathbf{1}$).

Definition 23: A peak pattern *covers* an element $x \in \mathcal{X}$ if $PP(x) = \mathbf{1}$.

Definition 24: Two fuzzy sets X_1 and X_2 are *equivalent* if they have the same peak pattern.

Definition 25: A fuzzy set with a peak pattern which covers only one element in $\mathcal{X}$ is called a *singular* set.

Definition 26: A relation $\boldsymbol{R}$ is a *maximal* relation if each row has at least one element of value **1**.

The analysis for the powers of a matrix $\boldsymbol{R}$, for the *max-product* composition, is for matrices which are ≤ *maximal*.

Proposition 12: For the *max-product* composition, the powers of $\boldsymbol{R}$:

(i) converge to the null matrix $[\mathbf{0}]$

(ii) converge to an idempotent $\boldsymbol{R}^c$

(iii) oscillate with a finite period $\boldsymbol{\tau}$

Proof:

(i) If $r_{ij} < \mathbf{1}$ for all $i, j \leq n$ then

$$\lim_{n \to \infty} r_{ij}^{\,n} \Rightarrow 0 \tag{6.31}$$

(ii) If a dominant factor is present in all powers of $\boldsymbol{R}$ and that factor is $\mathbf{1}$, then the matrix will converge to a finite $\boldsymbol{R}^c$. For example, the powers of the diagonal elements are all functions of themselves, so if they are set to the value 1, they will remain at the value 1.

$$r_{ii}^n = r_{ii}^n + r_{ij} r_{ji} r_{ii}^{n-2} + \dots \tag{6.32}$$

(iii) If the dominate factor, 1, does not occur in the same position in all powers of $\boldsymbol{R}$, then the matrix will oscillate. For example in a 3×3 matrix,

$$\boldsymbol{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad \boldsymbol{R}^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \neq \boldsymbol{R} \qquad \boldsymbol{R}^3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \boldsymbol{R}$$

Proposition 13: If the *peak pattern* equivalent of a matrix, $\boldsymbol{R}$, converges then the matrix $\boldsymbol{R}$ also converges

Proof: In the *peak pattern* equivalent, all matrix elements whose values are < 1 are replaced by zero (0), and matrix multiplication quickly eliminates all non-dominate factors. Only those values of 1 remain that can be sustained in repeated powers. This same elimination of non-dominate factors take place in the original matrix, however, convergence is slower.

Theorem 8: The powers of a matrix $\boldsymbol{R}$ will converge to an idempotent $\boldsymbol{R}^c$ for a finite c if:

(i) there exists at least one $i \leq n$ such that $r_{ii} = 1$

and (ii) there does not exist a $j, k \leq n$ such that $r_{jk} = r_{kj} = 1$, where $j, k \neq i$

Proof: Direct from Proposition 12(ii) and (iii).

Theorem 9: The powers of a matrix $\boldsymbol{R}$ will ocillate with a finite period τ if:

(i) there exists an $i \leq n$ such that $r_{ii} = 1$

and (ii) there exits a $j, k, i \leq n$ such that $r_{jk} = r_{kj} = 1$, where $j, k \neq i$.

or (iii) the matrix is *maximal* and there does not exist an $i \leq n$ such that $r_{ii} = 1$.

Proof: Direct from Proposition 12(iii).

Theorem 10: The powers of a matrix, $\boldsymbol{R}$, will converge to the null matrix [0] if:

(i) all $r_{ij} < 1$ for all $i, j \leq n$

or (ii) all $r_{ii} < 1$ for all $i \leq n$

and (iii) $\boldsymbol{R} <$ *maximal*

Proof: Direct from Proposition 12(i)

Definition 27: The determinant $|\boldsymbol{R}|$ of the fuzzy matrix $\boldsymbol{R}$ is defined:

$$|\boldsymbol{R}| = \sum_{p_1 p_2 \cdots p_n} r_{1p_1} r_{2p_2} \ldots r_{np_n} \tag{6.33}$$

where $\sum_{p_1 p_2 \cdots p_n}$ denotes the *max* for all permutations $(p_1, p_2, \ldots, p_n)$ of the indices $(1, 2, \ldots, n)$

Definition 28: The $n \times n$ adjoint matrix of $\boldsymbol{R}$, adj($\boldsymbol{R}$), is defined:

$$a_{ij} = |\boldsymbol{R}_{ji}| \tag{6.34}$$

where a_{ij} is an element of the adj($\boldsymbol{R}$), and $|\boldsymbol{R}_{ji}|$ denotes the determinant of the $(n\text{-}1)\times(n\text{-}1)$ matrix formed by deleting row j and column i from $\boldsymbol{R}$.

Proposition 14: If $\boldsymbol{R} \geq \mathbf{I}$ then the powers of $\boldsymbol{R}$ converge to $\boldsymbol{R}^c$ and $\boldsymbol{R}^c = \text{adj}(\boldsymbol{R})$

Proof: Similar to proof of Proposition 4 of Thomason [1977]

Due to the complexities and interaction of the multiplicative operator, the number of iterations, c, to reach the stable matrix formation may be greater than n. However, analysis on the *peak pattern* matrix equivalent, as defined by Tong [1978] can be performed to ensure convergence to a stable matrix form. For these matrices the operations *max-min* and *max-product* are equivalent. Thus all the results of Thomason [1977] are applicable to these matrix equivalents and convergence to c will be in $\leq n\text{-}1$ iterations.

The eigen fuzzy analysis is now concerned with the convergence of a initial $n \times 1$ state vector, $\boldsymbol{D}$, with elements $d_i \in [0,1]$, to final state, $\boldsymbol{D}^c$. Define the fuzzy state process,

$$\boldsymbol{D}^1 = \boldsymbol{D} \tag{6.35}$$

$$\boldsymbol{D}^k = \boldsymbol{R} \circ \boldsymbol{D}^{k-1}, \qquad \text{for } k = 2, 3, \ldots \tag{6.36}$$

Theorem 11: $\boldsymbol{D}$ converges to $\boldsymbol{D}^c$, iff $\boldsymbol{R}$ converges to $\boldsymbol{R}^c$.

Proof: $\Rightarrow$

$\boldsymbol{D}^2 = \mathbf{R} \circ \boldsymbol{D}^1$

$\boldsymbol{D}^3 = \boldsymbol{R} \circ \boldsymbol{D}^2 = \boldsymbol{R}^2 \circ \boldsymbol{D}^1$

so $\boldsymbol{D}^{c+1} = \boldsymbol{R}^c \circ \boldsymbol{D}$

$\boldsymbol{D}^{c+2} = \boldsymbol{R}^{c+1} \circ \boldsymbol{D} = \boldsymbol{R}^c \circ \boldsymbol{D}$ since $\boldsymbol{R}$ has converged to $\boldsymbol{R}^c$

$\therefore$ $\boldsymbol{D}^{c+1} = \boldsymbol{D}^{c+2}$ converges

$\Leftarrow$ $\boldsymbol{D}^{c+1} = \boldsymbol{R} \circ \boldsymbol{D}^c = \boldsymbol{R}^c \circ \boldsymbol{D}$

$\boldsymbol{D}^{c+2} = \boldsymbol{R} \circ \boldsymbol{D}^{c+1} = \boldsymbol{R}^{c+1} \circ \boldsymbol{D}$

but $\boldsymbol{D}^c = \boldsymbol{D}^{c+1} = \boldsymbol{D}^{c+2}$ since $\boldsymbol{D}$ has converged to $\boldsymbol{D}^c$

$\therefore$ $\boldsymbol{R}^{c-1} = \boldsymbol{R}^c$ converges

The solutions to the problem of the eigen fuzzy set for the *max-product* operator will now be considered. The solutions will be address the following problem definitions:

(a) Given $\boldsymbol{R}$, find the GEFS of $\boldsymbol{D}$

(b) Given $\boldsymbol{D}$, find the GEFS of $\boldsymbol{R}$

(c) Given $\boldsymbol{R}$, find $\boldsymbol{R}^c$

(d) Given $\boldsymbol{D}$, and $\boldsymbol{R}$ find $\boldsymbol{D}^c$

where GEFS stands for the *greatest eigen fuzzy set*.

Problem (a): **Algorithm to determine D_{GEFS}, given R**

Assume R has a *peak pattern* that converges, therefore R converges.

(1) Calculate $R^* = R$ with all columns without 1's deleted

(2) Calculate $D^* = \max((R^*)^{-1})^t$

(3) Then $D_{GEFS} = R \circ D^*$

If D_{GEFS} contains only one element with the value of 1, then all other eigen fuzzy vectors are linear multiples of D_{GEFS}.

Problem (b): **Algorithm to determine R_{GEFS}, given D**

This is the straight inverse of the problem $D = R \circ D$

$$R_{GEFS} = D^t \textcircled{\psi} D$$

Problem (c): **Algorithm to determine R^c, given R**

(i) If R is *maximal* and the ones are along the diagonal, then R^c can be determined directly by Proposition 3.

$$R^c = \text{adj}(R)$$

(ii) If R has only a single diagonal element with the value of 1 and any other non-diagonal 1's such that R converges or no other additional 1's, then R^c can be determined as follows:

(1) Calculate R^2

(2) Delete all entries from R^2 except the *i*th row and column, where *i* is the row and column positioning of the diagonal 1.

(3) The blank column entries calculated by multiplying the one non-blank column entry in each incomplete column by the *i*th column vector.

Example:

$$R = \begin{bmatrix} 0.2 & 0.7 & 0.4 \\ 0.8 & 0.9 & 0.5 \\ 0.4 & 0.7 & 1.0 \end{bmatrix} \quad R^2 = \begin{bmatrix} 0 & 0 & 0.4 \\ 0 & 0 & 0.5 \\ 0.56 & 0.7 & 1.0 \end{bmatrix} \quad R^c = \begin{bmatrix} 0.224 & 0.28 & 0.4 \\ 0.28 & 0.35 & 0.5 \\ 0.56 & 0.7 & 1.0 \end{bmatrix}$$

(iii) All other combinations have too many interactions for a simplified solution. Solutions are obtained by repeated iteration.

Problem (d): Algorithm to determine D^c, given D and R

(i) If the matrix, R, has only one diagonal **1**, then the following algorithm provides the D^c for the given D:

(1) Calculate R^2

(2) Delete all entries from R^2 except the *i*th row and column, where *i* is the row and column positioning of the diagonal **1**.

(3) Determine the *max-product* of D and the *i*th row.

(4) Multiply this result by the *i*th column

Example:

$$R = \begin{bmatrix} 0.2 & 0.7 & 0.4 \\ 0.8 & 0.9 & 0.5 \\ 0.4 & 0.7 & 1.0 \end{bmatrix} \quad R^2 = \begin{bmatrix} 0 & 0 & 0.4 \\ 0 & 0 & 0.5 \\ 0.56 & 0.7 & 1.0 \end{bmatrix} \quad D = \begin{bmatrix} 0.8 \\ 0.5 \\ 0.3 \end{bmatrix}$$

$\alpha = 0.448$

$$D^c = \begin{bmatrix} 0.1792 \\ 0.224 \\ 0.448 \end{bmatrix}$$

(ii) All other D^c's must be calculated as follows:

(1) Invert or transpose R.

$$R = \begin{bmatrix} 0.3 & 0.7 & 0.9 \\ 0.4 & 1.0 & 0.7 \\ 0.2 & 0.6 & 1.0 \end{bmatrix} \qquad R^{-1} = \begin{bmatrix} 0.3 & 0.4 & 0.2 \\ 0.7 & 1.0 & 0.6 \\ 0.9 & 0.7 & 1.0 \end{bmatrix}$$

(2) Calculate the *max-product* of D with each column of $\mathbf{R}^{-1}$ with a diagonal **1**.

$$R^{-1} = \begin{bmatrix} 0.3 & 0.4 & 0.2 \\ 0.7 & 1.0 & 0.6 \\ 0.9 & 0.7 & 1.0 \end{bmatrix} \circ \begin{bmatrix} 0.7 \\ 0.6 \\ 0.3 \end{bmatrix}$$

$$D^{\bullet} = \begin{bmatrix} 0 & 0.6 & 0.36 \end{bmatrix}$$

(3) Calculate the *max-product* of $D^{\bullet}$ with each column of $\mathbf{R}^{-1}$.

$$R^{-1} = \begin{bmatrix} 0.3 & 0.4 & 0.2 \\ 0.7 & 1.0 & 0.6 \\ 0.9 & 0.7 & 1.0 \end{bmatrix} \circ \begin{bmatrix} 0 \\ 0.6 \\ 0.36 \end{bmatrix}$$

$$D^{c} = \begin{bmatrix} 0.42 & 0.6 & 0.36 \end{bmatrix}$$

From a control perspective, set-point tracking requires both relational matrix stability and eigen fuzzy set determination, with the latter being impossible if the relational matrix is unstable. As evident from the stability of powers of R, from this thesis and the work by Thomason [1977], the requirements assigned to the relational matrix to ensure stability are rigorous. In many cases, these requirement may be impossible to ensure while maintaining acceptable control performance. So can practical fuzzy logic applications be formulated to meet control objectives and overcome instability caused by the powers of R? The following discussion explains how fuzzy logic applications maintain stability during setpoint tracking and offers a method to ensure stability for control situations when the output must be defuzzified for a discrete result.

In their work on fuzzy learning and identification, Shaw *et al.* [1992] introduced the term *leakage* and presented a method to overcome the problem that this distortion causes. *Leakage* occurs when a relational matrix, R, introduces additional non-zero terms into the calculated fuzzy output data that are not present in the actual output data. In these instances, the operations of fuzzification and defuzzification are no longer reciprocal. This phenomenon of *leakage* necessitates the use of approximate defuzzification techniques, such as *max of maximum* or *center of area*. To overcome the problem of *leakage* in their identification algorithm, Shaw *et al.* [1992] suggested defuzzifying the calculated output and then refuzzifying it before applying the value as a state estimator into the first order fuzzy system model. This same methodology can be used to stabilized control systems when the servo requirements of the system generate powers of R.

In most identification and control situations, the methods of fuzzification and defuzzification are not reciprocal. Now consider equation (6.22) and the stabilization of fuzzy outputs, y, (i.e. $y_k = y_{k-1}$) by the control action u_{ss}.

$$y_{k+1} = P \circ y_k \tag{6.22}$$

Feeding back the defuzzified and then refuzzified previously calculated output, y_k, eliminates the problem of instability caused by the powers of R, because it prevents increasing powers of R from being calculated since the fuzzified values of y_{k+1} and y_k are not equal. The discrete or defuzzified values of y_{k+1} and y_k are equal at steady state so there is no effect from this procedure on the measured output.

The presence of system noise can also eliminate this stability problems in practical discrete control applications. Consider equation (6.22) with $y_{sp} = y_{k+1}$, and y_{sp} calculated by the method of fuzzification. The value of y_k is the previous actual discrete output, fuzzified by the same method as y_{sp}, so there is a possibility that $y_k = y_{sp}$. In applications with noise, when the actual output is tracking in an acceptable region about the setpoint, the values of y_{k+1} and y_{sp} may not be equal for a sustained length of time, and therefore the powers of $\boldsymbol{R}$ generation is limited and model instability is prevented.

This method of defuzzifying and fuzzifying the previous output, y_k, prevents both the *max-product* and *max-min* compositions from oscillating, and, prevents the relational matrix of the *max-product* composition from converging to zero.

As a final note, since this Chapter deals with the similarities between the analysis of *fuzzy* and conventional control systems, it seems important to point out here that the term *eigen* used in *fuzzy systems* is not directly related to *eigen values*, poles and/or process stability as it pertains to conventional control. This is just a coincidence of terminology.

This completes the review of stability analysis for relational matrices using either the *max-min* or *max-product* composition. The analysis presented in this section was chosen to demonstrate the similarities between conventional and fuzzy control.

6.5 Controllability

For a conventional first order system, the definition of controllability is to be able to *drive* an arbitrary state to the origin in one step. Or more generally, to *drive* the state to the origin in $N < \infty$ steps.

The classical definition of complete or *hard* controllability is inappropriate for the fuzzy relational equation being considered here. There are two reasons for this [Gupta *et al.*, 1986]:

(*i*) fuzzy relational equations often have mapping defects, meaning that a state derived from a relational equation is not necessarily exactly equal to the states used in the construction of the relational matrix.

(*ii*) for these *fuzzy* or *soft* systems it is often adequate to attain a state *close* to the desired *fuzzy state*.

Consider the *controllability* of the *fuzzy first order model* with no time delay ($\tau = 0$):

$$y_{k+1} = u_k \circ y_k \circ R \tag{6.37}$$

The problem or goal of the *fuzzy controller* is to [Pedrycz, 1985(a)]:

Find the fuzzy control u_k *that allows the fuzzy system to obtain a fuzzy goal* $G(y_{k+1})$ *starting from a previously specified* y_k.

Or, in other words, *find the fuzzy cause*. So equation (6.37) can be rewritten:

$$G(y_{k+1}) = u_k \circ y_k \circ R \tag{6.38}$$

It is known from the inverse formulation that there may exist a family of solutions, $u_k(G, y_k)$, for this fuzzy goal, the greatest element of which is $\hat{u}_k$. So if the set $u_k(G, y_k)$ is non empty, the fuzzy model is controllable for (G, y_k). The property of controllability is tied to the current output and the goal simultaneously. It should be noted that the property of controllability is local, that is a system may be controllable for y_k but uncontrollable for y'_k.

For the case when the family of solutions, $u_k(G, y_k)$, is empty, Pedrycz [1985(b)] discusses *approximate controllability of the fuzzy system*. In this work by Pedrycz [1985(b)] a *fuzzy index of equivalence* is defined and treated as the *controllability index*.

The *fuzzy index of equivalence* or the degree of equivalence of fuzzy sets x and x', $[x = x']$, is calculated:

$$[x = x'] = \min_i [x(x_i) \;\psi\; x'(x_i)] \;\mathbf{t}\; \min_i [x'(x_i) \;\psi\; x(x_i)] \tag{6.39}$$

where $\mathbf{t}$ and ψ are a *t*-norm operation and its associated inverse, respectively, which have been defined in Chapter 5. Results of this calculation span the range [0,1], with a value of 1 indicating complete equivalence and a value of 0 meaning completely different.

A threshold level, $\alpha \in [0,1]$, can be introduced into the *equivalency measure* under the assumption that small values of the membership function are meaningless and equivalence of fuzzy sets should be evaluated with reference to higher values. So the *index of equivalence* can be restated as:

$$[x = x'](\alpha) = \min_i [x_\alpha(x_i) \;\psi\; x_\alpha'(x_i)] \;\mathbf{t}\; \min_i [x_\alpha'(x_i) \;\psi\; x_\alpha(x_i)] \tag{6.40}$$

where x_α and x_α' are the original fuzzy sets x and x', respectively, modified by α in the following manner:

$$x_\alpha = x \vee \alpha \tag{6.41}$$

$$x_\alpha' = x' \vee \alpha \tag{6.42}$$

With this definition of α-*equivalency*, x is exactly equal to x' if $[x = x'](\alpha) = 1$ for all α, and x is completely different from x' if $[x = x'](1) = 1$ and 0 otherwise.

This concept of *α-equivalency* is applied to the *controllability index* in the following manner. With a known *fuzzy model*, $\boldsymbol{R}$, a given goal, $G(y_{k+1}) = y_{sp}$, and the previous output, y_k, the fuzzy control is calculated as:

$$\hat{u}_k = (y_k \cdot R) \circledS y_{sp} \tag{6.43}$$

The prediction of the next output, $\hat{y}_{k+1}$ is then calculated as:

$$\hat{y}_{k+1} = \hat{u}_k \circ y_k \circ R \tag{6.44}$$

The (G, y_k)-controllability of the system is determined by the value of $[\hat{y}_{k+1} = y_{sp}]$. If $[\hat{y}_{k+1} = y_{sp}] = 1$, the system has absolute controllability. If $[\hat{y}_{k+1} = y_{sp}] = 0$, the system has total uncontrollability.

Now consider a *fuzzy constraint*, $C(u)$, on the fuzzy input, u, that must be satisfied simultaneously with the goal, $G(y_{k+1})$. To solve this problem $C(u)$ is introduced to both sides of equation (6.38) [Pedrycz, 1985(a)]:

$$G(y_{k+1}) \cdot C(u) = u_k \circ y_k \circ R \circ C(u) \tag{6.45}$$

Define:

$$W(y_{k+1}, u) = G(y_{k+1}) \cdot C(u) \tag{6.46}$$

So to determine (W, y_k)-controllability of the constrained system, the following equations must be considered:

$$\hat{u}_k = (C \cdot y_k \cdot R) \circledS W \tag{6.47}$$

$$\hat{W} = \hat{u}_k \circ y_k \circ R \circ C \tag{6.48}$$

and $$[\hat{W} = W]. \tag{6.49}$$

Gupta *et al.* [1986] also define an index of controllability based on the closeness of a calculated state $\boldsymbol{x}_k = \boldsymbol{x}_o \circ \boldsymbol{u}_o \circ \boldsymbol{u}_1 \circ \ldots \circ \boldsymbol{u}_{k-1} \circ \boldsymbol{R}^k$ to a final state $\boldsymbol{x}_f$ after a sequence on control moves $\boldsymbol{u}_o$, $\boldsymbol{u}_1$, ..., $\boldsymbol{u}_{k-1}$. Due to the fuzzifying property of $\boldsymbol{R}$, the expectation that $\boldsymbol{x}_k$ is exactly equal to $\boldsymbol{x}_f$ for some k, is unrealistic. Thus the controllability index, $\boldsymbol{C}_{\alpha,k}(\boldsymbol{x}_o, \boldsymbol{x}_f)$ is defined as:

$$\boldsymbol{C}_{\alpha,k}(\boldsymbol{x}_o, \boldsymbol{x}_f) = \max_{u_o,\ldots,u_{k-1}} \Phi((x_k \wedge x_f)_\alpha) \big/ \Phi((x_k)_\alpha) \tag{6.50}$$

where Φ is the measure of fuzziness defined earlier. Equation (6.50) is interpreted as; the larger $\boldsymbol{C}_{\alpha,k}(\boldsymbol{x}_o, \boldsymbol{x}_f)$, the closer the state $\boldsymbol{x}_k$ is to $\boldsymbol{x}_f$, or, the better is the controllability. Again the α-cut is introduced into this index to filter the fuzzification caused by the fuzzy relation, $\boldsymbol{R}$.

Xu [1990] equates a *fuzzy* r_o-reachability with controllability in linear systems. A system:

$$\boldsymbol{y}_k = \boldsymbol{y}_{k-\tau_1} \circ \boldsymbol{u}_{k-\tau_2} \circ \boldsymbol{v}_{k-\tau_3} \circ \boldsymbol{R} \tag{6.51}$$

where $\boldsymbol{y}_k$, $\boldsymbol{u}_k$, and $\boldsymbol{v}_k$ are the fuzzy output, control and disturbance variables at time instant k

$\boldsymbol{R}$ is the fuzzy relational matrix

$\circ$ is the *max-min* composition

is considered r_o-reachable if for any r_o-normal $\boldsymbol{y}_{k-\tau_1}$ and $\boldsymbol{v}_{k-\tau_3}$, a fuzzy control variable $\boldsymbol{u}_{k-\tau_2}$ can be found such that the fuzzy output $\boldsymbol{y}_k$ can be transferred to a given point yi. This point yi is such that the membership function of $\boldsymbol{y}_k$ has a unique peak at yi and $\boldsymbol{y}_k(yi) \geq r_o$, for $r_o \in [0,1]$. The term r_o-normal is defined as follows.

Let $\boldsymbol{X}$ be a universe and xi be a point in that universe. Let $\boldsymbol{x} \in \boldsymbol{X}$ be the fuzzy set for the point xi with membership functions $\boldsymbol{x}(xi)$. Then:

(a) $\boldsymbol{x}$ is said to be r_o-unimodal, iff

$$\exists!\, xo \in \boldsymbol{X}, \qquad \boldsymbol{x}(xo) \geq r_o \text{ and } \boldsymbol{x}(xi|xo) < r_o \tag{6.52}$$

($\exists! \equiv$ there exists one and only one)

(b) $\boldsymbol{x}$ is said to be r_o-normal, iff

(*i*) $\boldsymbol{x}$ is r_o-unimodal, (6.53)

and (*ii*) $\boldsymbol{x}$ is convex, (6.54)

(or, for all i, j, k, $xi \leq xj \leq xk$, $\boldsymbol{x}(xj) \geq \min[\boldsymbol{x}(xi), \boldsymbol{x}(xk)]$)

It should be noted here that the definition by Xu [1990] for *normal* is slightly different from the standard definition. Based on the theory presented for reachability of an open-loop fuzzy systems, Xu [1990] developed a theoretical fuzzy feedback/feedforward control law for one-step ahead reachability.

Several methodologies presented above indicate that controllability of *fuzzy relational systems* can be calculated in a manner analogous to classical systems, however in a form more suitable for the analyses of *fuzzy relational systems*.

6.6 Summary

The controllers that have been reviewed in this chapter are basically *fuzzy feedback controllers*, and, as was noted, fuzzy feedback or fuzzy PID control closely mirrors conventional PID control. However, as with the conventional PID controller, there is no predictive ability built into the fuzzy PID controller.

Chapter 6 is important from a control perspective as it consolidates fuzzy design stability theory and demonstrates the agreement between fuzzy control and conventional classical control. Two important results from this Chapter are:

(1) Stability analysis and convergence properties for relational matrices combined with the *max-product* operator [Bourke *et al.*, 1995]

(2) Eigen Fuzzy Sets analysis for *max-product* compositions [Bourke *et al.*, 1995]

The stability analysis and convergence property results of this work are critical for relational matrices combined with the *max-product* composition because of the possibility that these relational matrices may converge to a [0] or null matrix. Therefore, before the development of a control policy with the *max-product* composition, the conditions for the existence of the unstable solution matrices must be determined.

Ability of a systems to obtain and maintain a setpoint under a control is also critical, and knowledge of the conditions under which deterioration may result is crucial. Eigen fuzzy set analysis reveals the ability of a relational matrix, combined by successive *max-product* compositions, to maintain a setpoint under a control scenario. As well, a method to overcome poor or deteriorating response for those matrices that do not meet the criteria of stability with successive composition is provided for both *max-min* and *max-product* composition.

The stability and controllability analyses that have been reviewed deal with fuzzy models and can therefore be applied to fuzzy model-based control. With this basis of fuzzy control theory now established, the development of on-line identification algorithms will be considered, in the next Chapter, leading to the development of a self-learning fuzzy model-based predictive controller.

REFERENCES

[1] Bourke, M.M., Fisher, D.G., "Convergence, Eigen Fuzzy Sets and Stability Analysis of Relational Matrices", *Fuzzy Sets and Systems*, accepted 1995.

[2] Buckley, J.J., "Controllable Process and the Fuzzy Controller", *Fuzzy Sets and Systems*, Vol. 52, pp. 27-31, 1993.

[3] Chen, J.Q., Lu, J.H., Chen, L.J., "An On-Line Identification Algorithm for Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 64, pp. 63-72, 1994.

[4] Czogala, E., Pedrycz, W., "Control Problems in Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 7, pp. 257-273, 1982.

[5] Czogala, E., Pedrycz, W., "Fuzzy Rules Generation for Fuzzy Control", *Cybernetics and Systems: An International Journal*, Vol 13, pp. 275-293, 1982.

[6] Graham, B.P., Newell, R.B., "Fuzzy Identification and Control of a Liquid Level Rig", *Fuzzy Sets and Systems*, Vol. 26, pp. 255-273, 1988.

[7] Graham, B.P., Newell, R.B.. "Fuzzy Adaptive Control of a First Order Process, *Fuzzy Sets and Systems*, Vol. 31, pp. 47-65, 1989.

[8] Gupta, M.M., Trojan, G.M., Kiszka, J.B., "Controllability and Fuzzy Control Systems", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 16, pp. 576-581, 1986.

[9] Jang, S.-S., Wong, D. S.-H., Liau, C.-K., "On-line/Off-line Optimization of Complex Processes using a Linguistic Self-Organized Optimizing Control Scheme", *Fuzzy Sets and Systems*, Vol. 47, pp. 23-33, 1992.

[10] Kiszka, J.B., Gupta, M.M., Nikiforuk, P.N., "Energetistic Stability of Fuzzy Dynamic Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 15, pp. 783-792, 1985.

[11] Kosko, B., *Neural Networks and Fuzzy Systems*, Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1992.

[12] Lee, C.C., "Fuzzy Logic in Control Systems: Fuzzy Logic Controller", *IEEE Transactions on Systems, Man and Cybernetics*, Vol 20, Part I: pp. 404-418, Part II: pp. 419-435, 1990.

[13] Maeda, M., Murakami, S., "A Design for a Fuzzy Logic Controller", *Information Sciences*, Vol. 45, pp. 315-330, 1988.

[14] Mamdani, E.H., "Application of Fuzzy Algorithms for Control of Simple Dynamic Plant", *Proceeding of IEE*, Vol. 121, No. 12, pp. 1585-1588, December 1974.

[15] Mamdani, E.H., Assilian, S., "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", *International Journal of Man-Machine Studies*, Vol. 7, pp. 1-13, 1975.

[16] Negoita C.V., Ralescu, D.A., *Applications of Fuzzy Sets to System Analysis*, Birkhäuser Verlag, Basel, 197[illegible]

[17] Pedrycz, W., "An Approach to the Analysis of Fuzzy Systems", *International Journal of Control*, Vol. 34, pp. 403-421, 1[illegible]1.

[18] Pedrycz, W., "Numerical and Applicational Aspects of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 11, pp. [illegible]-18, 1983.

[19] Pedrycz, W., "Design of Fuzzy Control Algorithms with the Aid of Fuzzy Models", in: *Industrial Applications of Fuzzy Control*, Sugeno, M. (Editor), Elsevier Science Publishers, North-Holland, pp. 153-173, 1985(a).

[20] Pedrycz, W., "On Generalized Fuzzy Relational Equations and Their Applications", *Journal of Mathematical Analysis and Applications*, Vol. 107, pp. 520-536, 1985(b).

[21] Pedrycz, W., "Approximate Solutions of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 28, pp. 183-202, 1988.

[22] Pedrycz, W., *Fuzzy Control and Fuzzy Systems*, John Wiley & Sons, Inc., New York, N.Y., 1989.

[23] Pedrycz, W., "Inverse Problem in Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 36, pp. 277-291, 1990.

[24] Pedrycz, W., "Processing in Relational Structures: Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 40, pp. 77-106, 1991(a).

[25] Pedrycz, W., "Fuzzy Modeling: Fundamental, Construction and Evaluation", *Fuzzy Sets and Systems*, Vol. 41, pp. 1-15, 1991(b).

[26] Peng, X.-T, Tu, X.-C., "Studies on Fuzzy PID Controllers", *BUSEFAL*, Part I: Vol. 29, pp. 81-89, Part II: Vol. 30, pp. 81-92, 1987.

[27] Peng, X.-T., Liu, S.-M., Yamakawa, T., Wang, P., Liu, X., "Self-Regulating PID Controllers and its Applications to a Temperature Controlling Process", In: *Fuzzy Computing*, Gupta, M.M., Yamakawa, T. (Editors), Elsevier Science Publishers B.V. (North-Holland), pp. 355-364, 1988.

[28] Qiao, W.Z., Zhuang, W.P., Heng, T.H., Shan, S.S., "A Rule Self-Regulating Fuzzy Controller", *Fuzzy Sets and Systems*, Vol. [illegible]7, pp. 13-21, 1992.

[29] Sanchez, E., "Resolution of Eigen Fuzzy Sets Equations", *Fuzzy Sets and Systems*, Vol. 1, pp. 69-74, 1978.

[30] Sessa, S., "Some Results in the Setting of Fuzzy Relation Equations Theory", *Fuzzy Sets and Systems*, Vol. 14, pp. 281-297, 1984.

[31] Shao, S., "Fuzzy Self-Organizing Controller and its Application for Dynamic Processes", *Fuzzy Sets and Systems*, Vol. 26, pp. 151-164, 1988.

[32] Shaw, I.S., Krüger, J.J., "New Fuzzy Learning Model with Recursive Estimation for Dynamic Systems", *Fuzzy Sets and Systems*, Vol. 48, pp. 217-229, 1992.

[33] Song, J.J., Park, W., "A Fuzzy Dynamic learning Controller for Chemical Process Control", *Fuzzy Sets and Systems*, Vol. 54, pp. 121-133, 1993.

[34] Tang, K.L., Mulholland, R.J., "Comparing Fuzzy Logic with Classical Controller Designs", *IEEE Transactions on Systems, Man and Cybernetics*, Vol., SMC-17, pp. 1085-1087, 1987.

[35] Thomason, M.G., "Convergence of Powers of a Fuzzy Matrix", *Journal of Mathematical Analysis and Applications*, Vol. 57, pp. 476-480, 1977.

[36] Tong, R.M., "Analysis of Fuzzy Control Algorithms using the Relation Matrix", *International Journal of Man-Machine Studies*, Vol. 8, pp. 679-686, 1976.

[37] Tong, R.M., "Analysis and Control of Fuzzy Systems using Finite Discrete Relations", *International Journal of Control*, Vol. 27, pp. 431-440, 1978.

[38] Tong, R.M., "Some Properties of Fuzzy Feedback Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC-10, pp. 327-330, 1980.

[39] Tong, R.M., "An Annotated Bibliography of Fuzzy Control", In: *Industrial Applications of Fuzzy Control*, Sugeno, M. (Editor), Elsevier Science Publishers B.V. (North-Holland), Amsterdam, pp. 249-269, 1985.

[40] Xu, C.-W., Lu, Y.-Z., "Fuzzy Model Identification and Self-Learning for Dynamic Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol., 17, pp. 683-689, 1987.

[41] Xu, C.-W., "Analysis and Feedback/Feedforward Control of Fuzzy Relational Systems", *Fuzzy Sets and Systems*, Vol. 35, pp. 105-113, 1990.

CHAPTER 7 *FUZZY* IDENTIFICATION [1,2]

"We cannot learn without changing and we cannot change without learning"

[Kosko, 1992]

7.1 Introduction

Fuzzy models are appropriate in situations where goals, constraints and physical mechanisms of the process can not be clearly defined deterministically. Yet even in these environments, a fuzzy system must learn the relationship between the input and output data.

Fuzzy identification usually involves the formation of a set of control rules or fuzzy implications, as well as, the generation of the membership functions that will be used. These tasks are accomplished either manually by interviewing knowledgeable process operators, or automatically by means of relational-based equations which utilize fuzzification procedures and referential fuzzy sets. This latter method offer a systematic design procedure for the construction of fuzzy models while avoiding the problems associated with fuzzy control rules, such as consistency, optimum number, interactivity and the need for a intelligent human operator [Shaw *et al.*, 1992]. Identification methodologies, utilizing fuzzy relational equations, are discussed in this chapter.

7.2 The *Fuzzy* Model Identification Process

Consider the fuzzy identification problem:

*Given fuzzy input, **u** and fuzzy output, **y**,*

*find **R** such that:* $\mathbf{y} = \mathbf{u} \circ \mathbf{R}$ (7.1)

Fuzzy relational identification problems are solved in a manner similar to deterministic identification problems, once the data has been assigned to representative referential sets and fuzzy discretization has been completed. Fuzzy identification consists of the following steps:

---

[1] A version of this Chapter has been accepted for publication. Bourke M.M., Fisher, D.G., 1995. (*Fuzzy Sets and Systems*, 74: 225-236).

[2] A version of this Chapter has been submitted for publication. Bourke M.M., Fisher, D.G., 1995. (*Fuzzy Sets and Systems*).

***Fuzzification of Data*:**

(i) define the universes of discourse for the input-output variables

(ii) determine referential fuzzy sets and the shape of the membership functions

(ii) express all the I/O data in terms of the referential fuzzy sets.

***Model Identification*:**

(iv) define the model structure (e.g. order, composition, etc.)

(v) calculate the parameters of the fuzzy relational matrix

(vi) validate the model

Each of these points is discussed in the following six subsections.

7.2.1 Universe of Discourse

The universe of discourse defines the entire space or range for each of the input-output variables. For N pairs of input-output data, $\mathbf{u}_k$ and $\mathbf{y}_k$, the universes are defined such that:

$$\mathbf{u}_k \in \mathcal{U} \quad \text{and} \quad \mathbf{y}_k \in \mathcal{Y} \qquad \text{for all } k = 1, 2, \ldots N. \tag{7.2}$$

For the discussions in this Chapter, a *fuzzy* variable, $\boldsymbol{u}$, is *italicized* and a discrete variable, $\mathbf{u}$, is not.

7.2.2 Referential Fuzzy Sets and Membership Functions

The choice of the number of referential fuzzy sets and the shape of the associated membership functions can be determined statistically, by clustering, or subjectively. There is no evidence to show that any analytic or computational method is better than a subjective decision [Xu *et al.*, 1987]. However, to ensure the performance of the fuzzy model the fuzzy membership functions should be normal, convex and satisfy the completeness condition outlined next.

The partitioning of the universe of discourse, $\mathcal{U}$, (or $\mathcal{Y}$) into referential sets, $U_1, U_2, \ldots, U_{\rho_u}$, (or $Y_1, Y_2, \ldots, Y_{\rho_y}$) must be such that the *complete* space is *covered*, that is;

$$\forall\, \mathbf{u}_k \in \mathcal{U} \;\; \exists\, 1 \leq i \leq \rho_u \;\; \ni U_i(\mathbf{u}_k) > 0 \tag{7.3}$$

for all $k = 1, 2, \ldots N$. This ensures that each point of $\mathcal{U}$ has a finite (non-zero) membership in at least one of the U_i's. The output space $\mathcal{Y}$ can be defined in a similar manner.

Let ρ be the number of referential fuzzy sets for any given universe of discourse. The number of referential fuzzy sets, ρ, for a universe of discourse, should be selected by weighing model accuracy against computational efficiency. Model accuracy may be improved by increasing ρ, however, a large ρ requires more computer memory and CPU time. Pedrycz [1984(a)] showed that increasing the number of referential sets indefinitely does not necessarily continue to improve the solution and there appears to be an optimum number. However, it seems intuitive that increasing the number of referential fuzzy sets infinitely ($\rho \to \infty$) should result in fuzzy singularities and discrete control. The choice of the number of referential fuzzy sets depends on the developer/user of the fuzzy model. The number should be chosen consistent with the number of linguistic labels which represent the level of knowledge available for the system being modeled.

7.2.3 Fuzzification of the Data

If the input data is not inherently fuzzy (i.e. it is discrete), then *fuzzification* of the data is necessary. *Fuzzification* is defined as the 1-to-ρ mapping whereby a real value is converted into ρ fuzzy values [Shaw *et al.*, 1992]. So during fuzzification discrete data is transformed via the referential fuzzy set, into membership values as defined by the corresponding membership functions. This is accomplished by calculating the degree of membership of each $\mathbf{u}_k \in \mathcal{U}$ in each of the referential fuzzy sets $U_1, U_2, \ldots, U_{\rho_u}$.

$$\mu_i(\mathbf{u}) = f(U_i(\mathbf{u})) = u_i \qquad \text{for } i = 1, 2, \ldots, \rho_u \tag{7.4}$$

The fuzzified value of $\mathbf{u}$, defined u (*italicized*), is the vector:

$$u = [u_1, u_2, u_3, \ldots, u_{\rho_u}] \tag{7.5}$$

7.2.4 Model Structure

The structure of the relational fuzzy model is normally defined *a priori* by imposing the form of the fuzzy relational equation, the order, p, the system delay, τ, and fixing the composition, ∘, resulting in a state space design. The benefit of fuzzy relational models is that the state-space methodology permits numerical systems analysis.

A *fuzzy* state space model was outlined in detail in Chapter 3, Section 3.3. The following is a short summary. Let $u=\{u_i \mid i = \{1,2, \ldots, m\}\} \in \mathcal{U}$, $x=\{x_j \mid j = \{1,2, \ldots, n\}\} \in \mathcal{X}$, and $y=\{y_l \mid l = \{1,2, \ldots, n\}\} \in \mathcal{Y}$ be the *fuzzy* spaces of input, state and output, respectively, all defined on the finite fuzzy universes of discourses indicated. Then for a series of N state, output and control data points, the first order fuzzy state space relationship with delay is written:

$$x_k = u_{k-\tau-1} \circ u_{k-\tau-2} \circ \ldots \circ u_{k-\tau-p} \circ x_{k-1} \circ x_{k-2} \circ \ldots \circ x_{k-p} \circ R \tag{7.6}$$

$$y_k = x_k \circ S \tag{7.7}$$

where $x_k, x_{k-1}, \ldots, x_{k-p}$ are the fuzzy states at the time instances indicated,

$u_{k-\tau-1}, u_{k-\tau-2}, u_{k-\tau-p}$ are the fuzzy control at the time instances indicated,

y_k is the fuzzy output at time k.

τ is the system delay,

p is the order of the system

and $\circ \in o$, where o stands for a family of composition operators

(i.e. *max-min*, *max-product*, etc.)

By assuming that the fuzzy states and the fuzzy outputs are the same, (i.e. $x_k = y_k$ for all k), S is reduced to an identity matrix, with $S(x_j, y_l) = 1$ if $x_j = y_l$ and 0 otherwise. Therefore equation (7.7) can be ignored and the fuzzy p-th order state space equation is as follows:

$$y_k = u_{k-\tau-1} \circ u_{k-\tau-2} \circ \ldots \circ u_{k-\tau-p} \circ y_{k-1} \circ y_{k-2} \circ \ldots \circ y_{k-p} \circ R \tag{7.8}$$

7.2.5 Parameter Calculation

The optimization problem for parameter estimation can be formulated as follows:

$$\min_{\substack{\tau \geq 0 \\ \circ \in O \\ R}} Q(\tau, \circ, R) = Q(\tau_{opt}, \circ_{opt}, \tilde{R}_{opt}) \tag{7.9}$$

where $\circ$ stands for a family of composition operators (i.e. *max-min*, *max-product*, etc.). Q stands for the sum of the distances between the respective fuzzy sets,

$$Q = \sum_{k=\tau+2}^{N} \sum_{l} | \tilde{y}_k - y_k |^q \tag{7.10}$$

where

$$\tilde{y}_k = u_{k-\tau-1} \circ x_{k-1} \circ \tilde{R}$$

$$y = \{y_l \mid l = \{1, 2, \ldots, n\}\} \tag{7.11}$$

and q is an arbitrary integer power. For $q = 1$, $\boldsymbol{Q}$ is referred to as the Hamming distance, for $q = 2$, $\boldsymbol{Q}$ is referred to as the Euclidean distance.

Parameter estimation is done through *batch* identification runs with a sample of input-output data to determine $\tilde{\boldsymbol{R}}$ with systematic combinations of τ and p. The values of $\boldsymbol{Q}$ are then plotted to determine the *best* combination (i.e. lowest value of $\boldsymbol{Q}$) for the application being considered.

For any identification the *quality* of the model prediction is of key importance. The goal in parameter selection is to incorporate enough parameters so that the model is flexible enough to describe different system behaviour, and yet not so over parameterized so that the system is costly to run and maintain. In short, the system should be minimized with respect to the model structure [Graham *et al.*, 1988].

7.2.6 Model Validation

Once a model, $\tilde{\boldsymbol{R}}$, has been calculated from the input-output data, (as described in Section 7.3 - Identification Algorithms), the model must be validated. As with discrete systems, this validation is accomplished by comparing predicted output data, based on the model developed, against actual data. Whether the actual output data is inherently *fuzzy* or discrete, the model validation procedure is similar. However, if the *fuzzy* predicted output data is be compared against discrete actual output data, then the *fuzzy* predicted output data must be *defuzzified.*

7.2.6.1 Defuzzification

Defuzzification is defined as the ρ-to-1 mapping where the ρ fuzzy values are converted into a single discrete output value [Shaw *et al.*, 1992]. *Defuzzification* for relational systems is accomplished in the same manner as discussed in Chapter 2 for rule-based systems.

While *fuzzification* and the choice of *fuzzy membership functions* tends to be subjective, several authors have shown the benefits of several *defuzzification* procedures [Tong 1978(a); Xu. *et al.* 1987; Mizumoto 1989]. Tong [1978(a)] and Xu. *et al.* [1987] compare *defuzzification methodologies* which operate on the overall consolidated output membership function, while Mizumoto [1989] compared overall consolidated output defuzzification as well as methods that deal with each output membership function separately.

Tong [1978(a)] compared the defuzzification methods of average of maxima and median. The results of this testing showed that the median method always gives a lower mean square error than the average of maxima method. This result is due to the fact that the median method generates a less extreme output estimate, so that it will give smaller errors when the model is not accurate.

Xu *et al.* [1987] compare the three defuzzification methods; average of maxima, median and center of gravity. The results showed that for the majority of the simulations conducted the median method produced the best results, followed by the center of gravity. The average of maxima method consistently produced *poorer* results than the other two methodologies. These results are consistent with Tong [1978(a)].

Mizumoto [1989] showed that the height method and the area method obtained better control results than the center of gravity method, which is widely used in fuzzy control. The height and area methods produced results similar to the median method, shown as the preferred method by Tong [1978(a)] and Xu *et al.* [1987].

7.2.6.2 Minimization Criteria

Once the predicted and output data are in a form such that they can be compared, a minimization criterion must be selected. If the data is inherently fuzzy, then the minimization criterion is generally of the form:

$$Q_q = \frac{\sum_{k=\tau+2}^{N} \sum_{l} |y_k - \tilde{y}_k|^q}{N - \tau - 1} \tag{7.12}$$

where $q = 1$ or 2

For $q = 1$, Pedrycz [1991(b)] and Valente de Oliveira [1993] provided a neural learning strategy for this minimization problem for the *max-min* and *max-product* composition.

For $q = 2$, Pedrycz [1983] provides a quasi-Newton method to solve this minimization problem for a *max-min* and a *max-product* composition. Wang [1993] presents the properties of the quasi-Newton method for several compositions, but does not include the *max-product* composition in this analysis. Ikoma et al. [1993] propose the probabilistic descent method for the minimization problem with $q = 2$, for the *max-min* composition, only.

If the actual output data is not inherently fuzzy, the predicted output data must be *defuzzified* and then compared against the actual as follows:

$$J_q = \frac{\sum_{k=\tau+2}^{N} |y_k - \tilde{y}_k|^q}{N - \tau - 1} \tag{7.13}$$

where $q = 1, 2$

These minimization procedures will be discussed in more detail in Chapter 8 along with some of the problems associated with these methodologies.

7.3 The Challenge of *Fuzzy* Identification

Given, N input-output data points and keeping the idea of *parsimony* in mind, *do not use more parameter that necessary* [Ljung, 1987], the fuzzy state space relational model, $\tilde{R}$, will be learned or identified by a first order system, $p = 1$, with a time delay, τ .

$$y_k = u_{k-\tau-1} \circ y_{k-1} \circ R \tag{7.14}$$

As with deterministic systems, the first order model with time delay structure of equation (7.14) is normally a good approximation for higher order overdamped processes. The problem with fuzzy relational equations, is that the size of the relational matrix, R, grows exponentially with the order of the model, as does execution time. So for systems with long sampling intervals (i.e. > 30 seconds), higher order systems, (i.e. $p > 1$) can be considered because there is enough time available at each control interval to do the extensive calculations. But with higher order systems, there are extensive storage requirements. Chen *et al.* [1994] have outlined an algorithm to reduce the storage requirements for such systems.

There is an important difference between the on-line identification of a fuzzy scheme and that of a deterministic or stochastic scheme [Graham *et al.*, 1988]. With deterministic processes, an initial model is proposed and as the identification algorithm proceeds iteratively the parameters of the model normally become increasingly accurate. The important point is that at each time step the model is complete, although inaccurate. Therefore the currently available deterministic model can always be used to make predictions which are more or less accurate.

Fuzzy identification usually starts with a completely empty relational matrix from which it is impossible to make any predictions. At each iteration entries are added to the model which are essentially accurate but which cover only a portion of the input-output space. So, at any particular time step, the model may not be complete and so outputs will only be able to be obtained for certain inputs. This is not a desirable feature when using fuzzy models in a control scheme and this problem will be addressed later in this chapter.

Some of the concerns with *fuzzy relational identification* are:

(*i*) a large number of input-output points must usually be considered

The main concern with *fuzzy relational matrices* is model completeness. *Fuzzy relational identification* requires a large number of input-output points that must cover the entire operating range. Thus the larger the range the greater the number of input-output points that must be processed in order to complete the learning. *Fuzzy models* are unable to extrapolate their learning from one point of the operating range to another. The *fuzzy model* must learn from experience.

(*ii*) discontinuous data or periods of non-representative data arise in practical applications

Similar to discrete identification, discontinuous data or periods of non-representative data do not offer valid experience to the *fuzzy model*. For batch learning these periods can simply be omitted. For on-line learning, the effect of this data will be *forgotten* with time.

(*iii*) input or output variables may saturate

Input or output variable saturation provides the opportunity to confirm the limits of the operating range. However, extended periods of saturation offer little additional model information.

A good deal of the literature pertaining to fuzzy identification deals with fuzzy systems in a *batch* sense [Czogala and Pedrycz, 1981; Pedrycz, 1984(a); Pedrycz, 1985; Pedrycz, 1987; Pedrycz, 1988]. It is assumed that their is a large sample of input-output data available and various techniques are applied to this data to determine the relationship between the two. Recently some new results [Shaw *et al.*, 1992; Chen *et al.*, 1994; Bourke *et al.*, 1995(b)] show that a recursive or *self-learning* algorithm can produce comparatively better results than the batch algorithms previously proposed. The Box-Jenkins [1970] gas furnace data is widely used for testing *fuzzy identification algorithms*, [Tong, 1980; Pedrycz, 1984(a); Xu et.al, 1987; Ridley *et al.*, 1988, Xu, 1989; Sugeno *et al.*, 1991; Shaw *et al.*, 1992; Valente de Oliveira, 1993] so it is fairly easy to rank the various identification algorithms based on this benchmark.

7.3.1 The Exact Solution

Again consider the first order time delay model, equation (7.14).

$$y_k = u_{k-\tau-1} \circ y_{k-1} \circ R \tag{7.14}$$

If every ordered triple, $(u_{k-\tau-1}, y_{k-1}, y_k)$, for $k = 1, 2, \ldots, N$, satisfies equation (7.14), without error, then the solution is considered exact and the greatest fuzzy relation, $\hat{R}$, which satisfies this equation, as proposed by Di Nola *et al.* [1984], is given by :

$$\hat{R} = \bigcap_{k=\tau+2}^{N} [(u_{k-\tau-1} * y_{k-1}) \circledast y_k] \tag{7.15}$$

where $*$ is $\cdot$ for the *max-product* composition and *min* for the *max-min* composition.

and $\circledast$ is the ψ-composition for *max-product* composition and the α-composition for *max-min* composition

As shown by Pedrycz [1990] and discussed in the next section, this conclusion is flawed.

7.3.2 Approximate Solutions

The fuzzy relation $\tilde{R}$ resulting from the input-output data available may be such that not all of the input-output triples satisfy the model equation proposed. It is very difficult to obtain a system of equations by a purely analytical calculation such that an *exact* overall solution exists because of:

- model mismatch (i.e. the model structure differs from the system structure)
- choice of composition operator (i.e. operators may not infer the solution appropriately)
- corrupt data (i.e. by noise or other disturbances)

Data inconsistencies that effect the solution to a system of equations have two forms:

(*i*) individual input-output data triples do not satisfy the condition for the existence of a solution

(*ii*) each individual input-output triple satisfies the condition for solution existence, but the intersection of all the solutions is empty

Pedrycz [1988] has shown for the *max-min* composition and then later [1991(a)] for the *max-t-norm* composition, the necessary and sufficient conditions for the set $\mathcal{R} \neq \emptyset$, for the case of the input-output data *pair*.

Considering data inconsistency, point (*i*), Proposition 1 provides the necessary and sufficient condition for the existence of a solution for the data *triple* of a first order model.

Proposition 1: For $u=\{ u_i \mid i = \{1,2, ..., m\}\}$, $x=\{ x_j \mid j = \{1,2, ..., n\}\}$, and $y=\{ y_l \mid l = \{1,2, ..., n\}\}$, let $s_{ij} = (u_i * x_j)$, where $*$ is *product* for the *max-product* composition and *min* for the *max-min* composition.

Then for all $\mathcal{R} = \{R \mid y = u \circ x \circ R\}$, the necessary and sufficient condition for the set $\mathcal{R} \neq \emptyset$ is:

$$\exists\, i \in \{1, 2, \ldots, m\}, \quad j \in \{1, 2, \ldots, n\}$$

$$\ni \max_i(\max_j(s_{ij})) \geq \max(y_l) \quad \forall\, l \in \{1, 2, \ldots, n\} \tag{7.16}$$

Or in terms of u_i, x_j and y_l, $\exists\, i \in \{1, 2, \ldots, m\}$, $j \in \{1, 2, \ldots, n\}$ $\ni$:

for *max-product* $$[\max(u_i) \cdot \max(x_j)] \geq \max(y_l) \quad \forall\, l \in \{1, 2, \ldots, n\} \tag{7.17}$$

for *max-min* $$\min[\max(u_i), \max(x_j)] \geq \max(y_l) \quad \forall\, l \in \{1, 2, \ldots, n\} \tag{7.18}$$

Proof:

By definition of the inverse functions, ψ, for *max-product* and α, for *max-min*

$$\begin{aligned} &\text{if } s_{ij} \leq y_l && \text{for all } i, j \text{ and } l \\ &\text{then } r_{ijl} = 1 && \text{for all } i, j \text{ and } l \end{aligned} \tag{7.19}$$

and no solution exists. Thus it is enough that just one element of s_{ij} is greater than or equal to the maximum element of y_l for a solution to exist.

Now considering data inconsistency point (*ii*), each individual input-output triple satisfies the condition of solution existence but the intersection of all the solutions is empty. This concept has been demonstrated by [Pedrycz, 1990] as follows.

It has been proven that if the intersection of all the families of equations forms a non empty set:

$$\boldsymbol{R} = \bigcap_{k=1}^{N} \boldsymbol{R}_k \neq \varnothing \tag{7.20}$$

then its *greatest* element is:

$$\hat{\boldsymbol{R}} = \bigcap_{k=1}^{N} \hat{\boldsymbol{R}}_k \tag{7.21}$$

where $\hat{\boldsymbol{R}}_k$ is the *greatest* element of each k-th pair of input-output data. This intersection gives the *minimal* relational matrix from the *maximal* elements of $\boldsymbol{R}_k$.

According to Pedrycz [1990], the above statement contains a very strong assumption. It can be easily seen that the case might exist for one pair of input-output data ($\boldsymbol{u}_q$, $\boldsymbol{y}_q$) such that the solution set $\boldsymbol{R}_q$ is non empty, but that the intersection with all the remaining families $\boldsymbol{R}_k$, $k \neq q$, is empty. That is:

$$\boldsymbol{R}' = \bigcap_{k \neq i}^{N} \boldsymbol{R}_k \neq \varnothing, \qquad \text{but} \quad \boldsymbol{R}_q \bigcap \boldsymbol{R}' = \varnothing \tag{7.22}$$

So if the entire system of equations is treated together, it has no solution. Additionally, the possibility of this increases as the number of input-output pairs (N) grows [Pedrycz, 1988]. In general the condition that:

$$\bigcap_{k \neq i}^{N} \boldsymbol{R}_k \neq \varnothing \tag{7.23}$$

can not be expected to be completely fulfilled. However, the information obtained from equation (7.23), in these situations, should not be considered as totally irrelevant, but may infact form an approximate solution for the set of equations and is therefore valuable [Pedrycz, 1991(c)].

Although the procedure, as described by equation (7.15) provides *good* results [Pedrycz, 1984(a)], based on a minimization criteria such as Hamming distance or least squares, for exact and consistent data, it encounters problems when considering inexact data. The major problem lies in the fact that the *intersection* of the *maximum* solution or the *union* of the *minimum* solutions are unable to recover or unlearn erroneous information. Thus an averaging procedure, such as the one proposed by Shaw *et al.* [1992] and discussed in the next section, provides a method of *forgetting* erroneous data.

Another procedure has been recently presented by Chen *et al.* [1994], which updates the fuzzy model on-line by adding new entries to the fuzzy relational matrix while gradually deleting old unreasonable entries.. The method, presented by these authors, permits the unlearning of previous data by taking its complement. This procedure will also be reviewed in the next section.

Analyzing these situations, one should consider eliminating inconsistent data during calculations to determine a suitable relational matrix, $\boldsymbol{R}$. This situation has been addressed by several authors who have attempted to define inconsistent data and then detect it and measure or eliminate its effect on the solution of $\boldsymbol{R}$. Gottwald [1985] and Gottwald *et al.* [1986] have studied the solvability property of a system of fuzzy relational equations. They have proposed a solvability index on [0,1] which is an indication of how well a pair of input-output data contribute to a consistent solution, with 1 being completely solvable. So if those equations which lead to extremely low values of the solvability index are removed, a consistent solution would exist. This index is useful not only as a means of detecting situations where there is no solution, but also as a measure of how *easily* fuzzy systems of equations or a single fuzzy equation can be solved. However, this work represents a *passive* solution in that it evaluates the solvability property, but it does not give an indication of the structure of the fuzzy sets analyzed nor does it provide a means of data modification in order to ensure a solution.

Pedrycz [1985] has proposed a structured fuzzy model where the fuzzy relation is associated with a probability matrix which specifies the structure of the system and summarizes the frequencies of the links between the elements of the fuzzy relation. This same author [1988] investigated the structure of the system of equations with a consistent subset of the input-output data used to calculate the solution of the entire system. In this work the author introduces a feasibility index which measures the similarity of the pairs of input-output data and indicates the degree to which the system of equations can be solved. This index is not equivalent to the solvability index, proposed previously, but it is related to it. In [1990] Pedrycz proposed two algorithms to construct a fuzzy relational matrix from inconsistent data. The first method gathers statistics concerning the distribution of results for an α-composition (or α-cut) of the elements of the fuzzy input-output pairs and then looks for consistent pairs. The second method uses the results of the α-composition and then transforms them into a probabilistic set. The compositional operator is then extended to handle both the fuzziness and the probability. Wagenknecht *et al.* [1986, 1987] propose an interval-valued fuzzy set where intervals on the membership grades of the input-output are considered as tolerances and the fuzzy relational matrix is constructed with tolerances or an applicable interval is provided for each entry.

But is removal of the inconsistent data a valid procedure in fuzzy modeling? Unlike deterministic process, where the noise and unmeasurable disturbances are usually estimated separated from the process model, the fuzzy relational matrix of a fuzzy process incorporates not only the process but noise and disturbances as well. Therefore removing input-output data that is considered *corrupted* may rob the fuzzy model of required process information.

Pedrycz *et al.* [1981] have overcome the problem of inconsistent data by applying the concept of probabilistic sets. Pedrycz [1983], Ikoma *et al.* [1993], Pedrycz [1991(b)] and Valente de Oliveira [1993] all suggest an optimization approach to solve fuzzy relational equations. The optimization procedure proposed consists of finding a fuzzy relational matrix which minimizes a specified performance index. These numerical approaches will be considered in more detail in Chapter 8.

7.3.3 Estimate of Minimum Solution

Both *exact* and *approximate* solutions have been considered in this Chapter. Now consider an *estimated* solution, or more specifically, an *estimate of a minimum* solution. For this discussion, let the fuzzy relation $\boldsymbol{R}=\{((u_i, y_j), r_{ij}) \mid i = \{1,2, \ldots, m\}; j = \{1,2, \ldots, n\}\}$ be the mapping $\boldsymbol{R}$: $\mathcal{U}\times\mathcal{Y} \rightarrow [0,1]\times[0,1]$ where $r_{ij} \in [0,1]$ is the grade of membership between (u_i, y_j). Let $\boldsymbol{u} \in \mathcal{U}$ and $\boldsymbol{y} \in \mathcal{Y}$ be the fuzzy sets $\boldsymbol{u}=\{ u_i \mid i= \{1,2, \ldots, m\}\}$ and $\boldsymbol{y}=\{ y_j \mid j = \{1,2, \ldots, n\}\}$ where u_i, and $y_j \in [0,1]$ are the grades of membership in $\boldsymbol{U}_i$, and $\boldsymbol{Y}_j$, respectively. Now for a given series of N input-output data pairs, $(\boldsymbol{u}_k, \boldsymbol{y}_k)$, consider the identification problem of finding $\boldsymbol{R} \in \mathcal{U}\times\mathcal{Y}$ such that:

$$\boldsymbol{y}_k= \boldsymbol{u}_k \circ \boldsymbol{R} \qquad k = 1, 2, \ldots, N \tag{7.24}$$

where ∘ represents the *max-min* composition.

Baboshin *et al.* [1990] have shown, for the *max-min* composition, that to minimize the Hamming distance between the actual, $\boldsymbol{y}$, and the estimate, $\tilde{\boldsymbol{y}}$:

$$Q(y, \tilde{y}) = \sum_{j=1}^{n} |y_j - \tilde{y}_j| \tag{7.25}$$

the identification of $\boldsymbol{R}$ should be through an estimate of the *minimum* inverse calculation rather than by the Mamdani identification operator [Mamdani, 1974]. These authors present a method of determining the *smallest* fuzzy relation $\bar{R}$ for the case when u is a *normal* fuzzy set (i.e. $\max(u_i) = 1$, for $i=1,\ldots, m$). The results are then applied to a multi-dimensional fuzzy system.

The condition of a *normal* u presents a problem when actual process input is considered since there is no guarantee that any given input, u_k, will be *normal*. Therefore the work by Baboshin *et al.* [1990] was extended, to the case when the input data, u_k, is *unrestricted* and can therefore be *non-normal*, and then to the case of *unrestricted* input data, u_k, for the *max-product* composition.

As a brief review for the development of this theory, the following definitions are provided.

Definition 1: The *greatest* fuzzy relation $\hat{R} \in \mathcal{R}$ such that $\mathcal{R} = \{R \in \mathcal{U}\times\mathcal{Y} \mid y = u \circ R\} \neq \varnothing$ is given by:

$$\hat{R} = u^t \textcircled{\alpha} y \tag{7.26}$$

where $\textcircled{\alpha}$ is defined for the *max-min* composition as:

$$\hat{r}_{ij} = u_i \,\alpha\, y_j = \begin{cases} y_j & \text{if} \quad u_i > y_j \\ 1 & \text{otherwise} \end{cases} \tag{7.27}$$

and for the *max-product* composition as:

$$\hat{r}_{ij} = u_i \,\alpha\, y_j = \begin{cases} {}^{y_j}\!/_{u_i} & \text{if} \quad u_i > y_j \\ 1 & \text{otherwise} \end{cases} \tag{7.28}$$

Definition 2: The *union* of all the *smallest* fuzzy relations $\bar{R} \in \mathcal{R}$ such that $\mathcal{R} = \{R \in \mathcal{U}\times\mathcal{Y} \mid y = u \circ R\} \neq \varnothing$ is given by:

$$\bar{R} = u^t \textcircled{\sigma} y \tag{7.29}$$

where $\textcircled{\sigma}$ is defined for the *max-min* composition as:

$$\bar{r}_{ij} = u_i \,\sigma\, y_j = \begin{cases} y_j & \text{if} \quad u_i \geq y_j \\ 0 & \text{otherwise} \end{cases} \tag{7.30}$$

and for the *max-product* composition as:

$$\bar{r}_{ij} = u_i \sigma y_j = \begin{cases} y_j / u_i & \text{if} \quad u_i \geq y_j \\ 0 & \text{otherwise} \end{cases} \tag{7.31}$$

Definition 3: The Mamdani identification method for $\boldsymbol{R}^M \in \mathcal{R}$ such that $\mathcal{R} = \{\boldsymbol{R} \in \mathcal{U} \times \mathcal{Y} \mid \boldsymbol{y} = \boldsymbol{u} \circ \boldsymbol{R}\} \neq \varnothing$ is given by:

$$\boldsymbol{R}^M = \boldsymbol{u}^t \times \boldsymbol{y} \tag{7.32}$$

where $\times$ is defined for the *max-min* composition as:

$$r_{ij}^M = \min(u_i, y_j) = \begin{cases} y_j & \text{if} \quad u_i \geq y_j \\ u_i & \text{otherwise} \end{cases} \tag{7.33}$$

Although Mamdani did not work in the domain of *max-product* it is standard practice to assume the cartesian product identification for the *max-product* domain to be the *product* operator. Therefore $\times$ is defined for the *max-product* composition as:

$$r_{ij}^M = u_i \cdot y_j \qquad \text{for all } i \text{ and } j \tag{7.34}$$

The operation proposed by Baboshin *et al.* [1990] which determines a *smallest* fuzzy relation $\tilde{\boldsymbol{R}}$, with the condition that $\boldsymbol{u}$ is a *normal* fuzzy set, is summarized as follows:

Theorem 1: Given $\boldsymbol{u} \in \mathcal{U}$ and $\boldsymbol{y} \in \mathcal{Y}$, then the *smallest* fuzzy relation $\tilde{\boldsymbol{R}} \in \mathcal{R}$, where $\mathcal{R} = \{\boldsymbol{R} \in \mathcal{U} \times \mathcal{Y} \mid \boldsymbol{y} = \boldsymbol{u} \circ \boldsymbol{R}\} \neq \varnothing$, with $\boldsymbol{u}$ a *normal* fuzzy set and meeting the condition $\tilde{r}_{ij} \leq r_{ij}$ is specified by the operation:

$$\tilde{r}_{ij} = u_i \bar{\sigma}_n y_j = \begin{cases} y_j & \text{if} \quad u_i = 1 \\ 0 & \text{otherwise} \end{cases} \tag{7.35}$$

Theorem 1 can be extended to all fuzzy sets, *normal* or *non-normal* as follows:

Theorem 2: Given $\boldsymbol{u} \in \mathcal{U}$ and $\boldsymbol{y} \in \mathcal{Y}$, then a *smallest* fuzzy relation $\tilde{\boldsymbol{R}} \in \mathcal{R}$, where $\mathcal{R} = \{\boldsymbol{R} \in \mathcal{U} \times \mathcal{Y} \mid \boldsymbol{y} = \boldsymbol{u} \circ \boldsymbol{R}\} \neq \emptyset$, with $\boldsymbol{u}$ an *unrestricted* (i.e. *normal* or *non-normal*) fuzzy set and meeting the condition $\tilde{r}_{ij} \leq r_{ij}$ is specified by the operation:

$$\tilde{r}_{ij} = u_i \,\bar{\sigma}_u\, y_j = \begin{cases} y_j & \text{if} \quad u_i = \max(\boldsymbol{u}) \\ 0 & \text{otherwise} \end{cases} \tag{7.36}$$

Proof:

(*i*) Prove that $\boldsymbol{y} = \boldsymbol{u} \circ \tilde{\boldsymbol{R}}$:

$$\begin{aligned} y_j &= u_i \circ r_{ij} \\ y_j &= \max_i (\min(u_i, r_{ij})) \\ &= \max_{i:x_i = \max(x)} (\min(u_i, r_{ij})) \vee \max_{i:x_i \neq \max(x)} (\min(u_i, r_{ij})) \\ &= \max_{i:x_i = \max(x)} (\min(u_i, y_j)) \\ &= y_j \end{aligned}$$

This proves that $\boldsymbol{u} \circ \tilde{\boldsymbol{R}} = \boldsymbol{y}$ and so $\tilde{\boldsymbol{R}} \in \mathcal{R}$.

(*ii*) Prove that $\tilde{\boldsymbol{R}} \subseteq \boldsymbol{R}$ or $\tilde{r}_{ij} \leq r_{ij}$:

Assume that there exists an $\boldsymbol{R}' \in \mathcal{R}$ such that $\boldsymbol{R}' \subset \tilde{\boldsymbol{R}}$, then:

$$\begin{aligned} & r'_{ij} < \tilde{r}_{ij} & \\ \text{but} \quad & \tilde{r}_{ij} = y_j & \text{for } i: u_i = \max(\boldsymbol{u}) \\ \therefore \quad & r'_{ij} < y_j & \text{for } i: u_i = \max(\boldsymbol{u}) \end{aligned}$$

If $\boldsymbol{R}'$ exists then:

$$\begin{aligned} y'_j &= u_i \circ r'_{ij} \\ y'_j &= \max_i (\min(u_i, r'_{ij})) \\ &= \max_{i:x_i = \max(x)} (\min(u_i, r'_{ij})) \\ &= \max_{i:x_i = \max(x)} (r'_{ij}) \\ &< y_j \end{aligned}$$

Since $y'_j \neq y_j$ then $\boldsymbol{R}' \notin \mathcal{R}$ which contradicts the original premise. So $\tilde{\boldsymbol{R}} \subseteq \boldsymbol{R}$ and $\tilde{\boldsymbol{R}}$ is the *smallest* fuzzy relation.

Theorem 2 can be extended to the *max-product* domain as follows:

Theorem 3: Given $\boldsymbol{u} \in \mathcal{U}$ and $\boldsymbol{y} \in \mathcal{Y}$, then a *smallest* fuzzy relation $\tilde{\boldsymbol{R}} \in \mathcal{R}$, where $\mathcal{R} = \{\boldsymbol{R} \in \mathcal{U} \times \mathcal{Y} \mid \boldsymbol{y} = \boldsymbol{u} \circ \boldsymbol{R}\} \neq \emptyset$, with $\boldsymbol{u}$ an *unrestricted* (i.e. *normal* or *non-normal*) fuzzy set and meeting the condition $\tilde{r}_{ij} \leq r_{ij}$ is specified by the operation:

$$\tilde{r}_{ij} = u_i \,\bar{\sigma}_u\, y_j = \begin{cases} y_j / u_i & \text{if} \quad u_i = \max(\boldsymbol{u}) \\ 0 & \text{otherwise} \end{cases} \tag{7.37}$$

Proof:

(*i*) Prove that $\boldsymbol{y} = \boldsymbol{u} \circ \tilde{\boldsymbol{R}}$:

$$\begin{aligned} y_j &= u_i \circ r_{ij} \\ y_j &= \max_i \,(u_i \cdot r_{ij}) \\ &= \max_{i:x_i = \max(x)} (u_i \cdot r_{ij}) \vee \max_{i:x_i \neq \max(x)} (u_i \cdot r_{ij}) \\ &= \max_{i:x_i = \max(x)} (u_i \cdot r_{ij}) = \max_{i:x_i = \max(x)} \left(u_i \cdot \frac{y_j}{u_i}\right) \\ &= y_j \end{aligned}$$

This proves that $\boldsymbol{u} \circ \tilde{\boldsymbol{R}} = \boldsymbol{y}$ and so $\tilde{\boldsymbol{R}} \in \mathcal{R}$.

(*ii*) Prove that $\tilde{\boldsymbol{R}} \subseteq \boldsymbol{R}$ or $\tilde{r}_{ij} \leq r_{ij}$:

Assume that there exists an $\boldsymbol{R}' \in \mathcal{R}$ such that $\boldsymbol{R}' \subset \tilde{\boldsymbol{R}}$, then:

$$r'_{ij} < \tilde{r}_{ij}$$

but $\tilde{r}_{ij} = y_j$ for i: $u_i = \max(\boldsymbol{u})$

$\therefore$ $r'_{ij} < y_j$ for i: $u_i = \max(\boldsymbol{u})$

If $\boldsymbol{R'}$ exists then:

$$
\begin{aligned}
y'_j &= u_i \circ r'_{ij} \\
y'_i &= \max_i (u_i \cdot r'_{ij}) \\
&= \max_{i:x_i=\max(x)} (u_i \cdot r'_{ij}) \\
&= \max_{i:x_i=\max(x)} (r'_{ij}) \\
&< y_j
\end{aligned}
$$

Since $y'_j \neq y_j$ then $\boldsymbol{R'} \notin \mathcal{R}$ which contradicts the original premise. So $\tilde{\boldsymbol{R}} \subseteq \boldsymbol{R}$ and $\tilde{\boldsymbol{R}}$ is the *smallest* fuzzy relation.

The conditions for the existence of an inverse solution to the identification problem given by:

$$y = \boldsymbol{u} \circ \boldsymbol{R} \tag{7.38}$$

has been shown by Pedrycz [1988] to be:

$$\max_i (u_i) \geq \max_j (y_j) \tag{7.39}$$

This result is the precursor to Proposition 1, provided earlier. When this condition, which is the same for both *max-min* and *max-product*, is satisfied the corresponding relational matrices are assured of a solution.

When a system is being identified, the efficiency of the identification algorithm is as important a factor as the accuracy. Thus, if the relational matrix identification, based on an estimate calculation, results in superior performance, and is also computationally more efficient, it should be considered for on-line identification. It has been shown by Baboshin *et al.* [1990] that the estimate of the *minimum*, $\tilde{\boldsymbol{R}}$, determined from Theorem 1, with $\boldsymbol{u}$ *normal*, will ensure the minimum Hamming distance and is superior to the Mamdani identification operation. The questions to be considered now are: (*i*) will these results extend to the more general *unrestricted* case, as outlined in Theorem 2, (*ii*) is there another formulation that can produce *better* results using the same performance criterion, and (*iii*) can the performance results be generalized from the *fuzzy* domain to the *discrete* (or *defuzzified*) domain.

It is known from the properties of t-norms that:

$$(i) \quad \text{Given } u \circ \left(\bigcap_{k=1}^{N} R_k^{'}\right) = y' \text{ and } u \circ \left(\bigcap_{k=1}^{N} R_k^{''}\right) = y'' \tag{7.40}$$

$$\text{if } R' \subseteq R'' \tag{7.41}$$

$$\text{then } y' \leq y''$$

$$(i \quad \text{en } u \circ \left(\bigcup_{k=1}^{N} R_k^{'}\right) = y' \text{ and } u \circ \left(\bigcup_{k=1}^{N} R_k^{''}\right) = y'' \tag{7.42}$$

$$\text{if } R' \subseteq R'' \tag{7.43}$$

$$\text{then } y' \leq y''$$

Consider the definitions of R_k^M, $\tilde{R}_k$, $\hat{R}_k$ and $\overline{R}_k$ for the k-th input-output pair, and the condition for the existence of a solution, it is easy to see that:

$$\tilde{R}_k \subseteq \overline{R}_k \subseteq R_k^M \subseteq \hat{R}_k \tag{7.44}$$

Let R be the actual relational matrix that generates the output data, as defined in equation (7.24).

$$y_k = u_k \circ R \qquad \text{for } k = 1, \ldots, N. \tag{7.24}$$

Then for each k-th pair of data points:

$$\tilde{R}_k \subseteq R \subseteq \hat{R}_k \tag{7.45}$$

It is unknown at this point what the exact relationship is between R, $\tilde{R}_k$ and R_k^M.

Now consider the definitions of $\overline{R}_k$ and $\hat{R}_k$ for a single input-output data pair. For a particular matrix location for these inverses, $\overline{r}_{ij}$ is the *maximal* element and $\hat{r}_{ij}$ is the *minimal* element. But when $x_i > y_j$ then $\overline{r}_{ij} = \hat{r}_{ij}$. The existence criteria require that there is at least one occurrence of $x_i \geq y_j$ for all j, so for each input-output pair there are locations in each of the *maximum* and *minimum* matrices where the values are equal. Since the global $\overline{R}$ is determined by taking the *maximum* over all the $\overline{R}_k$'s and the global $\hat{R}$ is determined by taking the *minimum* over all the $\hat{R}_k$'s, as the number of input-output pairs, or N, approaches infinity, and the occurrence of $x_i > y_j$ becomes adequate to cover the range of the relational matrix, and:

$$\lim_{N\to\infty}\bigcup_{k=1}^{N}\overline{R}_k = \lim_{N\to\infty}\bigcap_{k=1}^{N}\hat{R}_k \tag{7.46}$$

An equally importance factor, which guarantees the equality of equation (7.46), is that for a series of data pairs, the calculation of each k-th *minimum* is a function of the *maximum* inverse for the series of N data pairs, (i.e. the global $\hat{R}$). The calculation of $\overline{R}$ is illustrated in Example 1 of Sessa [1984].

Now consider the global picture, that is the union or *maximum*, over all N data pairs, for $\tilde{R}_k$, R_k^M, and $\overline{R}_k$, and the intersection or *minimum* for $\hat{R}_k$. Since the global value of $\hat{R}$ bounds the values of $\overline{R}_k$, the global $\overline{R}$ is bounded, and therefore the order of the global form of these matrices changes from that given by equation (7.44). Based on the results of equation (7.46), clearly

$$\bigcup_{k=1}^{N}\overline{R}_k \subseteq \bigcap_{k=1}^{N}\hat{R}_k \tag{7.47}$$

Since there is no limit on the maximum values of $\bigcup_{k=1}^{N}\tilde{R}_k$, other than the maximum of 1 place on the fuzzy domain, these values, for large N, can exceed $\bigcap_{k=1}^{N}\hat{R}_k$, so the ordering of the global matrices is:

$$\bigcup_{k=1}^{N}\overline{R}_k \subseteq \bigcap_{k=1}^{N}\hat{R}_k \subseteq \bigcup_{k=1}^{N}\tilde{R}_k \subseteq \bigcup_{k=1}^{N}R_k^M \tag{7.48}$$

and

$$\overline{y} \leq \hat{y} \leq \tilde{y} \leq y^M \tag{7.49}$$

The theoretical development, made thus far, is based on a series of equations with exact solutions. Thus $\bigcap_{k=1}^{N}\hat{R}_k$ still represents the *maximum* value for the series so that the actual matrix, R, can be positioned such that $R \subseteq \bigcap_{k=1}^{N}\hat{R}_k$ and equation (7.48) as can be expanded as follows:

$$R \subseteq \bigcup_{k=1}^{N}\overline{R}_k \subseteq \bigcap_{k=1}^{N}\hat{R}_k \subseteq \bigcup_{k=1}^{N}\tilde{R}_k \subseteq \bigcup_{k=1}^{N}R_k^M \tag{7.50}$$

and

$$y \leq \overline{y} \leq \hat{y} \leq \tilde{y} \leq y^M \tag{7.51}$$

Clearly, $\bigcup_{k=1}^{N} \boldsymbol{R}_k^M$ and $\bigcup_{k=1}^{N} \tilde{\boldsymbol{R}}_k$ are outside the range of valid $\boldsymbol{R}$'s, if $\bigcap_{k=1}^{N} \hat{\boldsymbol{R}}_k$ is considered the true *maximum*, and as such will be unable to predict representative output values, y_k. However, the results of Baboshin *et al.* [1990] extend to the *unrestricted* $\boldsymbol{u}$, in that:

$$y \leq \tilde{y} \leq y^M \tag{7.52}$$

and

$$| y - \tilde{y} | \leq | y - y^M | \tag{7.53}$$

Thus, the *unrestricted* estimate of the *minimum* relational matrix will ensure a minimum Hamming distance compared to the Mamdani identification operation.

Now consider the calculations for estimation of the *minimum* and the *maximum* relational matrices. According to equation (7.50), $\boldsymbol{R} \subseteq \hat{\boldsymbol{R}} \subseteq \tilde{\boldsymbol{R}}$, so by equation (7.51), $y \leq \hat{y} \leq \tilde{y}$. Thus the *maximum* relational matrix identification should produce a Hamming distance smaller than either the estimate of the *minimum* or the Mamdani identification.

This same analysis holds true for the *max-product* composition, except for the Mamdani equivalent. From the definition of this operation, it is easy to see that for each k-th input-output pair:

$$\boldsymbol{R}_k^M \subseteq \hat{\boldsymbol{R}}_k \tag{7.54}$$

However, no conclusion can be made between $\boldsymbol{R}_k^M$ and either of $\tilde{\boldsymbol{R}}$ or $\overline{\boldsymbol{R}}_k$. As well, no conclusion can be made between $\bigcap_{k=1}^{N} \hat{\boldsymbol{R}}_k$ and $\bigcup_{k=1}^{N} \boldsymbol{R}_k^M$.

The analysis above has considered only input-output pairs. It can however be expanded to the first order state space equation. The following definitions describe the solution algorithms.

Definition 4: The *greatest* fuzzy relation $\hat{\boldsymbol{R}} \in \mathcal{R}$ such that $\mathcal{R} = \{\boldsymbol{R} \in \mathcal{U} \times \mathcal{X} \times \mathcal{Y} \mid \boldsymbol{y} = \boldsymbol{u} \circ \boldsymbol{x} \circ \boldsymbol{R}\} \neq \emptyset$ is given by:

$$\hat{\boldsymbol{R}} = (\boldsymbol{u}^t * \boldsymbol{x}) \circledast \boldsymbol{y} \tag{7.55}$$

where $*$ is *min* for *max-min* and *product* for *max-product*

and $\circledast$ is defined for the *max-min* composition as:

$$\hat{r}_{ijl} = (u_i \wedge x_j)\ \alpha\ y_l = \begin{cases} y_l & \text{if} \quad (u_i \wedge x_j) > y_l \\ 1 & \text{otherwise} \end{cases} \tag{7.56}$$

and for the *max-product* composition as:

$$\hat{r}_{ijl} = (u_i \cdot x_j)\ \alpha\ y_l = \begin{cases} y_l \big/ (u_i \cdot x_j) & \text{if} \quad (u_i \cdot x_j) > y_l \\ 1 & \text{otherwise} \end{cases} \tag{7.57}$$

Definition 5: The *union* of all the *smallest* fuzzy relations $\overline{\boldsymbol{R}} \in \mathcal{R}$ such that $\mathcal{R} = \{\boldsymbol{R} \in \mathcal{U} \times \mathcal{X} \times \mathcal{Y} \mid \boldsymbol{y} = \boldsymbol{u} \circ \boldsymbol{x} \circ \boldsymbol{R}\} \neq \varnothing$ is given by:

$$\overline{\boldsymbol{R}} = (\boldsymbol{u}' * \boldsymbol{x}) \circledcirc \boldsymbol{y} \tag{7.58}$$

where $*$ is *min* for *max-min* and *product* for *max-product*

and $\circledcirc$ is defined for the *max-min* composition as:

$$\overline{r}_{ijl} = (u_i \wedge x_j)\ \overline{\sigma}\ y_l = \begin{cases} y_l & \text{if} \quad (u_i \wedge x_j) \geq y_l \\ 0 & \text{otherwise} \end{cases} \tag{7.59}$$

and for the *max-product* composition as:

$$\overline{r}_{ijl} = (u_i \cdot x_j)\ \sigma\ y_l = \begin{cases} y_l \big/ (u_i \cdot x_j) & \text{if} \quad (u_i \cdot x_j) \geq y_l \\ 0 & \text{otherwise} \end{cases} \tag{7.60}$$

Definition 6: The Mamdani identification method for $\boldsymbol{R}^M \in \mathcal{R}$ such that $\mathcal{R} = \{\boldsymbol{R} \in \mathcal{U} \times \mathcal{X} \times \mathcal{Y} \mid \boldsymbol{y} = \boldsymbol{u} \circ \boldsymbol{x} \circ \boldsymbol{R}\} \neq \varnothing$ is given by:

$$\boldsymbol{R}^M = \boldsymbol{u} \times \boldsymbol{x} \times \boldsymbol{y} \tag{7.61}$$

where $\times$ is defined for the *max-min* composition as:

$$r_{ijl}^M = \min(u_i, x_j\, y_l) = \begin{cases} y_l & \text{if} \quad (u_i \wedge x_j) \geq y_l \\ (u_i \wedge x_j) & \text{otherwise} \end{cases} \tag{7.62}$$

and $\times$ is defined as the cartesian product for the *max-product* composition as:

$$r_{ijl}^{M} = u_i \cdot x_j \cdot y_l \qquad \text{for all } i, j \text{ and } l \tag{7.63}$$

Definition 7: The *estimate* of the *minimum* fuzzy relations $\tilde{\boldsymbol{R}} \in \mathcal{R}$ such that $\mathcal{R} = \{\boldsymbol{R} \in \mathcal{U} \times \mathcal{X} \times \mathcal{Y} \mid \boldsymbol{y} = \boldsymbol{u} \circ \boldsymbol{x} \circ \boldsymbol{R}\} \neq \varnothing$ is given by:

$$\tilde{\boldsymbol{R}} = (\boldsymbol{u}^t * \boldsymbol{x}) \circledS \boldsymbol{y} \tag{7.64}$$

where $*$ is *min* for *max-min* and *product* for *max-product*

and $\circledS$ is defined for the *max-min* composition as:

$$\tilde{r}_{ijl} = (u_i \wedge x_j)\, \bar{\sigma}\, y_l = \begin{cases} y_l & \text{if} \quad (u_i \wedge x_j) = \bigvee_{i,j}(u_i \wedge x_j) \\ 0 & \text{otherwise} \end{cases} \tag{7.65}$$

and for the *max-product* composition as:

$$\tilde{r}_{ijl} = (u_i \cdot x_j)\, \bar{\sigma}\, y_l = \begin{cases} y_l / (u_i \cdot x_j) & \text{if} \quad (u_i \cdot x_j) = \bigvee_{i,j}(u_i \cdot x_j) \\ 0 & \text{otherwise} \end{cases} \tag{7.66}$$

With the addition of Proposition 1, the existence condition for a solution, it is clear that the ordering of the identified matrices is the same as for the input-output pair case, for both *max-min* and *max-product*. Again the Mamdani identification for the *max-product* case exhibits the same problems in the first order state space domain as it did previously.

7.4 Literature Review of Identification Algorithms

As a means of leading into the development of a new identification algorithm presented as part of this thesis, this section reviews and compares several current batch and on-line identification algorithms that have been presented in the literature.

It appears to be customary in the fuzzy literature [Tong, 1980; Pedrycz, 1984(a); Xu *et al.*, 1987; Ridley *et al.*, 1988; Sugeno *et al.*, 1991; Shaw *et al.*, 1992; Valente de Oliveira, 1993] to test and compare ones algorithm with the Box-Jenkins gas furnace data [1970]. In keeping with this precedent, the identification algorithms reviewed and the algorithm developed in this thesis are tested using the Box-Jenkins [1970] data. The reviewed works are implemented to determine the computational requirements of the algorithm and the fact that different referential fuzzy sets and different defuzzification techniques, used by the various authors, can substantially change the output results. Thus the identification results based on these algorithms can be compared using the same basis.

7.4.1 The Comparison Basis

The Box-Jenkins gas furnace data [1970] consists of 296 pairs of input-output measurements. The input, $\mathbf{u}_k$, is the gas flow rate into the furnace and the output, $\mathbf{y}_k$, is the CO_2 concentration in the outlet gas. The sampling interval is 9 seconds.

Since the Box-Jenkins gas furnace data [1970] is discrete, it must be *fuzzified*. This requires the partitioning of the input and output space such that the *complete* space is *covered*. Since partitioning of the data is subjective [Pedrycz, 1983; Xu *et al.*, 1987], the 296 input-output data points were plotted and the data fit to a line in a least squares sense. From the plot a one-to-one mapping was determined based on this line of *best* fit. Subjectively, $\rho = 5$ referential fuzzy sets were chosen for each of the input, state and output data, since a 5^3 (= 125) relational matrix provided adequate definition of the data and is computationally expedient. The shape of the referential fuzzy sets was chosen to be isosceles triangles, which are easy to work with and which satisfy entropy equalization criteria [Pedrycz, 1994] as well as the requirements of being normal and convex. These fuzzy membership functions were distributed over the input-output space, via the blocks illustrated in Figure 7.1, so that the space was completely covered and the referential sets had a 50% overlap, which produces a zero value reconstruction error [Pedrycz, 1994]. The final input and output fuzzy referential sets are shown in Figure 7.2, (a) and (b), respectively.

The *defuzzification* formulation was chosen to be the weighted average center of area method because it uses all the information in the *fuzzy* output vector, and other than the fact that this *defuzzification* formula involves a division, and is therefore non-linear, its is one the simplest computationally.

$$y = \frac{\sum_i c_i \cdot \mu_i}{\sum_i \mu_i} \tag{7.67}$$

where y is the defuzzified output

c_i is the center of area of the i-th reference fuzzy set

μ_i is the degree of membership in the i-th reference fuzzy set

The *fuzzy* solutions will be compared by calculating the average deviation per point, shown in equation (7.12), while *defuzzified* solutions will be compared using equation (7.13), both for q = 1 or 2.

$$\mathcal{L}_q = \frac{\sum_{k=\tau+2}^{N}\sum_{l}|y_k - \tilde{y}_k|^q}{N-\tau-1} \tag{7.12}$$

$$J_q = \frac{\sum_{k=\tau+2}^{N}|y_k - \tilde{y}_k|^q}{N-\tau-1} \tag{7.13}$$

where $\tilde{y}$ and $\tilde{y}$ are the fuzzy and non-fuzzy (or discrete) output estimates, respectively.

The time delay is chosen to be $\tau = 3$ for both *max-min* and *max-product* identifications as per previous work [Pedrycz, 1984(a); Xu *et al.*, 1987; Shaw *et al.* 1992]. This choice is contrary to some evidence [Pedrycz, 1984(a)] that $\tau = 2$ may improve the *max-product* solution, but consistency between the *max-min* and *max-product* composition is preferred for comparison between the two operations.

To fully analyze the capability of relational matrix identification methodologies, the relational matrices are cross identified. That is, a matrix identification by *product* formulation is applied to the *max-min* composition and conversely, a matrix identified by *minimum* is applied to the *max-product* composition. This cross application also prevents the formation of rigid lines separating the identification of the two compositions, such as, *product* only with *max-product* and *min* only with *max-min*. The identification algorithm by Shaw *et al.* [1992] applies such a cross identification with improved results.

Normalization of the relational matrix is also considered as a method of improving the model results. A *normalized* relational matrix is calculated as follows.

$$\tilde{R}_t^n(u_i,x_j,y) = \tilde{R}_t(u_i,x_j,y) \Big/ \max(\tilde{R}_t(u_i,x_j,y)) \tag{7.68}$$

for each $i = \{1, 2, ..., m\}$ and $j = \{1, 2, ..., n\}$. An added feature of a *normalized* matrix is that if there has been sufficient learning of the relational matrix such that:

$$\sum_l \tilde{R}(u_i,x_j,y_l) \geq \varepsilon \tag{7.69}$$

for all i and j and an arbitrary $0 < \varepsilon < 1$, then *normalization* of the matrix ensures a *complete* matrix, since each u_i and x_i can determine a y_l.

The Box-Jenkins output data ranges in value from 45.6 to 60.5 %CO_2, with an expected value of 53.5 %. Therefore a value of $J_2 \leq 0.5$ would translate approximately to an average error of 1.3 %. From a process control point of view $J_2 \leq 0.5$ would be considered a *good* result. As the ultimate comparative reference, the Box-Jenkins [1970] **non-fuzzy, linear, time-series model**, has a value of $J_2 = 0.202$, when applied on-line using six predictor variables [Sugeno *et al.*, 1991], and a value of $J_2 = 0.71$, when applied in batch [Xu *et al.*, 1987]. Since J_2 is the discrete performance index used in papers that publi[illegible] value for a performance index [Tong, 1980; Pedrycz, 1984(a); Xu *et al.*, 1987; Ridley *et al.*, 1988; Sugeno *et al.*, 1991], it will be the index used here to judge the quality of the discrete performance of the various identification procedures.

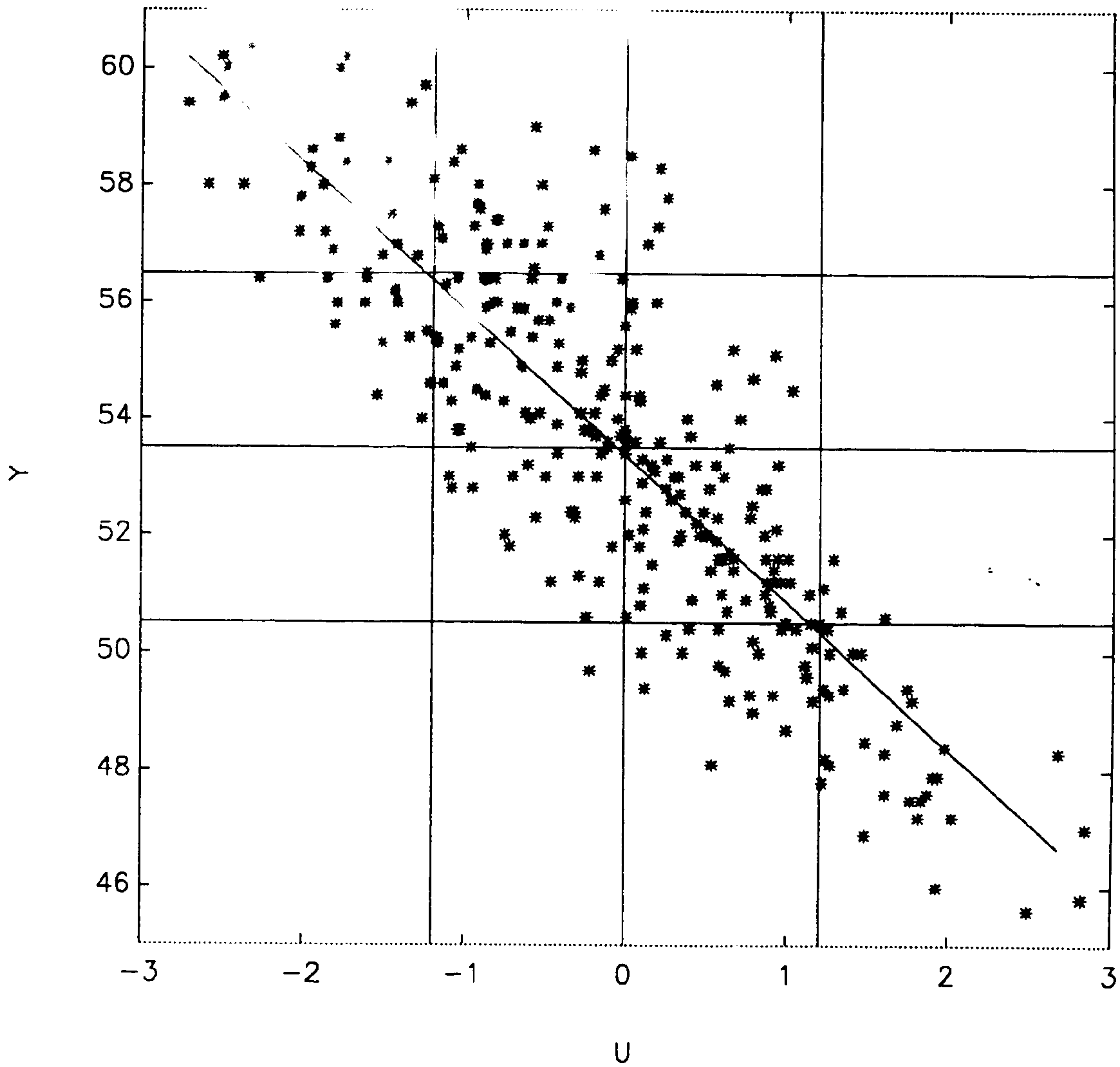


Figure 7.1: Input-Output Data Mapping

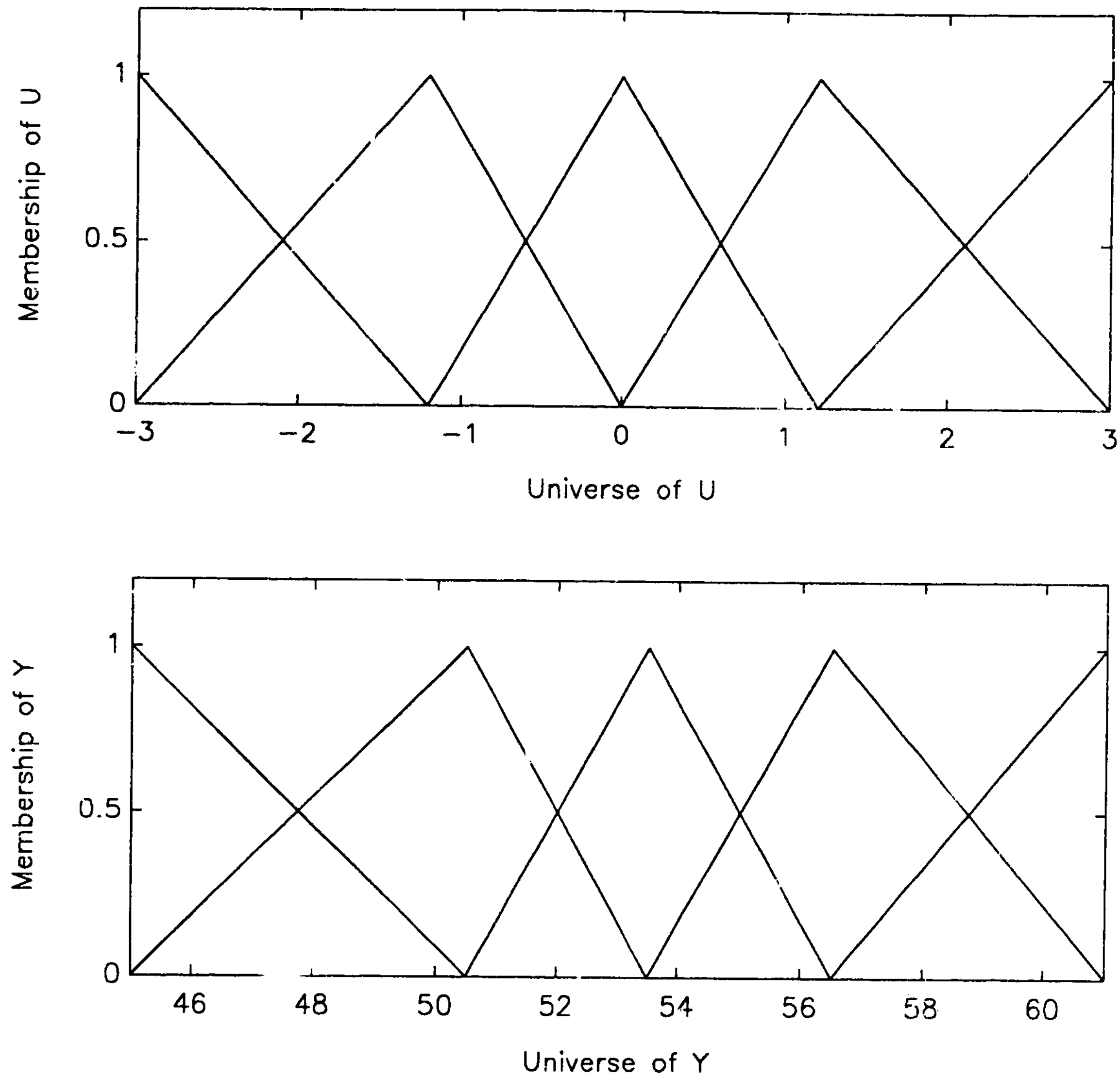


Figure 7.2: Referential Fuzzy Sets;
(a) Input Fuzzy Sets;
(b) Output Fuzzy Sets

7.4.2 The Identification Algorithm by DiNola *et al.* [1984]

The batch identification algorithm proposed by DiNola *et al.* [1984] is based on the inverse operation provided by Sanchez [1976].

$$\tilde{R} = \bigcap_{k=\tau+2}^{N} [(u_{k-\tau-1} * y_{k-1}) \circledast y_k] \tag{7.18}$$

where $*$ is *product* for the *max-product* composition and *min* for the *max-min* composition.

and $\circledast$ is the ψ-composition for *max-product* composition and the α-composition for *max-min* composition defined as:

Definition 8: For a and $b \in [0,1]$, the ψ-composition is defined as:

$$a \, \psi \, b = \begin{cases} 1 & \text{if} \quad a < b \quad \text{or} \quad a = b = 0 \\ b/a & \text{if} \quad a \geq b \end{cases} \tag{7.70}$$

Definition 9: For a and $b \in [0,1]$, the α-composition is defined as:

$$a \, \alpha \, b = \begin{cases} 1 & \text{if} \quad a < b \\ b & \text{if} \quad a \geq b \end{cases} \tag{7.71}$$

As noted in Section 7.3.2 there are two data inconsistencies that effect the solution when this equation is used. First, an exact solution does not exist if the data triples do not satisfy the condition of existence of a solution, and second, the intersection of valid solution sets may be empty [Pedrycz, 1990].

When the 296 data points of the Box-Jenkins data [1970], were tested for inconsistent data defined by equation (7.16), the condition for existence of a solution excluded **38** points for the *max-product* composition and **93** points for the *max-min* composition.

To address the problem of a solution existing for each instant of data, but the intersection of all solutions resulting in a empty solution, the following data screening algorithm was employed in conjunction with the model identification. The objective of the data screening was to prevent the identification algorithm from *minimizing* the relational matrices to an incomplete solution.

Define:

$$\tilde{R}_{k-1} = \bigcap_{t=\tau+2}^{k-1} [(u_{t-\tau-1} * x_{t-1}) \circledast y_t] \tag{7.72}$$

Calculate:

$$\tilde{R}'_k(u, x, y) = (u_{k-\tau-1} * x_{k-1}) \circledast y_k \tag{7.73}$$

$$\tilde{R}''_k(u, x, y) = \tilde{R}'_k \cap \tilde{R}_{k-1} \tag{7.74}$$

Equation (7.74) must be tested to ensure model completeness before it can be redefined as $\tilde{R}_k$. Model completeness is determined by summing the $\tilde{R}''_k(u, x, y)$ along the y axis.

$$\tilde{R}''_k(u, x) = \sum_l \tilde{R}''_k(u, x, y_l) \tag{7.75}$$

If any position of the $\tilde{R}''_k(u, x)$ matrix is zero, then the model will become incomplete if it is intersected or minimized with the new entry, $\tilde{R}'_k(u, x, y)$. For some small $\varepsilon \approx 0$, the data selection algorithm can be summarized:

If $\min_i (\min_i (\tilde{R}''_k(u_i, x_j))) > \varepsilon$

then $\tilde{R}_k = \tilde{R}''_k(u, x, y)$

else $\tilde{R}_k = \tilde{R}_{k-1}$

Testing with this algorithm resulted in the rejection of an additional **56** points for the *max-product* composition and **43** points for the *max-min* composition.

Eliminating inconsistent data and testing for an incomplete solution results in a *poor* solution with $J_2 = 1.078$ for the *max-product* composition and $J_2 = 1.775$ for the *max-min* composition, shown in Figure 7.3. When the DiNola *et al.* [1984] algorithm was tested only for an incomplete relational matrix, the prediction improved, with $J_2 = 0.9668$ for the *max-product* composition and $J_2 = 1.222$ for the *max-min* composition, shown in Figure 7.4. It should be noted that without testing for an incomplete relational matrix, the prediction from the relational matrix was unable to track the data.

Using all the inconsistent data points is considered a *brute force* method of identification [Pedrycz, 1987]. However, intuitively it would seem that it is better to use all the data since the relational model must learn both process information and disturbances.

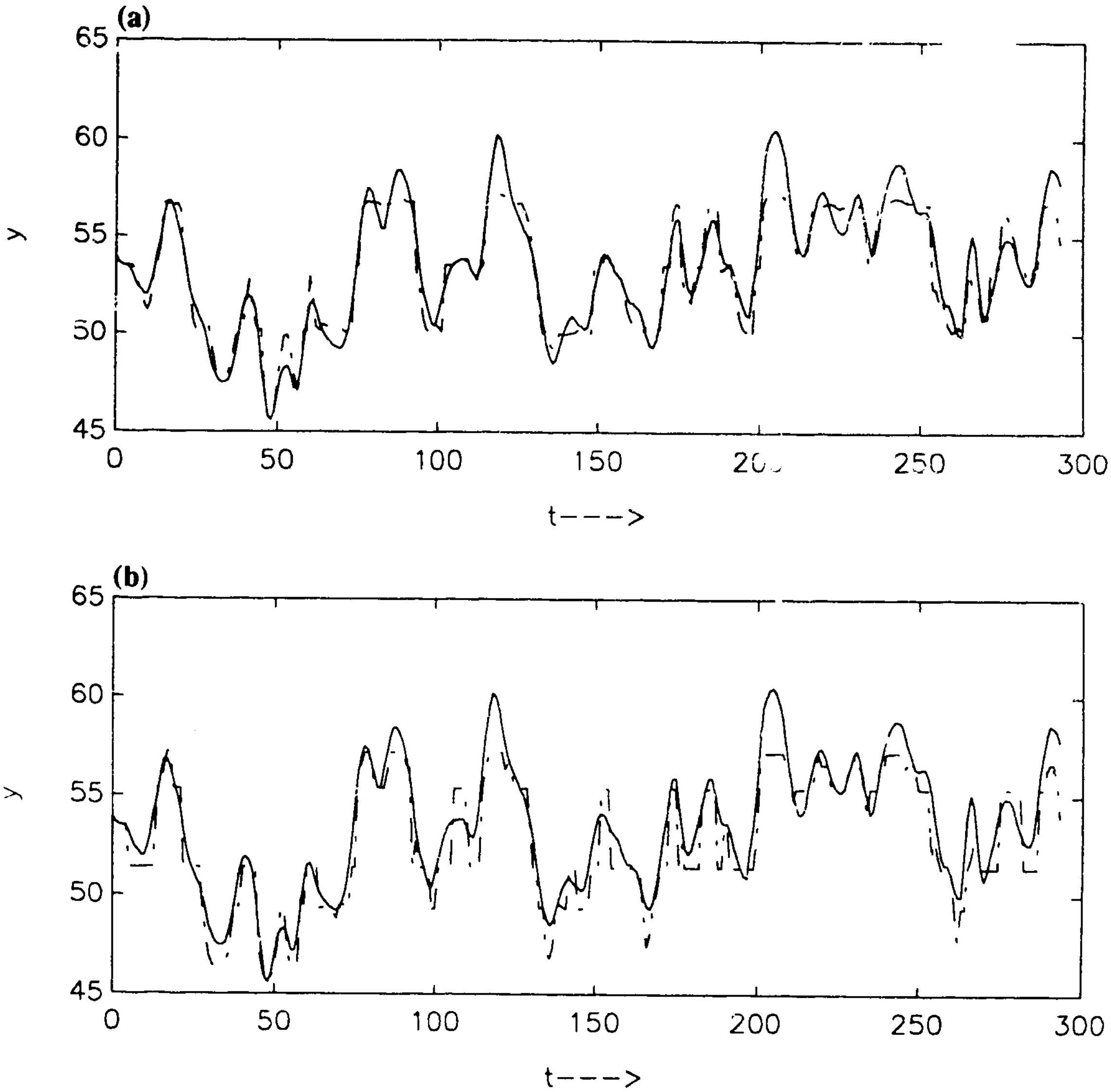
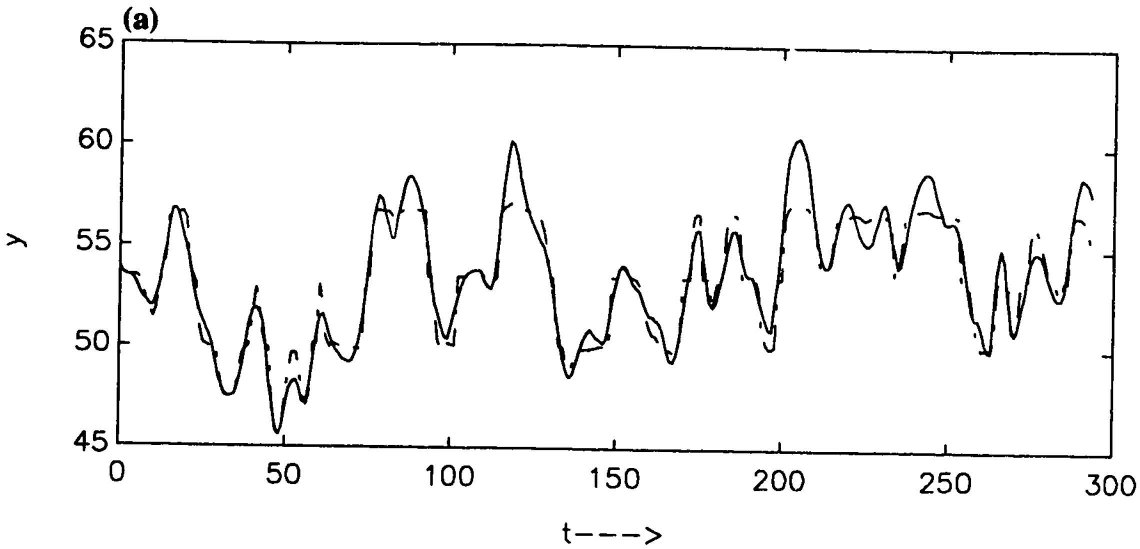


Figure 7.3: Batch ID for DiNola *et al.* [1984] Algorithm with Inconsistent Data and Zero Solution Adjustments;
(a) Max-Product: *R*(prod; reject (i) & (ii)), $J_1 = 0.7828$; $J_2 = 1.0780$; $Q_1 = 0.5804$; $Q_2 = 0.2427$;
(b) Max-Min: *R*(min; reject (i) & (ii)), $J_1 = 1.038$; $J_2 = 1.7750$; $Q_1 = 0.8752$; $Q_2 = 0.5361$;
(——— Actual; - - - - Model)



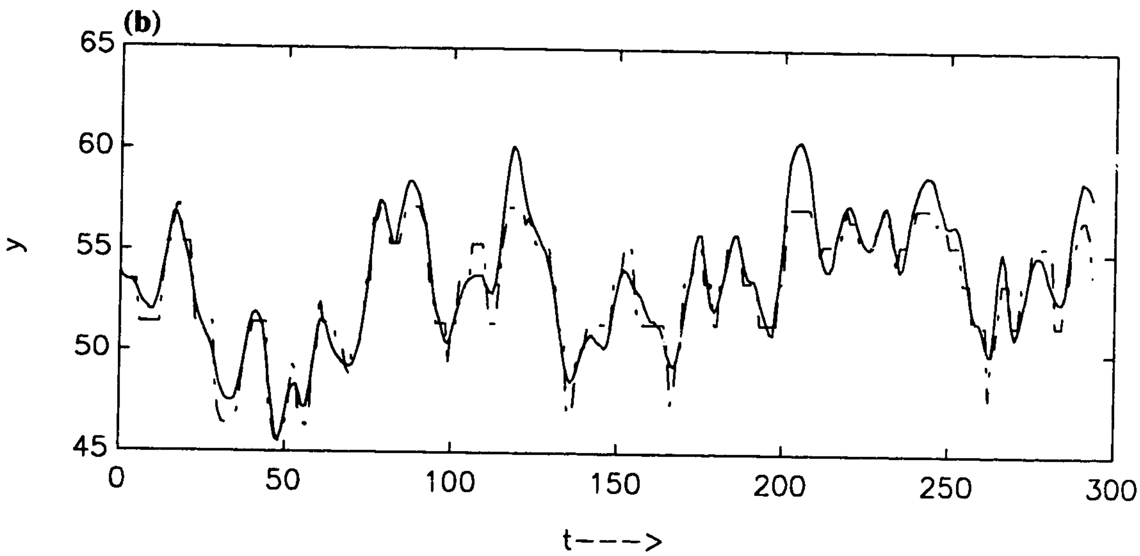


Figure 7.4: Batch ID for DiNola *et al.* [1984] Algorithm with Zero Solution Adjustment only; (a) Max-Product: R(prod; reject (ii)), $J_1 = 0.7321$; $J_2 = 0.9668$; $Q_1 = 0.6176$; $Q_2 = 0.2647$; (b) Max-Min: R(min; reject (ii)), $J_1 = 0.8522$; $J_2 = 1.2220$; $Q_1 = 0.8766$; $Q_2 = 0.5369$; (——— Actual; - - - - Model)

7.4.3 The Identification Algorithm by Pedrycz [1984(a)]

Pedrycz [1984(a)] presented a rather simple yet effective batch identification technique. For the first order delay process the fuzzy relational matrix, $\tilde{R}$, was calculated: by

$$\tilde{R} = \bigcup_{k=\tau+2}^{N} u_{k-\tau-1} \times x_{k-1} \times y_k \tag{7.76}$$

where $\times \equiv$ *product* for *max-product* composition

$\times \equiv$ *min* for *max-min* composition

The published results [1984(a)] for this algorithm for both a *max-min* and a *max-product* composition using the Box-Jenkins gas furnace data [1970] showed that for $n = m = 5$ ($\tilde{R}$ = 125 points) and $\tau = 2$, that the *max-product* composition was preferred with $J_2 = 0.776$. Increasing n and m decreased J_2 as follows; for $n = m = 7$, $J_2 = 0.478$ ($\tilde{R}$ = 343 points) and for $n = m = 9$, $J_2 = 0.320$ ($\tilde{R}$ = 729). Increasing n and m results in an exponential increase in the size of $\tilde{R}$, as well as processing time. Thus other methods of identification, as will be shown, should be considered before increasing the number of referential fuzzy sets in order to improve the solution.

Testing this algorithm with the referential sets as defined in Figure 7.2 show that for $n = m = 5$ and $\tau = 3$, $J_2 = 0.5450$ for *max-product* and $J_2 = 0.6378$ for *max-min*. Comparing this J_2 value for the *max-product* composition with that by Pedrycz [1984(a)] show that a simple change in the referential fuzzy sets resulted in an improved solution.

The cross application of the identification showed an improvement for *max-min* composition only ($J_2 = 0.4993$). No improvement in solution resulted for the *max-product* case ($J_2 = 0.5947$).

Now considering only the *product* generated relational matrices. Both output results improved if the relational matrices were *normalized*. By definition, the *product* operation produces results that are smaller than the *min* operation. *Normalizing* the relational matrix identified by *product* increases the values of the relational matrix which repeated learning has reduced. Predictions by these *normalized* relational matrices are also increased and offer an improved prediction as is evident for this example and several of the other examples that will be shown later. Thus for Pedrycz's [1984(a)] batch identification the *best* identification for both *max-product* ($J_2 = 0.3787$) and *max-min* ($J_2 = 0.4484$), shown in Figure 7.5, was from a $\tilde{R}$ identified by *product* formulation and then *normalized*. This same improvement is not evident for relational matrices identified with the *minimum*. A listing of the performance results for this algorithm are provided in the summary in Table 7.1.

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
Pedrycz [1984(a)]	*Prod*	*Max-prod*	0.5956	0.5450	0.5733	0.1558
		Max-min	0.5342	0.4993	0.6675	0.1603
R(non-normal)	*Min*	*Max-prod*	0.6293	0.5947	0.7742	0.2135
		Max-min	0.5899	0.6378	1.1060	0.3978
Pedrycz [1984(a)]	*Prod*	*Max-prod*	0.4813	0.3787	0.5395	0.1248
		Max-min	0.5155	0.4484	0.7514	0.2031
R(normal)	*Min*	*Max-prod*	0.5616	0.4796	0.7878	0.2186
		Max-min	0.6096	0.6837	1.1890	0.4531

Table 7.1: Identification Results of Pedrycz [1984(a)] Algorithm

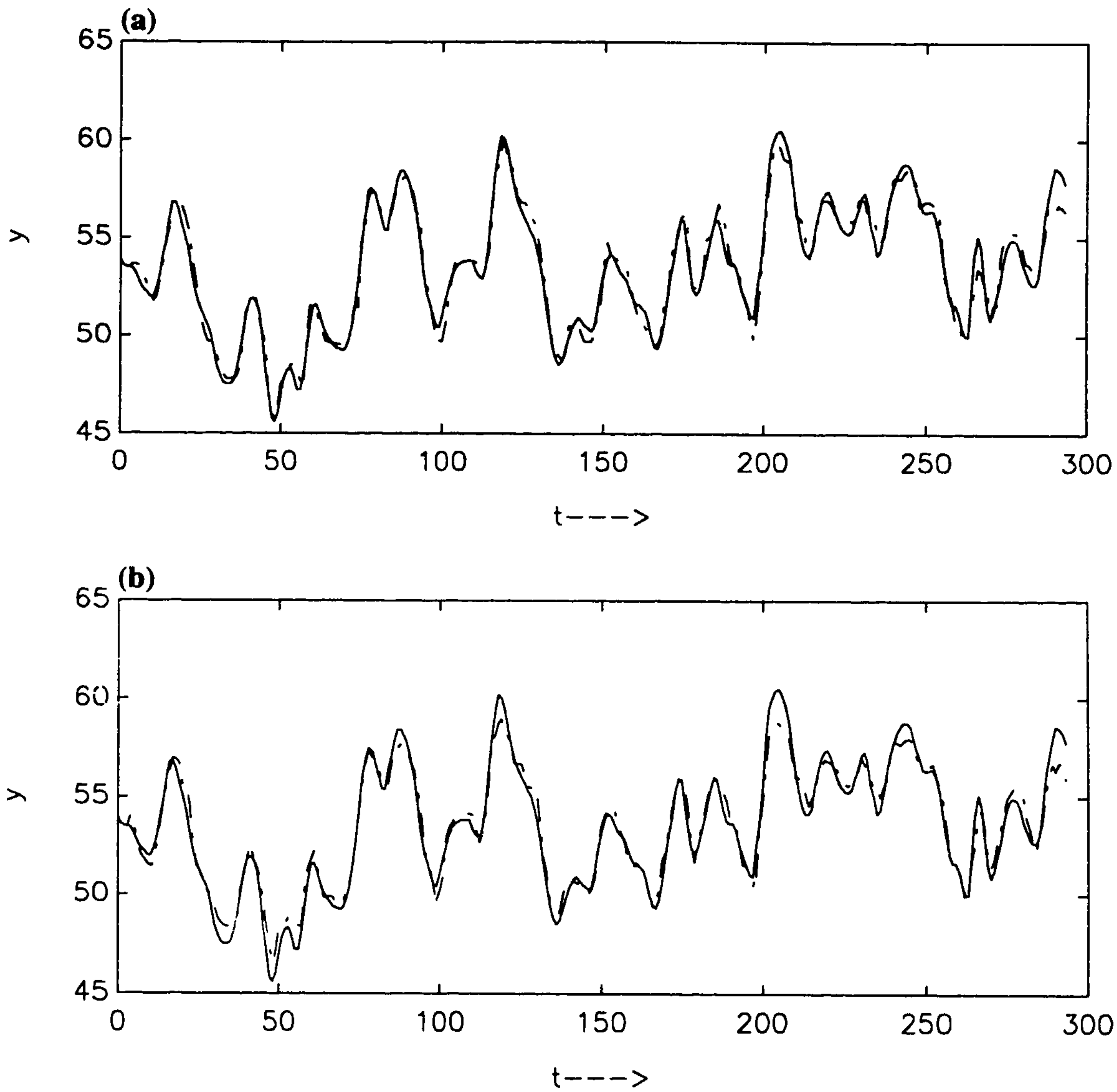


Figure 7.5: Batch ID for Pedrycz [1984(a)] Algorithm;
(a) Max-Product: *R*(prod; normal), $J_1 = 0.4813$; $J_2 = 0.3787$; $Q_1 = 0.5395$; $Q_2 = 0.1248$;
(b) Max-Min: *R*(prod; normal), $J_1 = 0.5155$; $J_2 = 0.4484$; $Q_1 = 0.7514$; $Q_2 = 0.2031$;
(——— Actual; - - - - Model)

7.4.4 The Identification Algorithm by Chen *et al.* [1994]

This section considers an *on-line* adaptation of the *off-line* identification algorithm by Pedrycz [1984(a)]. The algorithm presented by Chen *et al.* [1994] possesses the ability to *forget* old or invalid data. The methodology is simple, the old *unwanted* data is removed from the aggregated relational matrix, $\tilde{R}_{k+1}$, by *minimizing* over the complement of the current estimate, $\overline{R'_k}$. At each sampling instant the algorithm learns the relational matrix is as follows:

$$\tilde{R}_{k+1} = \{\tilde{R}_k \wedge \overline{R'_k}\} \vee R''_k \tag{7.77}$$

where
$$R'_k = u_k \wedge x_k \wedge y'_{k+1} \tag{7.78}$$

is an estimate based on the prediction y'_{k+1},

$$y'_{k+1} = u_k \circ x_k \circ \tilde{R}_k \tag{7.79}$$

is the predicted value based on the last estimate, $\tilde{R}_k$,

$$\overline{R'_k} = 1 - R'_k \tag{7.80}$$

is the complement,

and
$$R''_k = u_k \wedge x_k \wedge y_{k+1} \tag{7.81}$$

is the current estimate based on the actual k-th data triple.

The technique, as presented, does eliminate old data entries. The *forgetting* does not effect the whole matrix since only specific entries are discarded. However, the same technique used to *remove* old and invalid data may also result in the loss of valid data.

The Chen *et al.* [1994] algorithm produced no improvement over the simpler methodology by Pedrycz [1984(a)]. Almost all indices were higher for the algorithm proposed by Chen *et al.* [1994], as shown in Table 7.2. This algorithm was also considered for a batch identification. Again, there was virtually no improvement as compared to the original Pedrycz [1984(a)] algorithm. These batch results are also given in Table 7.2.

This technique of deleting old data was also incorporated into the DiNola *et al.* [1984] algorithm, again with no improvement in results. As well, a modification of this algorithm was made so that it only deleted data that would result in a 0 (zero) entry any position of the y_k fuzzy vector of the DiNola *et al.* [1984] algorithm, again with no improvement in results.

In conclusion, the idea to *unlearn or forget* unwanted results is good, however, the technique that these authors present to accomplish this goal does not, in general, appear to be effective.

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
Pedrycz [1984(a)] On-Line	*Prod*	*Max-prod*	0.6184	0.6141	0.5696	0.1689
		Max-min	0.5545	0.5647	0.6340	0.1548
	Min	*Max-prod*	0.6320	0.6426	0.7373	0.2081
		Max-min	0.6070	0.6636	1.013	0.3465
Chen *et al.* [1994] On-Line	*Prod*	*Max-prod*	0.6416	0.6662	0.6613	0.2327
		Max-min	0.5730	0.5814	0.6735	0.1702
	Min	*Max-prod*	0.6447	0.6661	0.8233	0.2685
		Max-min	0.6093	0.6668	1.0120	0.3363
Pedrycz [1984(a)] Batch	*Prod*	*Max-prod*	0.5956	0.5450	0.5733	0.1558
		Max-min	0.5342	0.4993	0.6675	0.1603
	Min	*Max-prod*	0.6293	0.5947	0.7742	0.2135
		Max-min	0.5899	0.6378	1.1060	0.3978
Chen *et al.* [1994] Batch	*Prod*	*Max-prod*	0.6065	0.5658	0.6818	0.2236
		Max-min	0.5422	0.5089	0.7991	0.1705
	Min	*Max-prod*	0.6522	0.6298	0.8873	0.2834
		Max-min	0.6030	0.6635	1.1150	0.3886

Table 7.2: Comparison of Identification Results by Pedrycz [1984(a)] and Chen *et al.* [1994]

7.4.5 The Identification Algorithm by Xu *et al.* [1987]

Xu *et al.* [1987] started with the same batch identification algorithm presented by Pedrycz [1984(a)] and tested it *on-line* for both a *max-min* and a *max-product* composition with the Box-Jenkins data [1970]. It is of interest to note that differences between the values of J_2 quoted in the paper by Pedrycz's [1984(a)] (J_2 = 0.776 for *max-product*) and the paper by Xu *et al.* [1987] (J_2 = 0.8501 for *max-product*) is a result of the different shape of the referential fuzzy sets used, since both papers had the same number of referential fuzzy sets and defuzzified using the center of gravity method.

By extending the batch algorithm of Pedrycz [1984(a)] to a self-learning formulation these authors attempt to overcome the main problem inherent in the identification algorithms proposed by DiNola *et al.* [1984] (i.e. the *minimization* of the R_k's to determine the overall process R) and Pedrycz [1984(a)], (i.e. the *maximization* of the R_k's). This gross *minimization* or *maximization* can leave the overall relational matrix unrecoverable if the process drifts, or if there are large inconsistencies in the data which in effect *minimizes* or *maximizes* the R such that the matrix can not recover. Since fuzzy models do not normally contain a disturbance term, the disturbance must be modeled in the R matrix along with the process information. Discarding all inconsistent data would result in discarding part of the process information. Therefore, an effective identification algorithm must have the ability to recover from process drift and large data inconsistency which could render the model ineffective.

The largest obstacle in the analysis of this paper by Xu *et al.* [1987] is the unconventional notation used to develop and express the algorithm. Chen *et al.* [1994] referenced this work by Xu *et al.* [1987] but did not compare their algorithm against it for the stated reason that:

> '*the formulas are not completely expressed and calculated in fuzzy sets, and this makes it difficult to systematically analyze and develop the fuzzy model in fuzzy set theory*'.

This same difficulty was also encountered during the preparation of this work. A paper by Ridley *et al.* [1988] reviewed the algorithm by Xu *et al.* [1987], however, the same unconventional notation was used as in former paper. An attempt has been made in this thesis to interpret the work by Xu *et al.* [1987] for comparative purposes, using a more conventional notation. Its is this interpretation that is presented below.

Xu *et al.* [1987] adapted the batch ID procedure to the on-line format by introducing a tuning parameter, a_l, which adjusts the speed of learning so that the relational matrix is calculated by the formula:

$$\tilde{R}_k(u,x,y_l) = a_l R''_k(u,x,y_l) + (1-a_l)\tilde{R}_{k-1}(u,x,y_l) \tag{7.82}$$

for $l = (1, 2, \ldots, n)$

where

$$R''_k = u^{q_u}_{k-\tau-1} \wedge x^{q_x}_{k-1} \wedge y_k \tag{7.83}$$

is the current estimate of the actual k-th data triple

and

$$u^{q_u} = \{u_i \mid u_i > q_u \quad , i = 1,\ldots ,m\}$$

$$x^{q_x} = \{x_j \mid x_j > q_x \quad , j = 1,\ldots ,n\}$$

are the α-cuts equivalents, or in this case the q-cuts of the contributing fuzzy input and state, respectively. The values of the q_*'s may or may not be unique.

When $a_l = 0$, there is no modification, when $a_l = 1$, the relational matrix is completely replaced. So selecting a value for a involves a tradeoff between the rate of learning and the sensitivity to noise. A large value of a_l will result in a faster learning rate, however, the model will be more easily affected by noise and disturbances. A smaller a_l will reduce the effect of noise on the model, but it will also result in a slower learning rate.

The autho.s [Xu *et al.*, 1987] determined a_l based on two factors:

(*i*) the amplitude of the non-fuzzy error:

$$|\mathbf{e}_k| = |\mathbf{y}_k - \tilde{\mathbf{y}}_k| \tag{7.84}$$

where $\tilde{\mathbf{y}}_k$ is the defuzzification of: $\tilde{\mathbf{y}}_k = \boldsymbol{u}^{q_u}_{k-\tau-1} \circ \boldsymbol{x}^{q_x}_{k-1} \circ \tilde{\boldsymbol{R}}_{k\text{-}1}$

Thus a large error will contribute to a large a_l, and $a_l = 0$ if $|\mathbf{e}_k| = 0$.

(*ii*) the relative contribution, b_l, of each of the n rules (i.e. y_l for $l = 1, \ldots, n$) of $\tilde{\boldsymbol{R}}_{k\text{-}1}$ to $\tilde{\boldsymbol{y}}_k$.

where $\tilde{\boldsymbol{y}}_k = \boldsymbol{u}^{q_u}_{k-\tau-1} \circ \boldsymbol{x}^{q_x}_{k-1} \circ \tilde{\boldsymbol{R}}_{k\text{-}1}$

So that if l-th slice of the matrix $\tilde{\boldsymbol{R}}_{k\text{-}1}$ (i.e. $\tilde{\boldsymbol{R}}_{k-1}(\boldsymbol{u},\boldsymbol{x},y_l)$)
contributes more to $\tilde{\boldsymbol{y}}_k$ it will undergo more of a modification due to a larger a_l.

So a_l is defined as:

$$a_l = h \cdot b_l \cdot |\mathbf{e}_k| \tag{7.85}$$

b_l is calculated as:

$$b_l = [\ \tilde{y}_l\ ]^2 \tag{7.86}$$

and h is a constant used to control the range of a_l and all are calculated for $l = 1, , \ldots, n$

The entire algorithm can be summarized as:

if $|\mathbf{e}_k| < \varepsilon$

then $\tilde{\boldsymbol{R}}_k = \tilde{\boldsymbol{R}}_{k-1}$

else $\tilde{\boldsymbol{R}}_k(\boldsymbol{u},\boldsymbol{x},y_l) = a_l \boldsymbol{R}''_k(\boldsymbol{u},\boldsymbol{x},y_l) + (1-a_l)\tilde{\boldsymbol{R}}_{k-1}(\boldsymbol{u},\boldsymbol{x},y_l)$

for $l = (1, 2, \ldots, n)$

As will be discussed in Chapter 8, many of the minimization criteria used in the optimization of fuzzy systems are based on the following fuzzy difference, [Pedrycz 1983; Pedrycz 1991(b); Ikoma *et al.* 1993; Valente de Oliveira 1993]:

$$\sum_{l=1}^{n} (\tilde{y}_l - y_l)^q \tag{7.87}$$

There are some problems with minimizations based on this fuzzy criterion, when the actual criterion for *good* identification is based on the discrete difference:

$$(\tilde{\mathbf{y}} - \mathbf{y})^q \tag{7.88}$$

The algorithm by Xu *et al.* [1987] is based on the discrete difference which is a practical contribution from a control perspective.

Xu *et al.* [1987] report a decrease in the value of J_2 from 0.4555, in a batch algorithm based on Pedrycz [1984(a)] to 0.328 for their on-line algorithm. However, to achieve this value of J_2 = 0.328, the same Box-Jenkins [1970] data was processed through the identification algorithm at least 3 times, once using the Pedrycz's [1984(a)] algorithm and twice using their algorithm in batch. Although, this reduction in J_2, illustrated in Figure 7.2 of their paper, demonstrates the ability of the algorithm to improve a relational matrix by learning, the opportunity to learn a set of data several times and then predict using the same data is seldom available in practical applications. Even the practice of predicting on the same data that was used during identification, and used here with these comparisons, is currently finding less favor in the realms of fuzzy identification [Shaw *et al.*, 1992], as well as in the realms of discrete identification.

The results from the application of the interpreted Xu *et al.* [1987] algorithm to the Box-Jenkins data [1970] both for batch and on-line approaches are listed in Table 7.3. Clearly, the interpretation of the work by Xu *et al.* [1987], which produced a J_2 = 0.6308, did not produce the expected results expected since any improvement in J_2 can be attributed entirely to the *q*-cuts of the original batch learning. Thus it appears that the interpretation of this work is in error. Consequently the dissemination of information anticipated by the publication of the paper by Xu *et al.* [1987] has not been achieved since the algorithm could not be reproduced from the information provided.

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
Xu *et al.* [1987]	*Prod* q=0.225	*Max-prod*	0.6247	0.6301	0.5438	0.1786
	h=0.05; ε=1.0	*Max-min*	0.5189	0.4696	0.5112	0.1305
On-Line	*Min* q=0.3	*Max-prod*	0.6240	0.5641	0.7401	0.2242
	h=0.05; ε=1.0	*Max-min*	0.6249	0.6021	0.9766	0.3746
Xu *et al.* [1987]	*Prod*	*Max-prod*	0.5337	0.5076	0.5164	0.1503
	q=0.225	*Max-min*	0.5023	0.4717	0.5429	0.1396
Batch	*Min*	*Max-prod*	0.6345	0.5749	0.7091	0.2065
	q=0.3	*Max-min*	0.6425	0.6308	0.9669	0.3737

Table 7.3: Results of Xu *et al.* [1987] Identification Algorithm

7.4.6 The Identification Algorithm by Shaw *et al.* [1992]

Shaw *et al.* [1992] propose a *weighted average fuzzy relational identification algorithm* for determining $\tilde{R}$. Their algorithm is based on a first order discrete-time model which utilizes a moving average window after the initial identification priming and is therefore considered self-learning. The moving average window introduces a bandpass characteristic into the identifier so that non-stationary effects present in the data can be rejected. Although their approach provides smooth averaging of the information, the idea of a windowed approach in a fuzzy system may be ill-advised due to the problem of model completeness. The procedure estimates the relational matrix, $\tilde{R}$, as follows:

$$\tilde{R}(u_{k-\tau-1}, x_{k-1}, y_k) = \frac{\sum_{k=\tau+2}^{N} \prod_{\substack{1\le i\le m \\ 1\le j\le n \\ 1\le l\le p}} (u_i^{(k-\tau-1)}, x_j^{(k-1)}, y_l^{(k)})}{\sum_{k=\tau+2}^{N} \prod_{\substack{1\le i\le m \\ 1\le j\le n}} (u_i^{(k-\tau-1)}, x_j^{(k-1)})} \tag{7.89}$$

The procedure in the paper by Shaw *et al.* [1992] is to combine the data with a *product* operation for the *max-min* composition. No values of J_2 for this algorithm are directly quoted in the paper by Shaw *et al.* [1992]. However, an earlier publication by Ridley *et al.* [1988], involving the same procedure as Shaw *et al.* [1992], quotes a value of J_2 = 0.2839 for the *max-min* composition.

Verification of this work for a batch *product* identification resulted in J_2 = .3923, for a *max-product* composition and J_2 = .3640, for the *max-min* composition. When the learned relational matrix, $\tilde{R}$, was normalized for this identification, the *max-product* composition results improved to $J_2 = .3545$, and the *max-min* results only deteriorated slightly to $J_2 = .3683$.

The batch computations were repeated using a *min* formulation, as shown in equation (7.90). The *max-product* prediction result improved to $J_2 = .3589$ and the *max-min* composition resulted in a poorer prediction with $J_2 = .4650$.

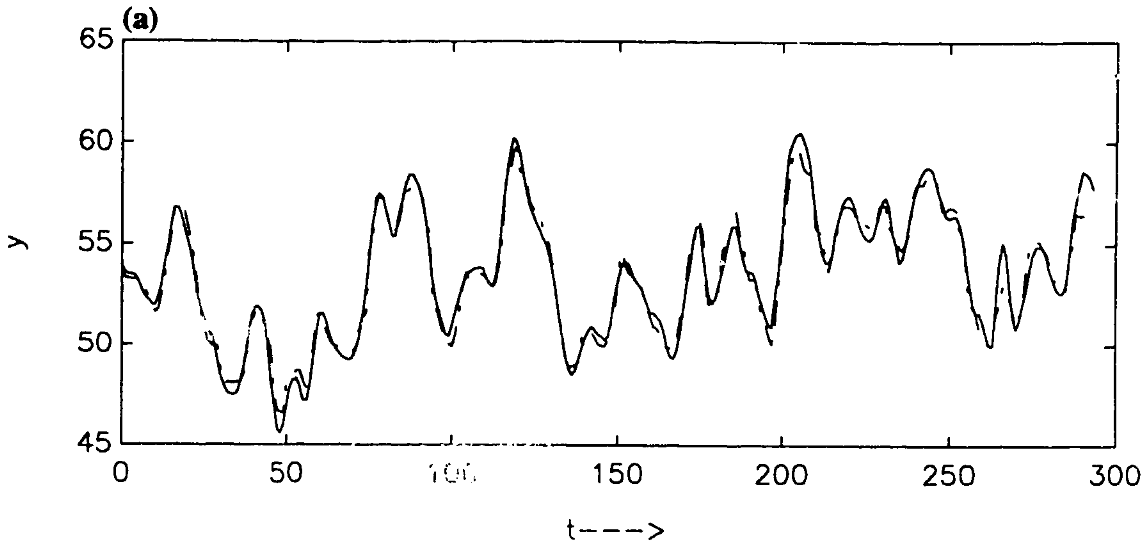
$$\tilde{R}(u_{k-\tau-1}, x_{k-1}, y_k) = \frac{\sum_{k=\tau+2}^{N} \prod_{1\le l\le n} \left(\bigwedge_{\substack{1\le i\le m \\ 1\le j\le n}} (u_i^{(k-\tau-1)}, x_j^{(k-1)}), y_l^{(k)} \right)}{\sum_{k=\tau+2}^{N} \bigwedge_{\substack{1\le i\le m \\ 1\le j\le n}} (u_i^{(k-\tau-1)}, x_j^{(k-1)})} \qquad (7.90)$$

Since the paper by Shaw *et al.* [1992] deals specifically in a *max-min* environment it is clear why the *product* operation was chosen for their identification algorithm since it produced the superior results. A summary of the batch results for the Shaw *et al.* [1992] identification algorithm are listed in Table 7.4.

The moving average window for on-line identification, presented by Shaw *et al.* [1992] is calculated as follows:

$$\tilde{R}(u_{k-\tau-1}, x_{k-1}, y_k) = \frac{\sum_{t=k-W}^{k} \prod_{\substack{1\le i\le m \\ 1\le j\le n \\ 1\le l\le n}} (u_i^{(t-\tau-1)}, x_j^{(t-1)}, y_l^{(t)})}{\sum_{t=k-W}^{k} \prod_{\substack{1\le i\le m \\ 1\le j\le n}} (u_i^{(t-\tau-1)}, x_j^{(t-1)})} \qquad (7.91)$$

where the width of the moving average window is W.



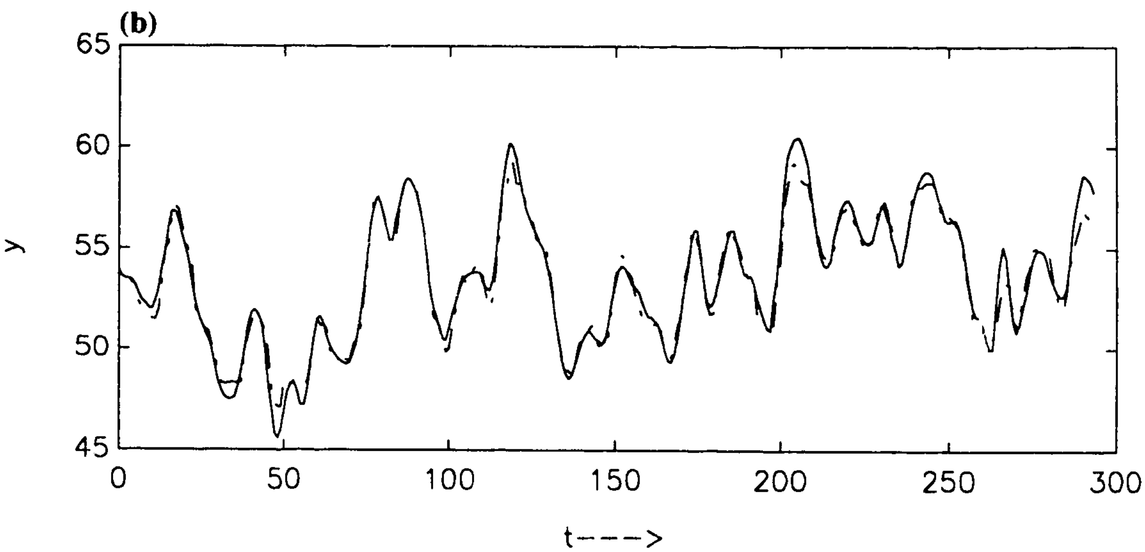


Figure 7.6: Batch ID for Shaw *et al.* [1992] Algorithm;
(a) Max-Product: *R*(prod; normal), $J_1 = 0.4473$; $J_2 = 0.3545$; $Q_1 = 0.4602$; $Q_2 = 0.1098$;
(b) Max-Min: *R*(prod; non-normal), $J_1 = 0.4390$; $J_2 = 0.3640$; $Q_1 = 0.4988$; $Q_2 = 0.1021$;
(——— Actual; - - - - Model)

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
Shaw *et al.* [1992]	*Prod*	*Max-prod*	0.4959	0.3923	0.5549	0.1780
		Max-min	0.4390	0.3640	0.4988	0.1021
***R*(non-normal)**	*Min*	*Max-prod*	0.4678	0.3589	0.5000	0.1189
		Max-min	0.4844	0.4650	0.6522	0.1641
Shaw *et al.* [1992]	*Prod*	*Max-prod*	0.4473	0.3545	0.4602	0.1098
		Max-min	0.4308	0.3683	0.5747	0.1388
***R*(normal)**	*Min*	*Max-prod*	0.4584	0.3547	0.5076	0.1202
		Max-min	0.4911	0.4768	0.6820	0.1797

Table 7.4: Batch Identification Results by Shaw *et al.* [1992] Algorithm

As mentioned previously, the moving average window approach is not well suited to relational matrix learning, since one can not guarantee that the relational matrix can be totally learned and remain in that state in the window specified. Windowing the learning, as outlined by Shaw *et al.* [1992], can render the relational matrix incomplete and present serious stability issues in a control application.

The continued summation of the input-output data required for this averaging procedure provides the algorithm with a robustness to disturbances, which on the other hand could render the algorithm slow to respond to process changes. A method to overcome the slow response inherent in this identification system will be presented in the next section.

7.5 Proposed Identification Algorithm

7.5.1 The Basic Algorithm

The development of this new on-line identification procedure begins with the inverse algorithm presented by DiNola *et al.* [1984]. As observed when comparing the results of the identification methods of DiNola *et al.* [1984] and Pedrycz [1984(a)], solving a system of fuzzy relation equations by aggregating the solutions of the individual equations (i.e. $\hat{\boldsymbol{R}} = \bigcap_{k \neq i}^{N} \hat{\boldsymbol{R}}_k$) may produce poor results if the relevant existence conditions are violated and ignored. Chen *et al.* [1994] attempted to improve the solution by unlearning or deleting old and presumably invalid results. However, as shown previously, the methodology presented did not improve the solution for the Box-Jenkins [1970] data tested. Xu *et al.* [1987] proposed a gradual learning with a forgetting factor, which appears to produce the desired results of a improved solution, however the results could not be duplicated. Shaw *et al.* [1992] improved the solution by summing the individual solutions and then determining a *weighted* average. This last algorithm has the distinct advantage of not requiring any tuning. A new identification algorithm is developed by combining the techniques of DiNola *et al.* [1984] and Shaw *et al.* [1992].

For this new identification algorithm the relational matrix, $\tilde{R}$, learns by making the following adaptations to the algorithm by Shaw *et al.* [1992], equation (7.89):

$$\tilde{R}(u_{k-\tau-1}, x_{k-1}, y_k) = \left. \sum_{k=\tau+2}^{N} \Big(\prod_{\substack{1\le i\le m \\ 1\le j\le n}} (u_i^{(k-\tau-1)}, x_j^{(k-1)}) \underset{1\le l\le n}{\theta} y_l^{(k)} \Big) \middle/ \sum_{k=\tau+2}^{N} \prod_{\substack{1\le i\le m \\ 1\le j\le n}} (u_i^{(k-\tau-1)}, x_j^{(k-1)}) \right. \tag{7.92}$$

where θ is the minimum inverse operation for the *max-product* composition defined as follows.

Definition 10: For a and $b \in [0,1]$, the θ-composition is defined as:

$$a\,\theta\,b = \begin{cases} 0 & \text{if} \quad a < b \quad \text{or} \quad a = b = 0 \\ b/a & \text{if} \quad a \ge b \end{cases} \tag{7.93}$$

This identification algorithm can also be applied to the *max-min* composition by making the appropriate modifications to equation (7.92) as shown (i.e. $\wedge$ for Π, and σ for θ) so that:

$$\tilde{R}(u_{k-\tau-1}, x_{k-1}, y_k) = \left. \sum_{k=\tau+2}^{N} \Big(\bigwedge_{\substack{1\le i\le m \\ 1\le j\le n}} (u_i^{(k-\tau-1)}, x_j^{(k-1)}) \underset{1\le l\le n}{\sigma} y_l^{(k)} \Big) \middle/ \sum_{k=\tau+2}^{N} \bigwedge_{\substack{1\le i\le m \\ 1\le j\le n}} (u_i^{(k-\tau-1)}, x_j^{(k-1)}) \right. \tag{7.94}$$

where σ is the inverse operation for the *max-min* composition with the following definition.

Definition 11: For a and $b \in [0,1]$, the σ-composition is defined as:

$$a\,\sigma\,b = \begin{cases} 0 & \text{if} \quad a < b \\ b & \text{if} \quad a \ge b \end{cases} \tag{7.95}$$

Cross identification for the algorithm consists of exchanging the *product* and *min* operations between the $\boldsymbol{u}$ and $\boldsymbol{x}$ fuzzy vectors only, not the inverse operations.

The results of a batch identification, provided in Table 7.5, applying this algorithm to the Box-Jenkins data [1970] shows that the J_2 values for all batch identifications decreased, for the unadjusted $\boldsymbol{R}$, versus the Shaw *et al.* [1992] algorithm. Graphical results are shown in Figure 7.7. However, *normalization* did not result in the same solution improvement as with the previous algorithm. This is due to the matrix check which this algorithm performs to ensure that the values of the estimated relational matrix remain in the unit interval. As a result the relational matrix remains close to *normal* at all times. So even though *normalization* of the relational matrix does not result in an improved solution as noted with the other algorithms, it does not result in a substantial degradation of the solution either.

The proposed fuzzy relational matrix identification algorithm was also tested *on-line*. The results of the *on-line learning* for this new algorithm, provided in Table 7.6, when compared against the algorithm by Shaw *et al.* [1992] show that the proposed algorithm is consistently better for the *max-product* composition, under either identification. Results for *max-min* composition are comparable, with the values of J_2 slightly higher for the new proposal.

The problem with the data averaging that takes place under the proposed identification system and similarly the algorithm by Shaw *et al.* [1992], is that if the system is changing quickly, the averaging operation makes the identification algorithm slow to react. Restarting the identification from *scratch* is not a viable alternative as valid information may be lost. A compromise between these two extremes is to *reset the matrix*. This on-line adaptive approach is presented in the next section.

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
Shaw *et al.* [1992]	*Prod*	*Max-prod*	0.4959	0.3923	0.5549	0.1780
		Max-min	0.4390	0.3640	0.4988	0.1021
Batch	*Min*	*Max-prod*	0.4678	0.3589	0.5000	0.1189
		Max-min	0.4844	0.4650	0.6522	0.1641
New Proposal	*Prod*	*Max-prod*	0.4732	0.3826	0.5314	0.1243
		Max-min	0.4433	0.3640	0.4878	0.1005
Batch	*Min*	*Max-prod*	0.4301	0.3359	0.4557	0.0993
		Max-min	0.4535	0.4127	0.4911	0.1017

Table 7.5: Comparison of Shaw *et al.* [1992] and New Proposal *Batch* Identification Techniques

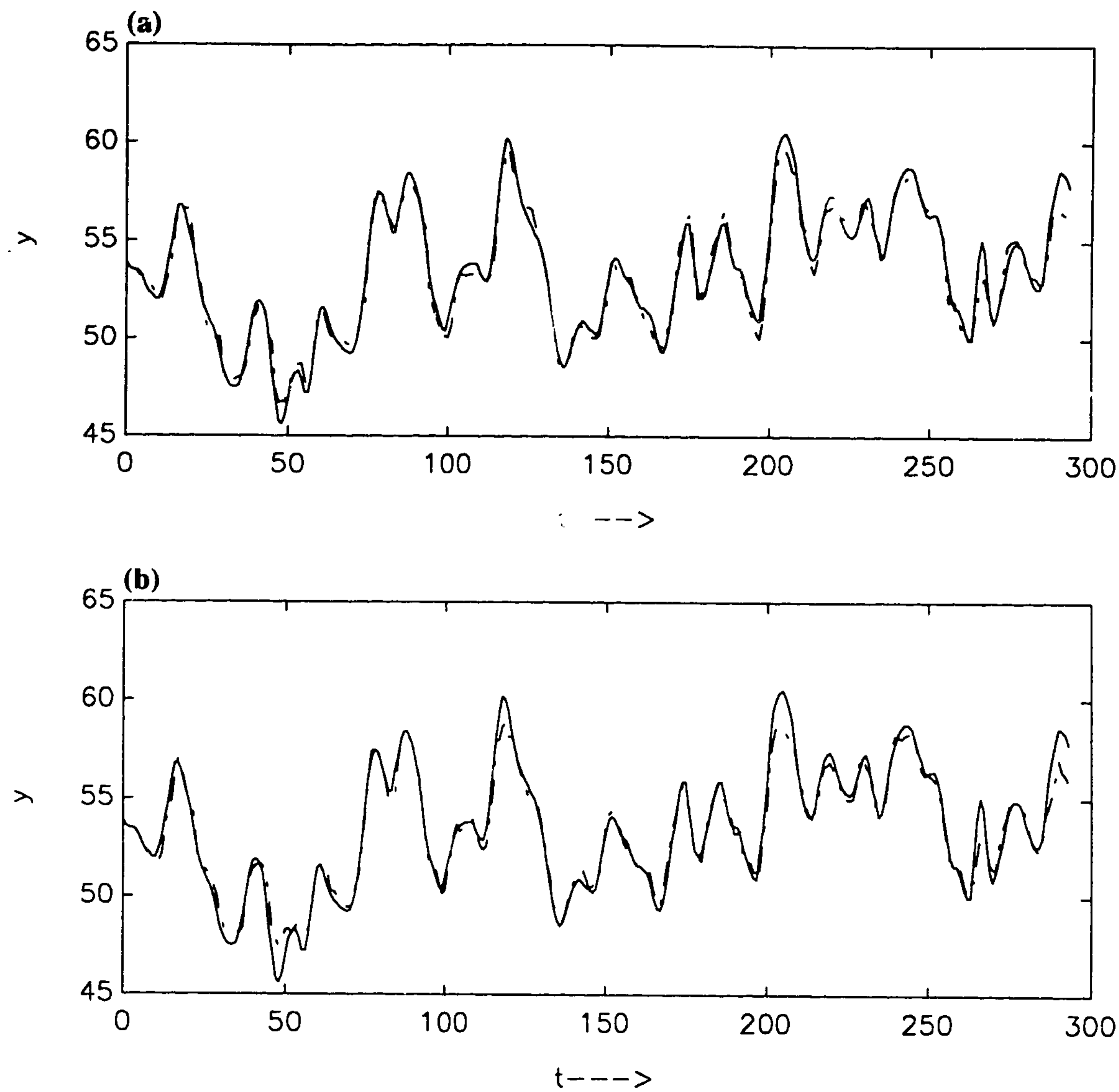


Figure 7.7: Batch ID for New Proposal Identification Algorithm;
(a) Max-Product: R(min; non-normal), $J_1 = 0.4301$; $J_2 = 0.3359$; $Q_1 = 0.4557$; $Q_2 = 0.0993$;
(b) Max-Min: R(prod; non-normal), $J_1 = 0.4433$; $J_2 = 0.3640$; $Q_1 = 0.4878$; $Q_2 = 0.1005$;
(——— Actual; - - - - Model)

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
Shaw *et al.* [1992]	*Prod*	*Max-prod*	0.5784	0.5495	0.5663	0.1871
		Max-min	0.5067	0.4748	0.5300	0.1193
On-Line	*Min*	*Max-prod*	0.5596	0.4970	0.5205	0.1309
		Max-min	0.5392	0.5246	0.6741	0.1803
New Proposal	*Prod*	*Max-prod*	0.5373	0.4674	0.5518	0.1370
		Max-min	0.4649	0.4114	0.5132	0.1160
On-Line	*Min*	*Max-prod*	0.5493	0.5142	0.5173	0.1263
		Max-min	0.5147	0.5105	0.5233	0.1189

Table 7.6: Comparison of Shaw *et al.* [1992] and New Proposal *On-Line* Identification Algorithms

7.5.2 Matrix Completeness and Matrix Reset

The basic on-line learning algorithm used in both the proposed algorithm and the algorithm by Shaw *et al.* [1992] is outlined as follows:

Using the proposed new *max-product* identification formulation, at any instant t, the relational matrix, $\tilde{\boldsymbol{R}}_t$, is calculated:

$$\tilde{\boldsymbol{R}}\boldsymbol{den}_t = \tilde{\boldsymbol{R}}\boldsymbol{den}_{t-1} + \prod_{\substack{1\le i\le m \\ 1\le j\le n}} (u_i^{(t-\tau-1)}, x_j^{(t-1)}) \tag{7.96}$$

$$\tilde{\boldsymbol{R}}\boldsymbol{num}_t = \tilde{\boldsymbol{R}}\boldsymbol{num}_{t-1} + \tilde{\boldsymbol{R}}\boldsymbol{den}_t \underset{1\le l\le n}{\psi} y_l^{(t)} \tag{7.97}$$

$$\tilde{\boldsymbol{R}}_t = \tilde{\boldsymbol{R}}\boldsymbol{num}_t \Big/ \tilde{\boldsymbol{R}}\boldsymbol{den}_t \tag{7.98}$$

for $t = \tau+2, \ldots, N$. As N increases the values of $\tilde{\boldsymbol{R}}\boldsymbol{den}_t$ and $\tilde{\boldsymbol{R}}\boldsymbol{num}_t$ increase such that new input-output data containing information of a recent process change would have little effect on the calculated value of $\tilde{\boldsymbol{R}}_t$. Thus a method to *reset* the matrix is required such that the information contained in the overall relational matrix, $\tilde{\boldsymbol{R}}_t$, is preserved, yet the cumulative values of $\tilde{\boldsymbol{R}}\boldsymbol{den}_t$ and $\tilde{\boldsymbol{R}}\boldsymbol{num}_t$ are reduced so that the model can respond more quickly to process changes. Such *forgetting* of old data is common in discrete identification.

Analyzing the affect of *normalization* for the various identification algorithms, summarized in Table 7.7, *normalization* of the relational matrix generally improves the solution, with only a small degradation for those solutions without improvement. Another benefit of *normalization* is that if there has been sufficient learning of the relational matrix such that:

$$\sum_{l} \tilde{R}(u_i, x_j, y_l) \geq \varepsilon \qquad \text{for all } i \text{ and } j \tag{7.99}$$

for an arbitrary $0 < \varepsilon < 1$, then *normalization* of the matrix ensures a *complete* matrix, since each u_i and x_j can determine a y_l.

The methodology for *resetting* the identification matrix consists of *normalizing* the current $\tilde{R}_t$ and then reassigning the values of $\tilde{R}\,den_t$ and $\tilde{R}\,num_t$ such that they are consistent with the *normalized* $\tilde{R}_t$. The *reset* algorithm is as follows.

Let $\tilde{R}_t^n(u_i, x_j, y) = \tilde{R}_t(u_i, x_j, y) \big/ \max(\tilde{R}_t(u_i, x_j, y))$ for each $i = \{1, 2, \ldots, m\}$ and $j = \{1, 2, \ldots, n\}$ be the *normalized* relational matrix. Then:

(1) $$\tilde{R}_t(u, x, y) = \tilde{R}_t^n(u, x, y) \tag{7.100}$$

(2) $$\tilde{R}\,num_t(u, x, y) = \tilde{R}_t^n(u, x, y) \tag{7.101}$$

(3) $$\tilde{R}\,den_t(u, x, y) = [1]_{m \times n} \tag{7.102}$$

The values of $\tilde{R}\,den_t$ and $\tilde{R}\,num_t$ have been *reset* to values in the unit interval. Any new input-output data added to these matrices will now have a larger impact of the calculation of $\tilde{R}_t$. The following example illustrates this feature.

Example 1:

A relational matrix, R_1, was identified in batch using the averaging algorithm of equation (7.89) and 12,000 points of data generated from an arbitrary process, P_1. The process was changed to P_2, and identification continued on-line. Relational matrix R_{21} continued learning with the information from R_1. Relation matrix R_{22} was *reset* from the information contained in R_1, so that learning was continued with the *reset matrix*. As shown in Figure 7.8(a), predictions from the *reset matrix* were better and responded more quickly than for the *unreset matrix*. Figure 7.8(b) graphs the J_2 values for the predictions from each of the reset and unreset matrices.

Since predictions from *normalized* matrices for the averaging algorithm do not deteriorate from those of the unadjusted matrices, *resetting* or *normalizing* a matrix to reduce the quantity of matrix is complete prior to *matrix resetting*, resetting the matrix does not change that status, an important consideration for fuzzy control.

			Batch		On-Line	
			R(un-adjusted)	R (normal)	R(un-adjusted)	R (normal)
Author	**ID**	**OSA**	J_2	J_2	J_2	J_2
Pedrycz [1984(a)]	*Prod*	*Max-prod*	0.5450	0.3787	0.6141	0.4590
		Max-min	0.4993	0.4484	0.5647	0.4499
	Min	*Max-prod*	0.5947	0.4796	0.6426	0.5602
		Max-min	0.6378	0.6837	0.6636	0.6704
Chen *et al.* [1994]	*Prod*	*Max-prod*	0.5658	0.4190	0.6662	0.5409
		Max-min	0.5089	0.4896	0.5814	0.5455
	Min	*Max-prod*	0.6298	0.5937	0.6661	0.6318
		Max-min	0.6635	0.7882	0.6668	0.7433
Xu *et al.* [1987]	*Prod*	*Max-prod*	0.5076	0.4339	0.6301	0.5014
		Max-min	0.4717	0.4957	0.4696	0.6174
	Min	*Max-prod*	0.5749	0.4629	0.5641	0.4957
		Max-min	0.6308	0.6745	0.6021	0.6532
Shaw *et al.* [1992]	*Prod*	*Max-prod*	0.3923	0.3545	0.5495	0.5126
		Max-min	0.3640	0.3683	0.4748	0.4827
	Min	*Max-prod*	0.3589	0.3547	0.4970	0.4942
		Max-min	0.4650	0.4768	0.5246	0.5361
New Proposal	*Prod*	*Max-prod*	0.3826	0.3835	0.4674	0.4684
		Max-min	0.3640	0.4128	0.4114	0.4528
	Min	*Max-prod*	0.3359	0.3681	0.5142	0.5219
		Max-min	0.4127	0.4972	0.5105	0.5888

Table 7.7: Comparison of J_2 for *R*(unadjusted) and *R*(normalized)

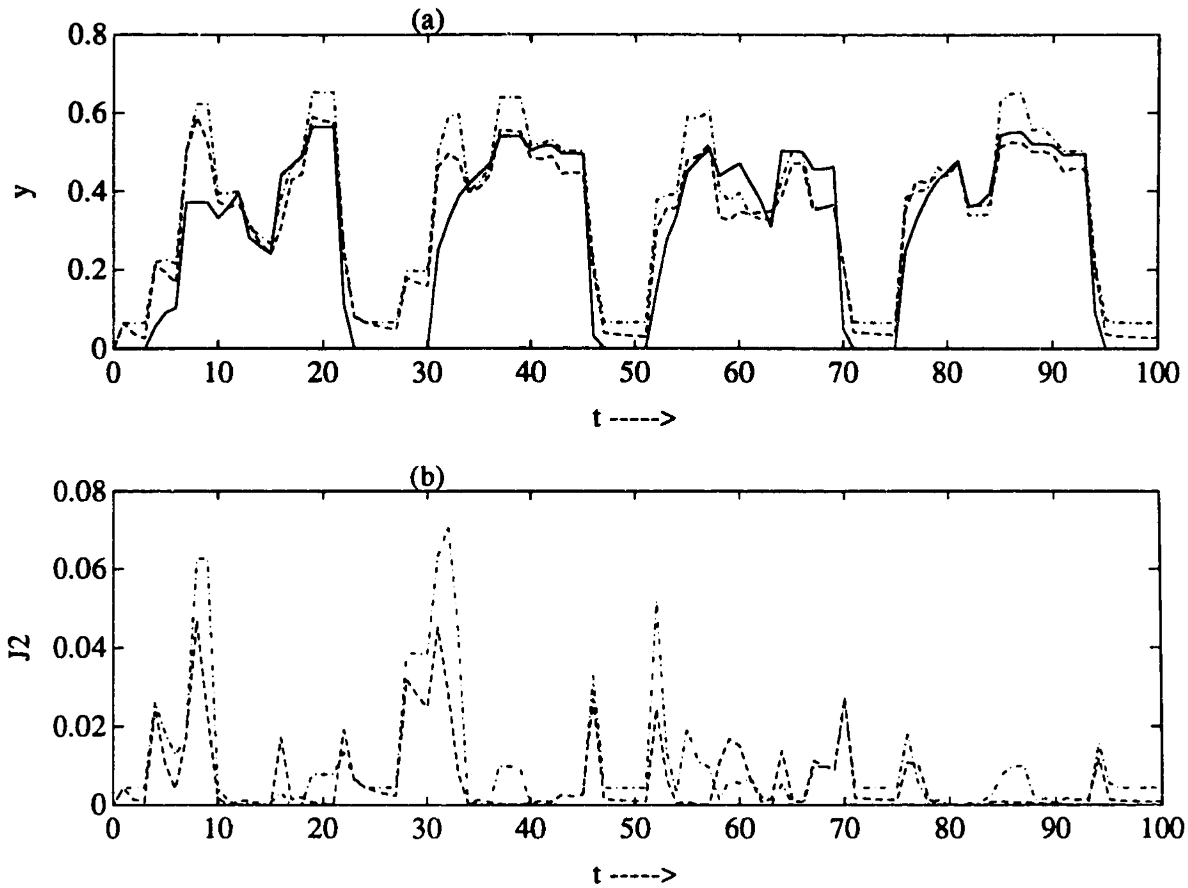


Figure 7.8: Comparison of Predictions with and without *Matrix Reset*

(a) Discrete output, y; y(actual) —— ; y(with reset) – – – –; y(without reset) – · – · –

(b) Performance Index, J_2; J_2(with reset) = 0.0062 – – – –; J_2(without reset) = 0.0097 – · – · –

7.6 Summary of Identification Algorithms Presented

The identification algorithms reviewed in this section provide a wide range of identification possibilities. Some conclusions regarding the identification algorithms tested are now provided. It should be noted that these conclusions do not include the results from the Xu *et al.* [1987] algorithm since it did not produce the expected results.

Identification Strategies

(1) The *max-min* composition predicts *better* when identified with a *product* operator.

(2) The *max-product* composition predicts *better* with a *product* operator for Pedrycz [1984(a)] based identifications and *better* with a *min* operator for the averaging procedures such as Shaw *et al.* [1992] and the New Proposed Algorithm.

Normalization of the Relational Matrix

Normalization of the relational matrix improved the predictive results except in the following cases:

(1) *Normalizing* relational matrices, identified with an inverse operation, does not improve the predictive solution (i.e. the New Proposal).

(2) *Normalization* does not improve a *max-min* composition predictive solution when the relational matrix is identified with a *min* operator.

Best Predictive Results

(1) ***Batch Unadjusted***

- *Max-min* prediction with a *product* identification for Pedrycz [1984(a)] based algorithms
- *Max-prod* prediction with a *min* identification for averaging algorithms (i.e. Shaw *et al.* [1992] and the New Proposal)

(2) ***Batch Normalized***

- *Max-product* prediction with a *product* identification for Pedrycz [1984(a)] based algorithms
- *Max-product* prediction with a *min* identification for averaging algorithms (i.e. Shaw *et al.* [1992] and New Proposal)

(3) ***On-Line Unadjusted***

- *Max-min* prediction with a *product* identification.

(4) ***On-Line Normalized***

- *Max-min* prediction with a *product* identification.

Of these results the most significant are the improved prediction results from cross identification, and, the improved results from matrix *normalization*. Cross identification permits a wider range of learning capability for relational matrices. Matrix *normalization* can be used with batch or on-line identifications as a method of sensitizing the relational matrix to process changes.

The strengths and weakness of the various identification algorithms have been discussed in the preceding sections, and will not be reiterated here. However, the algorithms can be ranked, based on the J_2 criterion. The algorithms are ranked by averaging the J_2 values for each author, after discarding the highest J_2 value. Table 7.8 shows the new proposed algorithm is ranked the *best* of the algorithms tested, with the Shaw *et al.* [1992] algorithm a very close second.

	Batch		On-Line		
Author	***R*(un-adjusted)**	***R* (normal)**	***R*(un-adjusted)**	***R* (normal)**	**Overall Rank**
New Proposal	1 (0.3608)	2 (0.3881)	1 (0.4631)	1 (0.4477)	1
Shaw *et al.* [1992]	2 (0.3717)	1 (0.3592)	2 (0.4988)	3 (0.4965)	2
Pedrycz [1984(a)]	4 (0.5463)	3 (0.4356)	4 (0.6071)	2 (0.4897)	3
Xu *et al.* [1987]	3 (0.5181)	4 (0.4642)	3 (0.5453)	4 (0.5382)	4
Chen *et al.* [1994]	5 (0.5682)	5 (0.5008)	5 (0.6379)	5 (0.5727)	5

Table 7.8: Identification Algorithm Ranking Based on J_2Criterion

It should be pointed out that the results from this work are dependent upon the Box-Jenkins [1970] data used in the evaluations. Being able to rank these algorithm using the same data, fuzzification, defuzzification and referential fuzzy sets provides a basis for judging their relative capability. However, the ranking of these identification algorithms is not absolute, and only reflects their relative capability for the data tested. Another set of data would, in all likelihood, produce a different ranking. But then again, an algorithm that produced *poor* predictive results with this data would most likely produce *poor* results with other data, and vice versa. Tables A3.1 to A3.4 of Appendix 3 summarize all the results from the test cases presented in this Chapter.

7.7 Summary

The literature review of Chapter 7 involves the evaluation, implementation and comparison of several identification algorithms [Pedrycz, 1984; DiNola *et al.*, 1984; Xu *et al.*, 1987; Shaw *et al.*, 1992; Chen *et al.*, 1994] using the same fuzzification and defuzzification methods, the same reference fuzzy sets and the Box-Jenkins [1970] gas furnace data [Bourke *et al.* 1995(b)]. This review is important in that it confirms the validity and ranks the relative ability of each of the algorithms tested.

Based on the identification techniques described by DiNola *et al.* [1984] and Shaw *et al.* [1992] a new identification algorithm is developed. The new algorithm uses the averaging technique, as described by Shaw *et al.* [1992], to determine the overall relation matrix $\boldsymbol{R}$ from the individual inverses calculated from a series of input-output data. The predictive results from this new algorithm are *better* than the other identification algorithms tested in this Chapter.

The new identification algorithm developed in this thesis is based on a batch learning technique. However, this new technique can be applied on-line, as well as the algorithm by Shaw *et al.* [1992], by using a matrix resetting mechanism developed in this thesis. The resetting technique maintains model completeness while increasing the speed at which learning can be performed, both important features for fuzzy identification systems.

The research into the development of the new identification algorithm, presented in Chapter 7, resulted in several other notable contributions.

(1) The necessary and sufficient condition for the existence of a solution to the first-order state space representation (i.e. $p = 1$) for input-output data triples for both *max-min* and *max-product* composition

This condition is key to the efficient on-line determination for the existence an exact solution in a control setting. The alternative is to determine the actual inverse and then apply the inverse result to the original equation to determine if the equation is satisfied or not. If the original equation is not satisfied, then an exact solution does not exist.

(2) Extension of *estimate of the minimum* relational matrix to include:

- *non-normal* input data [Bourke *et al.*, 1995(a)]
- the first process model representation
- the *max-product* composition

(3) Ordering of the relational matrix identification for the *max-min* composition [Bourke *et al.*, 1995(a)]:

$$\boldsymbol{R} \subseteq \bigcup_{k=1}^{N} \overline{\boldsymbol{R}}_k \subseteq \bigcap_{k=1}^{N} \hat{\boldsymbol{R}}_k \subseteq \bigcup_{k=1}^{N} \tilde{\boldsymbol{R}}_k \subseteq \bigcup_{k=1}^{N} \boldsymbol{R}_k^M$$

so that $y \leq \overline{y} \leq \hat{y} \leq \tilde{y} \leq y^M$

and for the *max-product* composition:

$$\boldsymbol{R} \subseteq \bigcup_{k=1}^{N} \overline{\boldsymbol{R}}_k \subseteq \bigcap_{k=1}^{N} \hat{\boldsymbol{R}}_k \subseteq \bigcup_{k=1}^{N} \tilde{\boldsymbol{R}}_k$$

so that $y \leq \overline{y} \leq \hat{y} \leq \tilde{y}$

On-line identification requires an algorithm which can be completed within the sampling interval of the given process. Identification of relational matrices through a procedure that estimates, assumes that the results are comparable to an exact procedure. The estimation theory presented by Baboshin *et al.* [1990] is analyzed in detail with the theoretical extensions as outline in point (2), above. The analysis and ordering of the variously identified relational matrices, as shown in point (3) above, are independent of the data used and are therefore representative of the capability of the estimation algorithms used.

REFERENCES

[1] Baboshin, N.P., Naryshkin, D.G., "On Identification of Multidimensional Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 35, pp. 325-331, 1990.

[2] Bourke, M.M., Fisher, D.G., "Calculation and Application of a Minimum Fuzzy Relational Matrix", *Fuzzy Sets and Systems*, 74, pp. 225-236, 1995(a).

[3] Bourke, M.M., Fisher, D.G., "A Comparative Analysis of Identification Algorithms for Fuzzy Relational Matrices", *Fuzzy Sets and Systems*, submitted 1995(b).

[4] Box, G.E.P., Jenkins, G.M., *Time Series Analysis, Forecasting and Control*, Holden Day, San Francisco, 1970.

[5] Chen, J.Q., Lu, J.H., Chen, L.J., "An On-Line Identification Algorithm for Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 64, pp. 63-72, 1994.

[6] Czogala, E., Pedrycz, W., "On Identification in Fuzzy Systems and its Applications in Control Problems", *Fuzzy Sets and Systems*, Vol. 6, pp. 73-83, 1981.

[7] Di Nola, A., Pedrycz, W., Sessa, S., "Some Theoretical Aspects of Fuzzy Relation Equations Describing Fuzzy Systems", *Information Sciences*, Vol. 34, pp. 241-264, 1984.

[8] Dubois, D., Prade, H., *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, Toronto, 1980.

[9] Gottwald, S., "Generalized Solvability Criteria for Fuzzy Equations", *Fuzzy Sets and Systems*, Vol. 17, pp. 285-296, 1985.

[10] Gottwald, S., Pedrycz, W., "Solvability of Fuzzy Relational Equations and Manipulation of Fuzzy Data, *Fuzzy Sets and Systems*, Vol 18, pp. 45-64, 1986.

[11] Graham, B.P., Newell, R.B., "Fuzzy Identification and Control of a Liquid Level Rig", *Fuzzy Sets and Systems*, Vol. 26, pp. 255-273, 1988.

[12] Graham, B.P., Newell, R.B., "Fuzzy Adaptive Control of a First Order Process, *Fuzzy Sets and Systems*, Vol. 31, pp. 47-65, 1989.

[13] Ikoma, N., Pedrycz, W., Hirota, K., "Estimation of Fuzzy Relational Matrix by using Probabilistic Descent Method", *Fuzzy Sets and Systems*, Vol. 57, pp. 335-349, 1993.

[14] Kosko, B., *Neural Networks and Fuzzy Systems*, Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1992.

[15] Lee, C.C., "Fuzzy Logic in Control Systems: Fuzzy Logic Controller", *IEEE Transactions on Systems, Man and Cybernetics*, Vol 20, Part I: pp. 404-418, Part II: pp. 419-435, 1990.

[16] Ljung, L., *System Identification: Theory for the User*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1987.

[17] Mamdani, E.H., "Application of Fuzzy Algorithms for Control of Simple Dynamic Plant", *Proceedings of the IEE*, Vol. 121, pp. 1585-1588, 1974.

[18] Mamdani, E.H., "Applications of Fuzzy Set Theory to Control Systems: A Survey", In: *Fuzzy Automata and Decision Processes*, Gupta, M.M., Saridis, G.N., Gaines, B.R. (Editors), North-Holland, Amsterdam, pp. 77-88, 1977.

[19] Mizumoto, M., "Improvement Methods of Fuzzy Controls", *Proceedings of 3rd IFSA*, Seattle, WA, Aug 6-11, pp. 60-62, 1989.

[20] Pedrycz, W., Hirota, K., "Fuzzy-Probabilistic Algorithms in Identification of Fuzzy Systems", Prep. 8th Triennial World Congress IFAC, Kyoto, Vol. 5, pp. 52-56, 1981.

[21] Pedrycz, W., "Numerical and Applicational Aspects of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 11, pp. 1-18, 1983.

[22] Pedrycz, W., "An Identification Algorithm in Fuzzy Relational Systems", *Fuzzy Sets and Systems*, Vol. 13, pp. 153-167, 1984(a).

[23] Pedrycz, W., "Identification in Fuzzy Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 14, pp. 361-366, 1984(b).

[24] Pedrycz, W., Czogala, E., Hirota, K., "Some Remarks on the Identification Problem in Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 12, pp. 185-189, 1984.

[25] Pedrycz, W., "Structured Fuzzy Models", *Cybernetics and Systems*, Vol. 16, pp. 103-117, 1985.

[26] Pedrycz, W., "Fuzzy Models and Relational Equations", *Mathematical Modeling*, Vol. 9., pp. 427-434, 1987.

[27] Pedrycz, W., "Approximate Solutions of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 28, pp. 183-202, 1988.

[28] Pedrycz, W., *Fuzzy Control and Fuzzy Systems*, John Wiley & Sons, Inc., New York, N.Y., 1989.

[29] Pedrycz, W., "Algorithms for Solving Fuzzy Relational Equations in a Probabilistic Setting", *Fuzzy Sets and Systems*, Vol. 38, pp. 313-327, 1990.

[30] Pedrycz, W., "Processing in Relational Structures: Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 40, pp. 77-106, 1991(a).

[31] Pedrycz, W., "Neurocomputations in Relational Systems", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, pp. 289-297, 1991(b).

[32] Pedrycz, W., "Fuzzy Modeling: Fundamentals, Construction and Evaluation", *Fuzzy Sets and Systems*, Vol. 41, pp. 1-15, 1991(c).

[33] Pedrycz, W., *Fuzzy Control and Fuzzy Systems, Second Extended Edition*, John Wiley & Sons, Inc., New York, N.Y., 1993.

[34] Pedrycz, W., "Why Triangular Membership Functions?", *Fuzzy Sets and Systems*, Vol. 64, pp. 21-30, 1994.

[35] Ridley, J.N., Shaw, I.S., Krüger, J.J., "Probabilistic Fuzzy Model for Dynamic Systems", *Electronics Letters*, Vol., 24, pp. 90-92, 1988.

[36] Sessa, S., "Some Results in the Setting of Fuzzy Relation Equations Theory", *Fuzzy Sets and Systems*, Vol. 14, pp. 281-297, 1984.

[37] Shaw, I.S., Krüger, J.J., "New Fuzzy Learning Model with Recursive Estimation for Dynamic Systems", *Fuzzy Sets and Systems*, Vol. 48, pp. 217-229, 1992.

[38] Sugeno, M., Tanaka, K., "Successive Identification of a Fuzzy Model and its Applications to Prediction of a Complex System", *Fuzzy Sets and Systems*, Vol. 42, pp. 315-334, 1991.

[39] Tong, R.M., "Analysis of Fuzzy Control Algorithms using the Relation Matrix", *International Journal of Man-Machine Studies*, Vol. 8, pp. 679-686, 1976.

[40] Tong, R.M., "Synthesis of Fuzzy Models for Industrial Processes - Some Recent Results", *International Journal of General Systems*, Vol. 4, pp. 143-162, 1978(a).

[41] Tong, R.M., "Analysis and Control of Fuzzy Systems using Finite Discrete Relations", *International Journal of Control*, Vol. 27, pp. 431-440, 1978(b).

[42] Tong, R.M., "The Evaluation of Fuzzy Models Derived From Experimental Data", *Fuzzy Sets and Systems*, Vol. 4, pp. 1-12, 1980.

[43] Valente de Oliveira, J., "Neuron Inspired Learning Rules for Fuzzy Relational Structures", *Fuzzy Sets and Systems*, Vol. 57, pp. 41-53, 1993.

[44] Vrba, J, "Peak-Pattern Concept and Max-min Inverse Problem in Fuzzy Control Modeling", *Fuzzy Sets and Systems*, Vol. 47, pp. 1-11, 1992.

[45] Wang, H.-F., "Numerical Analysis on Fuzzy Relation Equations with Various Operators", *Fuzzy Sets and Systems*, Vol. 53, pp. 155-166, 1993.

[46] Wagenknecht, M., Hartmann, K., "Fuzzy Modeling with Tolerances", *Fuzzy Sets and Systems*, Vol. 20, pp. 325-332, 1986.

[47] Wagenknecht, M., Hartmann, K., "On Direct and Inverse Problems for Fuzzy Equations Systems with Tolerances", *Fuzzy Sets and Systems*, Vol. 24, pp. 93-102, 1987.

[48] Xu, C.-W., Lu, Y.-Z., "Fuzzy Model Identification and Self-Learning for Dynamic Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol., 17, pp. 683-689, 1987.

[49] Xu, C.-W., "Fuzzy Systems Identification", *IEE Proceeding*, Vol. D-136, pp. 146-150, 1989.

[50] Zadeh, L.A., "Fuzzy Sets", *Information and Control*, Vol. 8, pp. 338-353, 1965.

CHAPTER 8 MINIMIZATION ALGORITHMS FOR FUZZY PERFORMANCE OBJECTIVES [1]

Since fuzzy systems are by their nature imprecise, it is not unreasonable to consider approximate solutions for those times when an exact solution can not be determined.

[Tong, 1980]

8.1 Introduction

When the simple fuzzy control problem, given y and $\boldsymbol{R}$, find $\boldsymbol{u}$ such that

$$y = \boldsymbol{u} \circ \boldsymbol{R} \tag{8.1}$$

satisfies the conditions for the existence of a solution, the results of Sanchez [1976] can be used to solve the problem. However, if the conditions for existence of a solution fail, then some approximate solution must be sought.

This chapter deals with inexact solution algorithms and the idea of minimization of a performance objective for fuzzy systems. In classical control when one considers minimization of an objective function the solution generally involves a derivative calculation and in many cases an iterative solution. This situation is paralleled in fuzzy systems.

The theory of derivatives for fuzzy systems is reviewed and shown to be valid. Then this functionality is applied to various minimization criteria using the Box-Jenkins gas furnace data [1970], as outlined in Chapter 7. The problems and results of applying three numerical procedures that utilize *fuzzy* derivatives are then reviewed in this Chapter.

8.2 Fuzzy Differential

The existence of a differential for a *fuzzy relation*, $\boldsymbol{R}$, will be shown to exist through the existence of a differential for a *fuzzy mapping*, f. To achieve this it must first be shown that the properties of a *fuzzy relation* are equivalent to a *fuzzy mapping*.

8.2.1 Properties of Fuzzy Relations

The extension principle, introduced by Zadeh [1975] is basic to the idea of fuzzy set theory. It provides a method of extending non-fuzzy mathematical concepts, such as arithmetic operations and calculus, to the fuzzy domain. This definition, presented in Chapter 3, lays the foundation for the development of a fuzzy derivative and is repeated here for convenience.

---

[1] A version of this chapter has been submitted for publication. Bourke M.M., Fisher D.G., 1995, *(Fuzzy*

Let $\mathcal{U}$ be a Cartesian product of universes, $\mathcal{U} = \mathcal{U}_1 \times \mathcal{U}_2 \times \ldots \times \mathcal{U}_n$, and $A_1, A_2, \ldots A_n$ be n fuzzy sets in $\mathcal{U}_1, \mathcal{U}_2, \ldots \mathcal{U}_n$, respectively. Let f be a mapping from $\mathcal{U}_1 \times \mathcal{U}_2 \times \ldots \times \mathcal{U}_n$ to a universe $\mathcal{V}$ such that $v = f(u_1, u_2, \ldots u_n)$. The extension principle allows us to deduce from the n fuzzy sets of A_i a fuzzy set B on $\mathcal{V}$ through f such that:

$$\mu_B(v) = \max_{\substack{u_1, \ldots, u_n \\ v = f(u_1, \ldots, u_n)}} \min(\mu_{A_1}(u_1), \ldots, \mu_{A_n}(u_n)) \tag{8.2}$$

$$\mu_B(v) = 0 \quad \text{if} \quad f^1(v) = \varnothing \tag{8.3}$$

where $f^1(v)$ is the inverse image of v and $B = f(A_1, A_s, \ldots A_n)$

Now let A be a *fuzzy set* in $\mathcal{U}$, B be a *fuzzy set* in $\mathcal{V}$ and R a *fuzzy relation* in $\mathcal{U} \times \mathcal{V}$. *The extension principle* can be shown to be a particular case of the composition of an n-dimensional fuzzy relations by writing it as follows [Dubois and Prade, 1980]:

$$\mu_B(v) = \mu_{A \circ R}(v) = \max_x (\min(\mu_A(u), \mu_R(u, v))) \tag{8.4}$$

which can be interpreted as:

$$B = A \circ R \tag{8.5}$$

B is the *fuzzy set* deduced from A through R.

8.2.2 Equivalency of Fuzzy Relations and Fuzzy Mappings

The fuzzy differentials will be developed using fuzzy mappings. Prior to this it will be shown that a fuzzy mapping is equivalent to a fuzzy relation.

Proposition 1: A fuzzy mapping is strictly equivalent to a fuzzy relation R such that:

$$\forall\, u \in \mathcal{U}, \quad \exists\, v \in \mathcal{V}, \quad \mu_R(u, v) > 0 \tag{8.6}$$

Proof [Dubois *et al.*, 1982(a)]:

Let f be a fuzzy mapping from $\mathcal{U}$ to $\mathcal{V}$. The equivalent fuzzy relation is defined by R such that:

$$\forall (u, v) \in \mathcal{U} \times \mathcal{V}, \qquad \mu_R(u, v) = \mu_{f(u)}(v) \tag{8.7}$$

Conversely, a fuzzy relation can be viewed as a fuzzy mapping if $\mu_R(u, *)$ determines a non empty fuzzy set $f(u)$

8.2.3 Differentials of Fuzzy Mappings

Fuzzy differentiation can be considered in a number of ways:

(1) *Differentiation* at a fuzzy point of a non-fuzzy differentiable mapping [Dubois *et al.*, 1982(c)]

(2) *Differentiation* of a fuzzy mapping at an ordinary point [Dubois *et al.*, 1982(c)].

(3) *Differential equations* with fuzzy coefficients [Kandel *et al.*, 1978]

In many practical situations it is the knowledge about the mappings that is fuzzy, so it is situation (2) that will be briefly reviewed here.

The concept of a derivative is defined for a fuzzy mapping, $\tilde{f}$, from an interval $[a, b]$ to the set of fuzzy sets of $\mathcal{U}$. For all $x \in [a, b]$, $\tilde{f}$ is assumed to be:

(*i*)	normalized	$\exists\, \bar{y} \in \mathcal{U} \quad \ni \mu_{\tilde{f}}(\bar{y}) = 1$
(*ii*)	continuous	$\mu_{\tilde{f}(x)}$ is continuous
(*iii*)	support-bounded	there is some interval $[y_1, y_2]$ such that $S(\tilde{f}(x)) \subseteq [y_1, y_2]$
(*iv*)	strictly convex	$\forall s,\ \forall t > s,\ \forall y \in]s, t[,$ $\mu_{\tilde{f}(x)}(y) > \min(\mu_{\tilde{f}(x)}(s), \mu_{\tilde{f}(x)}(t))$

This definition implies that $\tilde{f}(x)$ is a fuzzy number and that $\mu_{\tilde{f}(x)}(y)$ increases when y is in the range $]-\infty, \bar{y}[$, is 1 when $y = \bar{y}$, and then continuously decreases for y in the range $]\bar{y}, +\infty[$.

For all $\alpha \in \,]0, 1[$, the α-*section* of $\tilde{f}$, defined as:

$$s_\alpha \tilde{f} = \{(x, y) \in \mathcal{U} \mid \mu_{\tilde{f}(x)}(y) = \alpha\} \tag{8.8}$$

is non-empty, because $\tilde{f}$ is continuous, normal and support-bounded.

Let $s_1 \tilde{f}$ or $\tilde{f}_1$, also called a 1-curve, denote a single mapping due to the strict convexity of $\tilde{f}$. For $\alpha < 1$, $s_\alpha \tilde{f}$, splits into two mappings, $\tilde{f}^-_\alpha$ and $\tilde{f}^+_\alpha$, called *lower* and *upper* α-*curves*, respectively, of $\tilde{f}$, and such that:

$$\forall\, x \in [a, b], \quad \tilde{f}^-_\alpha(x) < \tilde{f}_1(x) < \tilde{f}^+_\alpha(x) \tag{8.9}$$

If $\forall\, \alpha \in \,]0, 1[$ the mappings $\tilde{f}^-_\alpha$ and $\tilde{f}^+_\alpha$ are differentiable on $[a, b]$, then the membership function of the fuzzy value $\tilde{f}'(x_o)$, the *derivative* of $\tilde{f}$ at x_o, is defined by:

$$\mu_{\tilde{f}'(x_o)}(y) = \sup_{h:\, y = h'(x_o)} \mu_{\omega(\tilde{f})}(h) \tag{8.10}$$

where $$\{\omega(\tilde{f})\} = \{\tilde{f}^-_\alpha, \tilde{f}^+_\alpha \mid \alpha \in \,]0, 1[\} \cup \{\tilde{f}_1\} \tag{8.11}$$

and $$\mu_{\omega(\tilde{f})}(\tilde{f}^-_\alpha) = \mu_{\omega(\tilde{f})}(\tilde{f}^+_\alpha) = \alpha, \qquad \mu_{\omega(\tilde{f})}(\tilde{f}_1) = 1 \tag{8.12}$$

The membership value of y of the fuzzy derivative $\tilde{f}'(x_o)$ is the greatest level α of all the α-section curves whose slope at x_o is y. So $\tilde{f}'(x_o)$ is an estimation of how parallel the bundle of α-section curves is at x_o. The less fuzzy $\tilde{f}'(x_o)$, the more parallel the are curves at x_o.

8.2.4 Summary

It has been shown that a *fuzzy mapping*, f is equivalent to a *fuzzy relation*, $\boldsymbol{R}$, so the results from the section on differentials of fuzzy mappings can be applied to fuzzy relations. Therefore the differential of a fuzzy relation exists if the fuzzy relation meets the same property definitions as the fuzzy mapping (i.e. normalized, continuous, support-bounded and strictly convex) [Dubois *et al.*, 1982(c)].

8.3 An Analytical Analysis

Since derivatives have a solid basis in fuzzy set theory, this theory can be used to determine an analytical solution for the minimization of a objective function. Consider the simple control problem of equation (8.13), given $\boldsymbol{R}$ and $\boldsymbol{y}$ find $\boldsymbol{u}$ such that:

$$\boldsymbol{y} = \boldsymbol{u} \circ \boldsymbol{R} \tag{8.13}$$

Let us extend the necessary and sufficient conditions for a solution to the single input-single output from the identification problem [Pedrycz, 1988 (*max-min*); Pedrycz, 1991(b) (*max-t-norm*)] to the fuzzy cause problem for both *max-min* and *max-product*.

Let $\boldsymbol{u}=\{ u_i \mid i = \{1,2, ..., m\}\} \in \mathcal{U}$ and $\boldsymbol{y}=\{ y_j \mid j = \{1,2, ..., n\}\} \in \mathcal{Y}$ be the *fuzzy* spaces of input and output, respectively, both defined on the finite fuzzy universes of discourses. And let $\boldsymbol{R} \in \mathcal{U} \times \mathcal{Y}$ be the fuzzy relation mapping $\mathcal{U} \to \mathcal{Y}$

Proposition 1: The necessary and sufficient conditions for $\boldsymbol{u} \neq \varnothing$ for the *max-min* composition are that for every $j = \{1,2, ..., n\}$ there exists an $i = \{1,2, ..., m\}$ such that:

$$r_{ij} \geq y_j \tag{8.14}$$

Proposition 2: The necessary and sufficient conditions for $\boldsymbol{u} \neq \varnothing$ for the *max-product* composition are for every $j = \{1,2, ..., n\}$ there exists a $i = \{1,2, ..., m\}$ such that:

$$r_{ij} \geq y_j \tag{8.15}$$

and

$$\left(\frac{y_j}{r_{ij}}\right) \leq \left(\frac{y_s}{r_{is}}\right)_{s \neq j} \qquad \text{for } s \in \{1,2, ..., n\} \tag{8.16}$$

If several columns $\{ j \}$ are mapped to one row $\{ i \}$ then the ratio $\left(\frac{y_j}{r_{ij}} \right)$ must be the same for all the $\{ j \}$'s mapped to the same $\{ i \}$.

Figure 8.1 illustrates this concept, with the symbol x_* ($* = 1, 2, 3$) representing the minimum y/r ratio for the given row. Figure 8.1(a) shows a one-to-one mapping of i to j. Figure 8.1(b) illustrates that two of the minimum y/r ratios must be the same, (i.e. x_1), since there are more j columns than i rows. And Figure 8.1(c) represents the case when there are more j columns than i rows, so a minimum y/r ratio is not required for every row.

$$\begin{array}{c} i \backslash j \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} x_1 & & \\ & & x_2 \\ & x_3 & \end{pmatrix} \end{array} \qquad \begin{array}{c} i \backslash j \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} x_1 & & x_1 \\ & x_2 & \end{pmatrix} \end{array} \qquad \begin{array}{c} i \backslash j \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cc} 1 & 2 \end{array} \\ \begin{pmatrix} x_1 & \\ & \\ & x_2 \end{pmatrix} \end{array}$$

(a) (b) (c)

Figure 8.1: Illustration of Existence Condition of Proposition 2

If the necessary and sufficient conditions are followed then, from Sanchez [1976, 1977], an exact solution exists. In many and perhaps most applications this is not the case. Thus the problem becomes one of minimizing an objective function such as:

$$Q_q = \sum_{j=1}^{n} (\tilde{y}_j - y_j)^q \tag{8.17}$$

where

$$\tilde{y}_j = \bigvee_{1 \le i \le m} (\tilde{u}_i \, \mathbf{t} \, r_{ij}) \tag{8.18}$$

and $\tilde{u}_i$ is the input estimate

$\mathbf{t}$ is a *max-t-norm* composition (i.e. *max-min* or *max-product*)

q is an integer

The most commonly used objective functions are for $q = 1$ or 2. Consider the analytical calculations for each of these cases.

8.3.1 Analysis of the Minimization of Q_1

The analysis of the absolute case (i.e. $q = 1$) will minimize Q_1 with respect to $\tilde{u}$ to determine the optimum $\tilde{u}$ to minimize the absolute sum of the distance between the fuzzy actual output, y and the fuzzy estimate of the output, $\tilde{y}$.

Define:

$$Q_1 = \sum_{j=1}^{n} |\tilde{y}_j - y_j| \tag{8.19}$$

$$\min_{w.r.t.\ \tilde{u}} Q_1$$

$$\frac{\partial Q_1}{\partial \tilde{u}_s} = \frac{\partial}{\partial \tilde{u}_s} \sum_{j=1}^{n} |\tilde{y}_j - y_j| \tag{8.20}$$

$$= \sum_{j=1}^{n} \begin{cases} \dfrac{\partial \tilde{y}_j}{\partial \tilde{u}_s} & \text{if} \quad \tilde{y}_j > y_j \\ 0 & \text{if} \quad \tilde{y}_j = y_j \\ -\dfrac{\partial \tilde{y}_j}{\partial \tilde{u}_s} & \text{if} \quad \tilde{y}_j < y_j \end{cases} \tag{8.21}$$

and

$$\frac{\partial \tilde{y}_j}{\partial \tilde{u}_s} = \frac{\partial}{\partial \tilde{u}_s} \bigvee_{1 \le i \le m} (\tilde{u}_i \,\mathrm{t}\, r_{ij}) \tag{8.22}$$

$$= \frac{\partial}{\partial \tilde{u}_s} \{ \bigvee_{i \ne s} (\tilde{u}_i \,\mathrm{t}\, r_{ij}) \vee (\tilde{u}_s \,\mathrm{t}\, r_{sj}) \} \tag{8.23}$$

If $\mathbf{t}$ = *product*:

$$\frac{\partial \tilde{y}_j}{\partial \tilde{u}_s} = \begin{cases} r_{sj} & \text{if} \quad \tilde{u}_s \cdot r_{sj} \geq \bigvee_{i \neq s} (\tilde{u}_i \cdot r_{ij}) \\ 0 & \text{otherwise} \end{cases} \tag{8.24}$$

If $\mathbf{t}$ = *min*:

$$\frac{\partial \tilde{y}_j}{\partial \tilde{u}_s} = \begin{cases} 1 & \text{if} \quad \tilde{u}_s \wedge r_{sj} \geq \bigvee_{i \neq s} (\tilde{u}_i \wedge r_{ij}) \\ & \text{and} \quad \tilde{u}_s \leq r_{sj} \\ 0 & \text{otherwise} \end{cases} \tag{8.25}$$

for $s = \{1,2, ..., m\}$.

Based on these equations, $\frac{\partial \tilde{y}_j}{\partial \tilde{u}_s} = 0$ only if $\tilde{u}_i = 0$ for all i or under the conditions listed.

Now consider the equation analytically.

$$\frac{\partial Q}{\partial \tilde{u}_s} = \frac{\partial}{\partial \tilde{u}_s} \sum_{j=1}^{n} |\tilde{y}_j - y_j| = 0 \tag{8.26}$$

$$= \frac{\partial}{\partial \tilde{u}_s} \sum_{j=1}^{n} |\bigvee_{1 \leq i \leq m} (\tilde{u}_i \,\mathbf{t}\, r_{ij}) - y_j| \tag{8.27}$$

If an exact solution exists, then:

$$\left| \bigvee_{1 \le i \le m} (\tilde{u}_i \, t \, r_{ij}) - y_j \right| = 0 \qquad (8.28)$$

$$\boldsymbol{u} \, t \, \boldsymbol{R} - \boldsymbol{y} = 0$$

$$\boldsymbol{u} \, t \, \boldsymbol{R} = \boldsymbol{y}$$

$$\therefore \quad \boldsymbol{u} = \boldsymbol{y} \circledast \boldsymbol{R} \qquad (8.29)$$

where $\circledast$ is the appropriate inverse for the t operator

which corresponds to the results of Sanchez [1976,1977]. This analysis also shows that for the case of $q = 1$ the analytical derivative reduces to the exact solution, when it exists.

8.3.2 Analysis of the Minimization of Q_2

The analytical analysis of the quadratic case (i.e. $q = 2$) will minimize Q_2 with respect to $\tilde{u}$ to determine the optimum $\tilde{y}$ to minimize the sum of the square of the distance between the fuzzy actual output, y and the fuzzy estimate of the output, $\tilde{y}$.

Define:

$$Q_2 = \sum_{j=1}^{n} (\tilde{y}_j - y_j)^2 \qquad (8.30)$$

$$\min_{w.r.t.\ \tilde{u}} Q_2$$

$$\frac{\partial Q_2}{\partial \tilde{u}_s} = \frac{\partial}{\partial \tilde{u}_s} \sum_{j=1}^{n} \left(\bigvee_{1 \le i \le m} (\tilde{u}_i \, t \, r_{ij}) - y_j \right)^2 \qquad (8.31)$$

$$= 2 \sum_{j=1}^{n} \left(\bigvee_{1 \le i \le m} (\tilde{u}_i \, t \, r_{ij}) - y_j \right) \cdot \frac{\partial}{\partial \tilde{u}_s} \bigvee_{1 \le i \le m} (\tilde{u}_i \, t \, r_{ij}) = 0 \qquad (8.32)$$

Based on this equation, there are three possible ways of achieving a minimum:

(1) $$\sum_{j=1}^{n} \left(\bigvee_{1 \le i \le m} (\tilde{u}_i \, \mathrm{t} \, r_{ij}) - y_j \right) = 0 \qquad (8.33)$$

(2) $$\frac{\partial}{\partial \tilde{u}_s} \bigvee_{1 \le i \le m} (\tilde{u}_i \, \mathrm{t} \, r_{ij}) = 0 \qquad (8.34)$$

(3) $$\sum_{j=1}^{n} \left(\bigvee_{1 \le i \le m} (\tilde{u}_i \, \mathrm{t} \, r_{ij}) - y_j \right) \text{ and } \frac{\partial}{\partial \tilde{u}_s} \bigvee_{1 \le i \le m} \tilde{u}_i \, \mathrm{t} \, r_{ij} = 0 \text{ are orthogonal} \qquad (8.35)$$

If case (1) is true, then:

$$\sum_{j=1}^{n} \left(\bigvee_{1 \le i \le m} (\tilde{u}_i \, \mathrm{t} \, r_{ij}) - y_j \right) = 0 \qquad (8.36)$$

$$\boldsymbol{u} \, \mathrm{t} \, \boldsymbol{R} - \boldsymbol{y} = 0$$

$$\boldsymbol{u} \, \mathrm{t} \, \boldsymbol{R} = \boldsymbol{y}$$

$$\therefore \quad \boldsymbol{u} = \boldsymbol{y} \circledast \boldsymbol{R} \qquad (8.37)$$

where $\circledast$ is the appropriate inverse for the t operator

which again are the results of Sanchez [1976,1977] and proves that the analytical derivative reduces to the exact solution, when it exists.

If case (2) is true, then:

$$\frac{\partial}{\partial \tilde{u}_s} \bigvee_{1 \le i \le m} (\tilde{u}_i \, \mathrm{t} \, r_{ij}) = \frac{\partial}{\partial \tilde{u}_s} \{ (\bigvee_{i \ne s} \tilde{u}_i \, \mathrm{t} \, r_{ij}) \vee (\tilde{u}_s \, \mathrm{t} \, r_{sj}) \} \qquad (8.38)$$

If t = *product*:

$$\frac{\partial}{\partial \tilde{u}_s} \bigvee_{1 \le i \le m} (\tilde{u}_i \, t \, r_{ij}) = \begin{cases} r_{sj} & \text{if} \quad \tilde{u}_s \cdot r_{sj} \ge \bigvee_{i \ne s} (\tilde{u}_i \cdot r_{ij}) \\ 0 & \text{otherwise} \end{cases} \tag{8.39}$$

If t = *min*:

$$\frac{\partial}{\partial \tilde{u}_s} \bigvee_{1 \le i \le m} (\tilde{u}_i \, t \, r_{ij}) = \begin{cases} 1 & \text{if} \quad \tilde{u}_s \wedge r_{sj} \ge \bigvee_{i \ne s} (\tilde{u}_i \wedge r_{ij}) \\ & \text{and} \quad \tilde{u}_s \le r_{sj} \\ 0 & \text{otherwise} \end{cases} \tag{8.40}$$

If case (3) is true, and the vectors are orthogonal, then a global minimum will not be obtained.

As seen from the minimization criteria for $q = 1$:

$$\frac{\partial Q_1}{\partial \tilde{u}_s} = = \frac{\partial}{\partial \tilde{u}_s} \sum_{j=1}^{n} | \bigvee_{1 \le i \le m} (\tilde{u}_i \, t \, r_{ij}) - y_j | \tag{8.41}$$

and for $q = 2$:

$$\frac{\partial Q_2}{\partial \tilde{u}_s} = 2 \cdot \sum_{j=1}^{n} (\bigvee_{1 \le i \le m} (\tilde{u}_i \, t \, r_{ij}) - y_j) \cdot \frac{\partial}{\partial \tilde{u}_s} \bigvee_{1 \le i \le m} (\tilde{u}_i \, t \, r_{ij}) = 0 \tag{8.42}$$

to achieve an analytical minimum for $\tilde{u}_s$ it is necessary to separate the value $\tilde{u}_s$ from each equation. This is not possible for two reasons:

(1) there is no direct inverse for the *max* function

(2) the summation over which the expression is calculated.

Additionally, in control problems, often the emphasis is on minimization of the absolute discrete difference (q=1) or the discrete distance squared (q=2) as follows:

$$J_q = |\mathbf{y} - \tilde{\mathbf{y}}|^q \tag{8.43}$$

8.4 Numerical Methods

This section reviews three (3) minimization algorithms which numerically optimize a fuzzy performance index, neural learning minimization of Q_1 [Pedrycz, 1991(b); Valente de Oliveira, 1993], quasi-Newton minimization of Q_2 [Pedrycz, 1983] and probabilistic descent minimization of Q_2 [Ikoma *et al.*, 1990].

The calculations that are presented in this Chapter minimize the causal performance index with respect to the input, $\boldsymbol{u}$, however, minimization can also be performed under identification with respect to the relational matrix $\boldsymbol{R}$. A review of these latter calculations is provided by Bourke *et al.* [1995]. The identification results of Bourke *et al.* [1995] are provided in Section 8.5 and then the identified matrices and the causal performance criterion is used to follow a set trajectory. Results of the minimization algorithms with a causal criterion are presented in section 8.6.

8.4.1 Neural Learning Minimization of Q_1

A numerical method for solving the optimization problem with $q = 1$:

$$Q_1 = \sum_{j=1}^{n} |(\tilde{y}_j - y_j)|^1 \tag{8.44}$$

for the case of identification is provided by Pedrycz [1991(b)] and then expanded by Valente de Oliveira [1993]. It should be noted that the procedure given by these authors can easily be extended to the fuzzy cause problem. These papers maximize Q_1 using the following equality index:

$$Q_1 = (y_j \equiv \tilde{y}_j) = \begin{cases} 1 + y_j - \tilde{y}_j & \text{if} \quad \tilde{y}_j > y_j \\ 0 & \text{if} \quad \tilde{y}_j = y_j \\ 1 + \tilde{y}_j - y_j & \text{if} \quad \tilde{y}_j < y_j \end{cases} \tag{8.45}$$

This maximization is the exact inverse of the minimization of Q_1, (analyzed in Section 8.3.1 Analysis of the Minimization of Q_1) and defined as:

$$Q_1 = |y_j - \tilde{y}_j| = \begin{cases} \tilde{y}_j - y_j & \text{if } \tilde{y}_j > y_j \\ 0 & \text{if } \tilde{y}_j = y_j \\ y_j - \tilde{y}_j & \text{if } \tilde{y}_j < y_j \end{cases} \tag{8.46}$$

A brief extension of the neural learning method [Pedrycz, 1991(b); Valente de Oliveira, 1993] for the fuzzy cause problem with either a *max-product* or *max-min* composition is presented here. The paper by Valente de Oliveira [1993] deals only with identification. Additionally, the learning of relational matrix in this same paper takes place in the presence of a bias term which the paper later shows can be ignored. Thus the bias term will also be ignored for this analysis. So for the fuzzy control problem:

$$\tilde{y} = \tilde{u} \circ x \circ R \tag{8.47}$$

where x is the fuzzy state, and $\circ$ represents the *max-product* or the *max-min* composition, the objective function is:

$$Q_1 = \sum_{l=1}^{n} \left(\bigvee_{\substack{1 \le i \le m \\ 1 \le j \le n}} (\tilde{u}_i * x_j * r_{ijl}) - y_l \right)^1 \tag{8.48}$$

The minimization: $\min_{w.r.t.\ \tilde{u}} Q_1$

is obtained by:

$$\frac{\partial Q_1}{\partial \tilde{u}} = 0 \tag{8.49}$$

Calculation of this derivative results in the following system of equations:

$$\frac{\partial Q_1}{\partial \tilde{u}_s} = \frac{\partial}{\partial \tilde{u}_s} \sum_{l=1}^{n} |\tilde{y}_l - y_l| \tag{8.50}$$

$$= \sum_{l=1}^{n} \begin{cases} \dfrac{\partial \tilde{y}_l}{\partial \tilde{u}_s} & \text{if} \quad \tilde{y}_l > y_l \\ 0 & \text{if} \quad \tilde{y}_l = y_l \\ -\dfrac{\partial \tilde{y}_l}{\partial \tilde{u}_s} & \text{if} \quad \tilde{y}_l < y_l \end{cases} \tag{8.51}$$

$$\frac{\partial \tilde{y}_l}{\partial \tilde{u}_s} = \frac{\partial}{\partial \tilde{u}_s} \bigvee_{\substack{1 \le i \le m \\ 1 \le j \le n}} (\tilde{u}_i * x_j * r_{ijl}) \tag{8.52}$$

$$= \frac{\partial}{\partial \tilde{u}_s} \{ \bigvee_{\substack{i \ne s \\ 1 \le j \le n}} (\tilde{u}_i * x_j * r_{ijl}) \vee (\bigvee_{1 \le j \le n} \tilde{u}_s * x_j * r_{sjl}) \} \tag{8.53}$$

If $*$ = *product*:

$$\frac{\partial \tilde{y}_l}{\partial \tilde{u}_s} = \begin{cases} \bigvee_{1 \le j \le n} (x_j \cdot r_{sjl}) & \text{if} \quad \bigvee_{1 \le j \le n} (\tilde{u}_s \cdot x_j \cdot r_{sjl}) \ge \bigvee_{\substack{i \ne s \\ 1 \le j \le n}} (\tilde{u}_i \cdot x_j \cdot r_{ijl}) \\ 0 & \text{otherwise} \end{cases} \tag{8.54}$$

If $*$ = *min*:

$$\frac{\partial \tilde{y}_l}{\partial \tilde{u}_s} = \begin{cases} 1 & \text{if} \quad \bigvee_{1 \le j \le n} (\tilde{u}_s \wedge x_j \wedge r_{sjl}) \ge \bigvee_{\substack{i \ne s \\ 1 \le j \le n}} (\tilde{u}_i \wedge x_j \wedge r_{ijl}) \\ & \text{and} \quad \bigvee_{1 \le j \le n} (x_j \wedge r_{sjl}) \ge \tilde{u}_s \\ 0 & \text{otherwise} \end{cases} \tag{8.55}$$

for $s = 1, 2, \ldots, m$.

The estimate of the fuzzy input, $\tilde{u}$, is updated iteratively according to the formula:

$$\tilde{u}^{(k+1)} = \tilde{u}^{(k)} + \eta[\Delta\tilde{u}^{(k+1)} + \alpha\Delta\tilde{u}^{(k)}] \tag{8.56}$$

where $$\Delta\tilde{u} = -\frac{\partial Q_1}{\partial \tilde{u}} \tag{8.57}$$

In equation (8.56) the superscript (k) refers to the iteration number, $0 < \eta \leq 1$ is the learning rate or step size parameter, and $\alpha \geq 0$ is the momentum term.

The paper by Valente de Oliveira [1993] illustrates the functionality of the maximization algorithm with the Box-Jenkins [1970] data, however a calculated value of the resulting Q_1 is not provided. The graphical comparison for the identification (Figure 8; Valente de Oliveira [1993]) between actual and predicted results shows *reasonable* agreement between the two values, however, the author points out that to achieve this agreement, the maximum value of the relational matrix, $\boldsymbol{R}$, was allows to exceed the unit interval [0, 1]. As will be shown in the next section, when this algorithm is required to remain within the unit interval consistent with the definition of a fuzzy relation, the identification results are within the range of identification techniques considered to provide *good* results. Valente de Oliveira [1993] also provided a method to improve the convergence property of their batch identification algorithm. This improvement has not been included in the identification analysis for this algorithm so that all three numerical algorithms are tested on the same level. Results given in Section 8.5 confirm that the improvement is in fact required.

8.4.2 Quasi-Newton Minimization with Q_2

Numerical methods for solving the basic optimization problem for $q = 2$:

$$Q_2 = \sum_{l=1}^{n} (\tilde{y}_l - y_l)^2 \tag{8.58}$$

are provided by Pedrycz [1983], who presents algorithms for fuzzy identification and fuzzy cause for both the *max-min* and *max-product* compositions. The numerical solution algorithm is a modification of the iterative Newton method for a derivative called the quasi-Newton method. The main adaptation of this quasi-Newton method over the original Newton method is that the inversion of a second derivative of Q is not required. Wang [1993] presents the properties of this quasi-Newton method for various composition, however, the *max-product* composition is not included in this analysis.

A brief review of the quasi-Newton method [Pedrycz, 1983] for fuzzy cause problem with either a *max-product* or *max-min* composition is presented here. For the fuzzy control problem:

$$\tilde{y} = \tilde{u} \circ x \circ R \tag{8.59}$$

where x is the fuzzy state, and $\circ$ represents the *max-product* or the *max-min* composition, the objective function is:

$$Q_2 = \sum_{l=1}^{n} \Big(\bigvee_{\substack{1 \le i \le m \\ 1 \le j \le n}} (\tilde{u}_i * x_j * r_{ijl}) - y_l \Big)^2 \tag{8.60}$$

The minimization: $\min\limits_{w.r.t.\ \tilde{u}} Q_2$

is obtained by:

$$\frac{\partial Q_2}{\partial \tilde{u}} = 0 \tag{8.61}$$

Calculation of this derivative results in the following system of equations:

$$\frac{\partial Q_2}{\partial \tilde{u}_s} = \frac{\partial}{\partial \tilde{u}_s} \sum_{l=1}^{n} \Big(\bigvee_{1 \le j \le n} (\tilde{u}_s * x_j * r_{sjl}) - y_l \Big)^2 \tag{8.62}$$

$$= 2 \sum_{l=1}^{n} \Big(\bigvee_{1 \le j \le n} (\tilde{u}_s * x_j * r_{sjl}) - y_l \Big) \frac{\partial \tilde{y}_l}{\partial \tilde{u}_s} \tag{8.63}$$

where

$$\frac{\partial \tilde{y}_l}{\partial \tilde{u}_s} = \frac{\partial}{\partial \tilde{u}_s} \bigvee_{1 \le j \le n} (\tilde{u}_s * x_j * r_{sjl}) \tag{8.64}$$

$$= \frac{\partial}{\partial \tilde{u}_s} \Big\{ \bigvee_{i \ne s} (\tilde{u}_i * x_j * r_{ijl}) \vee (\tilde{u}_s * x_j * r_{sjl}) \Big\} \tag{8.65}$$

If $* = product$:

$$\frac{\partial \tilde{y}_l}{\partial \tilde{u}_s} = \begin{cases} \bigvee_{1\le j\le n} (x_j \cdot r_{sjl}) & \text{if} \quad \bigvee_{1\le j\le n} (\tilde{u}_s \cdot x_j \cdot r_{sjl}) \ge \bigvee_{\substack{i\ne s \\ 1\le j\le n}} (\tilde{u}_i \cdot x_j \cdot r_{ijl}) \\ 0 & \text{otherwise} \end{cases} \tag{8.66}$$

If $* = min$:

$$\frac{\partial \tilde{y}_l}{\partial \tilde{u}_s} = \begin{cases} 1 & \text{if} \quad \bigvee_{1\le j\le n} (\tilde{u}_s \wedge x_j \wedge r_{sjl}) \ge \bigvee_{i\ne s} (\tilde{u}_i \wedge x_j \wedge r_{ijl}) \\ & \text{and} \quad \bigvee_{1\le j\le n} (x_j \wedge r_{sjl}) \ge \tilde{u}_s \\ 0 & \text{otherwise} \end{cases} \tag{8.67}$$

for $s = 1, 2, \ldots, m$.

The quasi-Newton iteration method to solve for $\tilde{u}$ is a follows:

$$\tilde{u}^{(k+1)} = \tilde{u}^{(k)} - \alpha_k \, \Delta\tilde{u}^{(k)} \tag{8.68}$$

where

$$\Delta\tilde{u} = \frac{\partial Q_2}{\partial \tilde{u}} \tag{8.69}$$

The superscript, (k) in equation (8.68) is defined the same as for neural learning. The scalar multiplier, α_k, which replaces the inverted second derivative of the Newton method, is a function of the number of iterations required for convergence. This simplification results in a slow linear convergence rate, rather than the quadratic rate of the Newton method, however, Pedrycz [1983] claims that good convergence can be achieved with the proper choice of α_k.

In order to obtain a good convergence rate, α_k was defined as a non increasing gain factor:

$$\alpha_k = \frac{1}{c + k^w} \tag{8.70}$$

where $$c = \max\left\{ 2\sum_{l=1}^{n}\left[\bigvee_{1\le j\le n} (\tilde{u}_s * x_j * r_{sjl}) - y_l \right] \frac{\partial \tilde{y}_l}{\partial \tilde{u}_s} \right\} = 2n \tag{8.71}$$

k is the iteration number

and $w \geq 0$ is chosen to based on experience to achieve *good* convergence properties and to avoid oscillation.

This identification algorithm methodology was not tested by Pedrycz [1983] with the Box-Jenkins data [1970], however, testing with other data showed that under that same tuning parameters, the *max-product* composition was slower to converge than the *max-min*, but convergence was to a smaller value of Q_2.

8.4.3 Probabilistic Descent Minimization with Q_2

Ikoma *et. al.* [1993] propose the probabilistic descent method for the same minimization criteria, Q_2, as Pedrycz [1983], and applied this methodology for identification only under the *max-min* composition. The probabilistic descent method presented by these authors is computationally very similar to the quasi-Newton method. The main feature that differentiates this method of calculating the derivative is the replacement of the non-differentiable *maximum* and *minimum* functions by a differentiable approximations.

The probabilistic descent methodology is briefly review here for the fuzzy cause problem with either a *max-product* or *max-min* composition. Again, for the fuzzy control problem:

$$\tilde{y} = \tilde{u} \circ x \circ R \tag{8.72}$$

where x is the fuzzy state, and $\circ$ represents the *max-product* or the *max-min* composition, the objective function is:

$$Q_2 = \sum_{l=1}^{n} \Big(\bigvee_{\substack{1\le i\le m \\ 1\le j\le n}} (\tilde{u}_i * x_j * r_{ijl}) - y_l \Big)^2 \tag{8.73}$$

The minimization: $\min_{w.r.t.\ \tilde{u}} Q_2$

is obtained by:

$$\frac{\partial Q_2}{\partial \tilde{u}} = 0 \tag{8.74}$$

Calculating of this derivative:

$$\frac{\partial Q_2}{\partial \tilde{u}_s} = \frac{\partial}{\partial \tilde{u}_s} \sum_{l=1}^{n} \left(\left(\bigvee_{\substack{1 \le i \le m \\ 1 \le j \le n}} \tilde{u}_i * x_j * r_{ijl}\right) - y_l\right)^2 \tag{8.75}$$

$$= 2 \cdot \sum_{l=1}^{n} \left(\left(\bigvee_{\substack{1 \le i \le n \\ 1 \le j \le n}} \tilde{u}_i * x_j * r_{ijl}\right) - y_l\right) \cdot \frac{\partial}{\partial \tilde{u}_s} \left(\bigvee_{\substack{1 \le i \le m \\ 1 \le j \le n}} \tilde{u}_i * x_j * r_{ijl}\right) = 0 \tag{8.76}$$

It is required that the derivative be taken over the *maximum*, which is piecewise differentiable.

Consider the derivatives taken over the *maximum* and *minimum* operations given as follows:

$$\frac{\partial}{\partial x}[\max(x, y)] = \begin{cases} 1, & x > y \\ 0, & x < y \end{cases} \tag{8.77}$$

$$\frac{\partial}{\partial x}[\min(x, y)] = \begin{cases} 1, & x < y \\ 0, & x > y \end{cases} \tag{8.78}$$

Both derivatives are not defined at $x = y$. This means that the probabilistic descent method cannot be applied directly to equation (8.76) when the *max* is defined as in equation (8.77) In order to overcome this problem, it has been proposed to define the derivatives as follows [Ikoma *et al.*, 1993]:

$$\frac{\partial}{\partial x}[\max(x, y)] = \begin{cases} 1, & x \ge y \\ 0, & x < y \end{cases} \tag{8.79}$$

$$\frac{\partial}{\partial x}[\min(x, y)] = \begin{cases} 1, & x \le y \\ 0, & x > y \end{cases} \tag{8.80}$$

Both these formula return 0 and 1 and can be treated a two-valued predicates (i.e. 0 or 1). However, the use of these formula can cause problems in that the minimization may find only local minima. As well, under specific initial conditions a zero value could result for all predicates [Ikoma *et al.*, 1993]. To avoid these conditions these authors proposed approximating the discontinuous derivatives of equations (8.79) and (8.80) by smooth functions. The functions that were considered are as follows:

$$\frac{\partial h_{max}(a,x)}{\partial x} = \frac{1}{1+e^{-(x-a)/k}} \tag{8.81}$$

$$\frac{\partial h_{min}(a,x)}{\partial x} = \frac{1}{1+e^{(x-a)/k}} \tag{8.82}$$

where $k > 0$ is used to control the shape of the function. Figure 8.2 (a) and (b) show the plots of equations (8.81) and (8.82), respectively, for various k along with the original derivatives, equations (8.77) and (8.78). Although the plots for the original derivatives and the approximation when $k = 0.01$ are not identical, the difference would be negligible in a control setting.

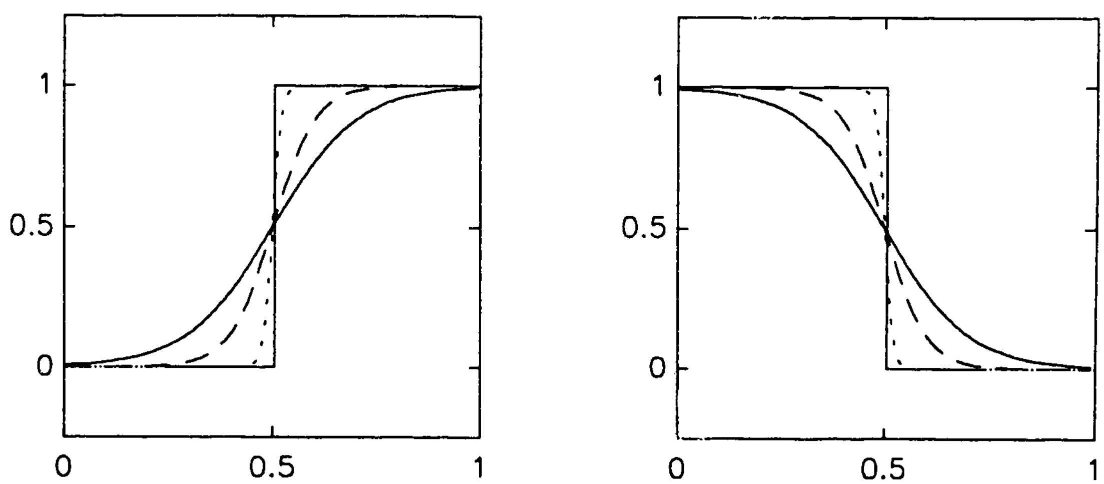


Figure 8.2: (a) Derivative of $h_{max}(.5, x)$, equation (8.78);
(b) Derivative of $h_{min}(.5, x)$, equation (8.79)
($k = 0.1$, —— ; $k = 0.05$, – – – ; $k = 0.01$, · · ·)

Making the appropriate substitutions into equation (8.76) produces the following equations:

If $*$ = *product*:

$$\frac{\partial Q_2}{\partial \tilde{u}_s} = 2\sum_{l=1}^{n} \left(\left(\bigvee_{\substack{1\le i\le m \\ 1\le j\le n}} \tilde{u}_i \cdot x_j \cdot r_{ijl} \right) - y_l \right) \cdot \bigvee_{1\le j\le n} (x_j \cdot r_{sjl}) \frac{1}{1+e^{-(b_s-a)/k}} \tag{8.83}$$

where $$b_s = \bigvee_{1\le j\le n} (\tilde{u}_s \cdot x_j \cdot r_{sjl}) \tag{8.84}$$

$$a = \bigvee_{\substack{i\ne s \\ 1\le j\le n}} (\tilde{u}_i \cdot x_j \cdot r_{ijl}) \tag{8.85}$$

If $*$ = *min*:

$$\frac{\partial Q_2}{\partial \tilde{u}_s} = 2\sum_{l=1}^{n} \left(\left(\bigvee_{\substack{1\le i\le m \\ 1\le j\le n}} \tilde{u}_i \wedge \cdot x_j \wedge r_{ijl} \right) - y_l \right) \frac{1}{1+e^{(d_s-c)/k}} \cdot \frac{1}{1+e^{-(b_s-a)/k}} \tag{8.86}$$

where $$b_s = \bigvee_{1\le j\le n} (\tilde{u}_s \wedge x_j \wedge r_{sjl}) \tag{8.87}$$

$$a = \bigvee_{\substack{i\ne s \\ 1\le j\le n}} (\tilde{u}_i \wedge x_j \wedge r_{ijl}) \tag{8.88}$$

$$d_s = \tilde{u}_s \tag{8.89}$$

$$c = \bigvee_{\substack{i\ne s \\ 1\le j\le n}} (x_j \wedge r_{ijl}) \tag{8.90}$$

The updated value of $\tilde{u}$ is calculated iteratively by the formula:

$$\tilde{u}^{(k+1)} = \tilde{u}^{(k)} + \varepsilon \, \Delta \tilde{u}^{(k)} \tag{8.91}$$

where $$\Delta \tilde{u} = -\frac{\partial Q_1}{\partial \tilde{u}} \tag{8.92}$$

In equation (8.91) the superscript (k) refers to the iteration number and $0 < \eta \leq 1$ is a small positive constant used to scale the modification term, $\tilde{u}$.

The strength of the probabilistic descent algorithm [Ikoma *et al.*, 1993] *is that it avoids the problem of converging to local minima and other difficulties caused by specific initial conditions that are inherent in the other two methodologies.* Testing of the identification algorithm in the paper by Ikoma *et al.*, [1993] is performed with exact data or under ideal conditions. The ability of this algorithm using plant data is not demonstrated in the paper.

8.5 Comparison of Identification Minimization Results

In order to evaluate the effectiveness of each of these strategies in a predictive capacity, the *minimization* algorithms were considered for the case of identification. The assumption being made is:

if the performance is good from an identification perspective, it will be good from a causal perspective.

The identification algorithms associated with neural learning [Valente de Oliveira, 1993; Pedrycz, 1991(b)], quasi-Newton minimization [Pedrycz, 1983] and probabilistic descent minimization [Ikoma *et al.*, 1993] are compared using the same data and performance indexes used for the identification analysis in Chapter 7. In doing so these iterative algorithm can also be compared versus the algorithms discussed and analyzed in Chapter 7.

For each of the cases being considered, the initial $\boldsymbol{R}$ value used to initiate the learning was set at:

(1) $\boldsymbol{R} = [0]_{m \times n \times n}$

(2) $\boldsymbol{R} = [.5]_{m \times n \times n}$

(3) $\boldsymbol{R} = [1]_{m \times n \times n}$

Studying these initial values of $\boldsymbol{R}$ will indicate which, if any, algorithms experience difficulties caused by specific initial conditions.

For each of the algorithms tested, the data was learned in *batch* over 50 iterations and the convergence factors were *tuned* to improve the performance of the algorithm. The convergence factors were chosen large enough to ensure good convergence properties and small enough to avoid oscillations. Graphical results of the convergence trajectories of these simulations are given in Figure 8.3 to 8.6. Comparative data for the optimization algorithms are presented in Tables 8.1 and 8.2.

Consider the optimized results of the *max-product* composition. For all methods, neural learning, quasi-Newton and probabilistic descent, the algorithms performed well with the resulting $J_2 < 0.5$. *Tuning* was performed for $\boldsymbol{R} = [.5]_{mxnxn}$ and then the other two initial $\boldsymbol{R}$'s were optimized based on the convergence factor *tuned* for $\boldsymbol{R} = [.5]_{mxnxn}$ so comparison could be made using the same tuning basis. Between the three methods, the convergence rates for quasi-Newton and probabilistic descent, were virtually the same, which seems reasonable since both methods are minimizing the same objective function. The neural learning required more time for convergence but the final results are equivalent to the other two faster methodologies. This slower learning rate confirms that need of a method to improve the convergence property of the neural learning algorithm which is provided in the work by Valente de Oliveira [1993].

For the example tested, neither of the *max-product* optimizations of neural learning or quasi-Newton methods exhibited problems with a local minimum, which the probabilistic descent method has been formulated to overcome.

As mentioned previously, the paper by Valente de Oliveira [1993] illustrates the functionality of a maximization algorithm for neural learning with the *max-product* composition, using the Box-Jenkins [1970] data, but includes no calculated value of the resulting Q_1. The author states that to achieve *good* agreement between that actual and predicted data the maximum value of the relational matrix, $\tilde{\boldsymbol{R}}$, was allows to exceed the unit interval [0, 1]. However, as is evident by the results in Table 8.1, when the neural learning algorithm is required to remain within the unit interval, consistent with the definition of a fuzzy relation, the identification results are within the range considered to provide *good* results.

The *max-min* composition results are sparse because the algorithm is unable to converge for an initial $\boldsymbol{R} = [1]_{mxnxn}$. This non-convergence for initial $\boldsymbol{R} = [1]_{mxnxn}$, for each method, is due to the second condition on the derivative for the *max-min* composition. The conditions for calculation of the *max-min* composition derivative for neural learning and quasi-Newton are as follows:

$$\frac{\partial \tilde{y}_l}{\partial \tilde{r}_{vsl}} = \begin{cases} 1 & \text{if} \quad u_v \wedge x_s \wedge \tilde{r}_{vsl} \geq \bigvee_{\substack{i \neq v \\ j \neq s}} (u_i \wedge x_j \wedge \tilde{r}_{ijl}) \\ & \text{and} \quad u_v \wedge x_s \geq \tilde{r}_{vsl} \\ 0 & \text{otherwise} \end{cases} \tag{8.93}$$

It is easy to see from equation (8.93) that if $\boldsymbol{R} = [\mathbf{1}]$, then from the second condition of this equation:

$$\frac{\partial \tilde{y}_t}{\partial r_{vst}} = 0 \qquad \text{for all } v, s \text{ and } t. \tag{8.94}$$

and no change in $\boldsymbol{R}$ can occur. Although this second condition is not explicitly stated in the probabilistic descent method, its is implicit in the calculation of the exponential:

$$\frac{1}{1+e^{(d_{vst}-c)/k}} \tag{8.95}$$

where

$$d_{vsl} = r_{vsl} \tag{8.96}$$

$$c = (u_v \wedge x_s) \tag{8.97}$$

This second condition for the derivative is not present in the *max-product* composition because of the *product* calculation.

The inability of the *max-min* composition algorithms to converge for an initial $\boldsymbol{R} = [\mathbf{0}]_{m \times n \times n}$ is due to the aggressive setting of the convergence term. The large convergence term causes a large initial oscillation in $\boldsymbol{R}$ and the updated $\boldsymbol{R}$ is outside the unit interval. All the optimization algorithms check to ensure each $\boldsymbol{R}$ remains in the unit interval. So after checking the update of the first oscillation of $\boldsymbol{R}$, the update is reset to $\boldsymbol{R} = [\mathbf{1}]_{m \times n \times n}$ and hence the algorithm is unable to converge. A smaller value of the convergence factor could be chosen for the *max-min* composition for an initial $\boldsymbol{R} = [\mathbf{0}]_{m \times n \times n}$, however, the convergence rate would be slower.

The neural learning algorithm for the *max-min* composition found a local minimum, as shown in Figure 8.6, however, the algorithm was able to move off of the local minimum due to aggressive *tuning* of the convergence factor.

For all methodologies with the *max-min* composition, the final values of $\boldsymbol{J}_2 > 0.5$. So not only is the *max-min* composition more sensitive to the tuning factor and initial values of $\boldsymbol{R}$, than the *max-product* composition, but the final results also are *poorer*.

Cross-learning of the relational matrix did not produce the same improvements noted with the non-optimized algorithms of Chapter 7. *Max-product* prediction was never improved with cross-learning. *Max-min* prediction improved only slightly for the neural learning and probabilistic descent methods.

As with the non-optimized identification algorithms, discussed in Chapter 7, the optimized algorithms can be ranked, based on the J_2 criterion. The ranking, given in Table 8.3 is based on the *max-product* composition J_2 values for an initial R = [.5]. The results show that the probabilistic descent method is the *best*, although the results of all three methods are very close.

Method	Composition	ID Method	Initial R	J_1	J_2	Q_1	Q_2
Neural Learning	*Max-Prod*	*Prod*	[0]	0.4528	0.3628	0.3784	0.0986
		$\eta = 0.009$	[.5]	0.4467	0.3530	0.3759	0.0967
		$\alpha = 0$	[1]	0.4591	0.3852	0.3875	0.0974
	Max-Min	*Min*	[0]	n/a	n/a	n/a	n/a
		$\eta = 0.005$	[.5]	0.6244	0.8297	0.4171	0.0867
		$\alpha = 0$	[1]	n/a	n/a	n/a	n/a
Quasi-Newton	*Max-Prod*	*Prod*	[0]	0.4442	0.3322	0.3818	0.0971
		$w = 1$	[.5]	0.4626	0.3630	0.3882	0.0963
			[1]	0.4911	0.4261	0.3979	0.0971
	Max-Min	*Min*	[0]	n/a	n/a	n/a	n/a
		$w = 3$	[.5]	0.5901	0.7216	0.3837	0.0825
			[1]	n/a	n/a	n/a	n/a
Probabilistic Descent	*Max-Prod*	*Prod*	[0]	0.4357	0.3224	0.3800	0.0970
		$\varepsilon = 0.15$	[.5]	0.4505	0.3410	0.3822	0.0962
		$k = 0.01$	[1]	0.4650	0.3760	0.3867	0.0965
	Max-Min	*Min*	[0]	n/a	n/a	n/a	n/a
		$\varepsilon = 0.05$	[.5]	0.5690	0.7150	0.3690	0.0802
		$k = 0.01$	[1]	n/a	n/a	n/a	n/a

Table 8.1: Comparison of Identification Algorithms with Optimization

Method	Composition	ID Method	Initial R	J_1	J_2	Q_1	Q_2
Neural Learning	*Max-Prod*	*Min*	[0]	n/a	n/a	n/a	n/a
		$\eta = 0.005$	[.5]	0.4645	0.4232	0.4251	0.1188
		$\alpha = 0$	[1]	n/a	n/a	n/a	n/a
	Max-Min	*Prod*	[0]	0.4297	0.3264	0.3774	0.1008
		$\eta = 0.009$	[.5]	0.4994	0.5173	0.3940	0.1030
		$\alpha = 0$	[1]	0.5706	0.7307	0.4245	0.1075
Quasi-Newton	*Max-Prod*	*Min*	[0]	n/a	n/a	n/a	n/a
		$w = 3$	[.5]	0.5049	0.4261	0.4053	0.1134
			[1]	n/a	n/a	n/a	n/a
	Max-Min	*Prod*	[0]	0.4412	0.3368	0.3972	0.1036
		$w = 1$	[.5]	0.6341	0.8068	0.4510	0.1102
			[1]	0.6862	0.9155	0.4731	0.1143
Probabilistic Descent	*Max-Prod*	*Min*	[0]	n/a	n/a	n/a	n/a
		$\varepsilon = 0.05$	[.5]	0.4555	0.3874	0.4180	0.1230
		$k = 0.01$	[1]	n/a	n/a	n/a	n/a
	Max-Min	*Prod*	[0]	0.4332	0.3272	0.3908	0.1033
		$\varepsilon = 0.15$	[.5]	0.5722	0.6823	0.4202	0.1071
		$k = 0.01$	[1]	0.5850	0.7466	0.4257	0.1086

Table 8.2: Comparison of Identification Algorithms with Optimization for Cross Identification

Method	Rank/(J_2)
Probabilistic Descent	1 (0.3410)
Neural Learning	2 (0.3530)
Quasi-Newton	3 (0.3630)

Table 8.3: Optimization Identification Algorithm Ranking Based on J_2 Criterion

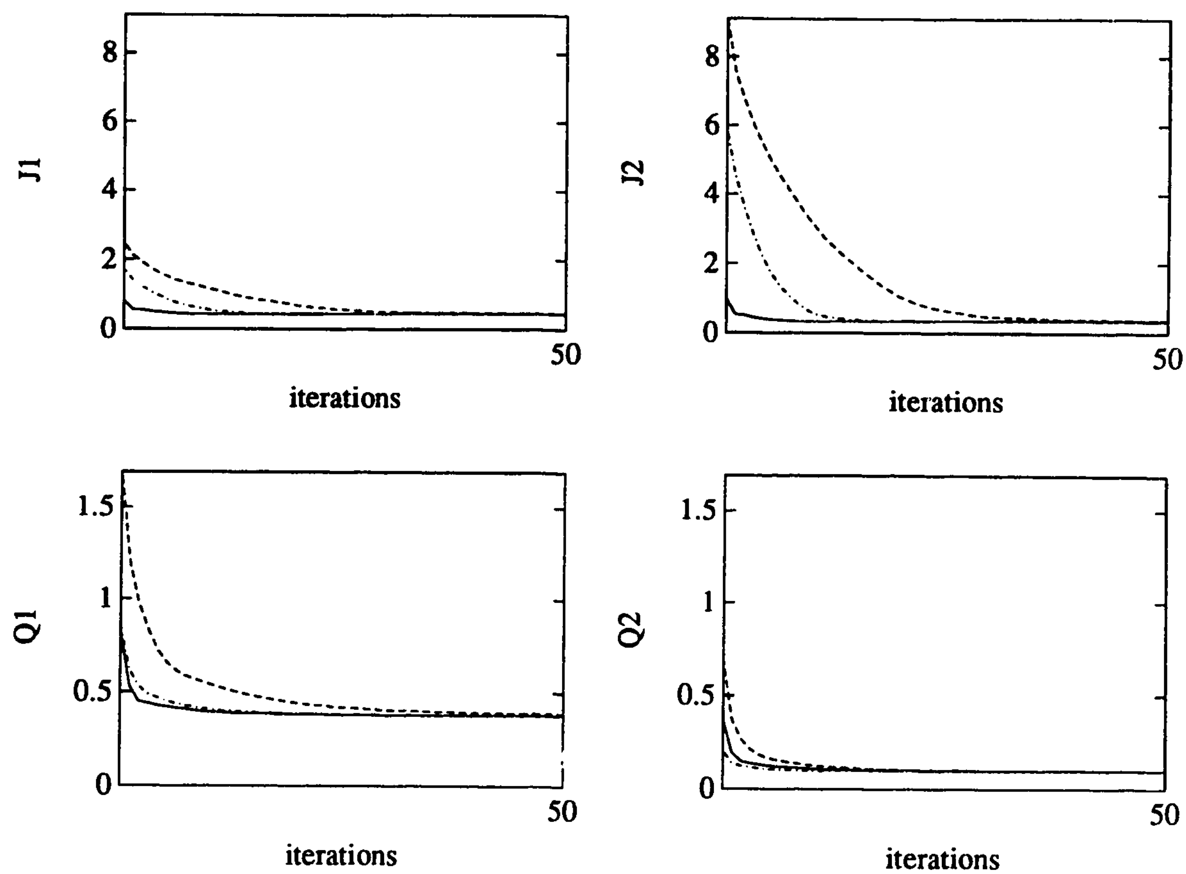


Figure 8.3: Trajectory of Neural Learning Indices for *Max-product*
R **= [0] —— ;** ***R*** **= [.5] – · – · – ;** ***R*** **= [1] – – – –**

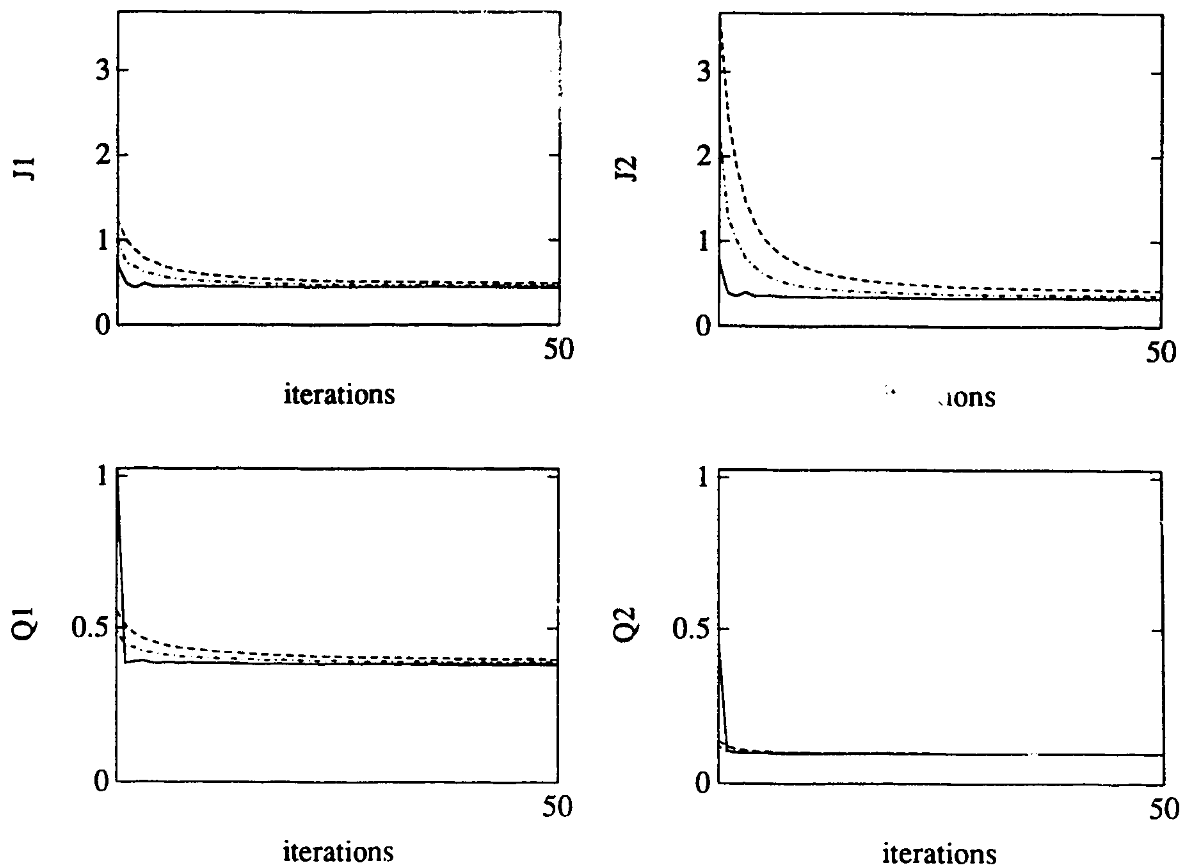


Figure 8.4: Trajectory of Quasi-Newton Indices for *Max-product*
R **= [0] ——— ;** ***R*** **= [.5] – · – · – ;** ***R*** **= [1] – – – –**

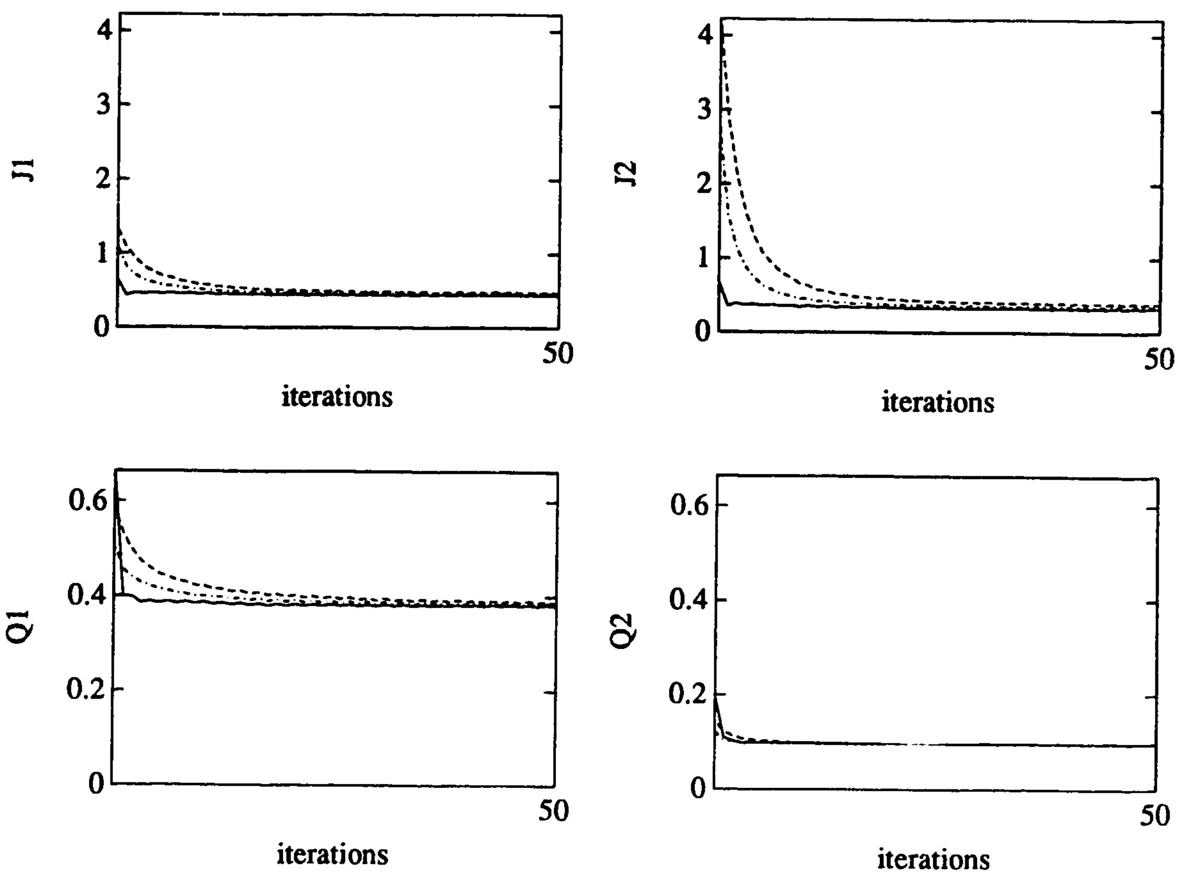


Figure 8.5: Trajectory of Probabilistic Descent Indices for *Max-product*

R **= [0] ——— ; *R* = [.5] – · – · – ; *R* = [1] – – – –**

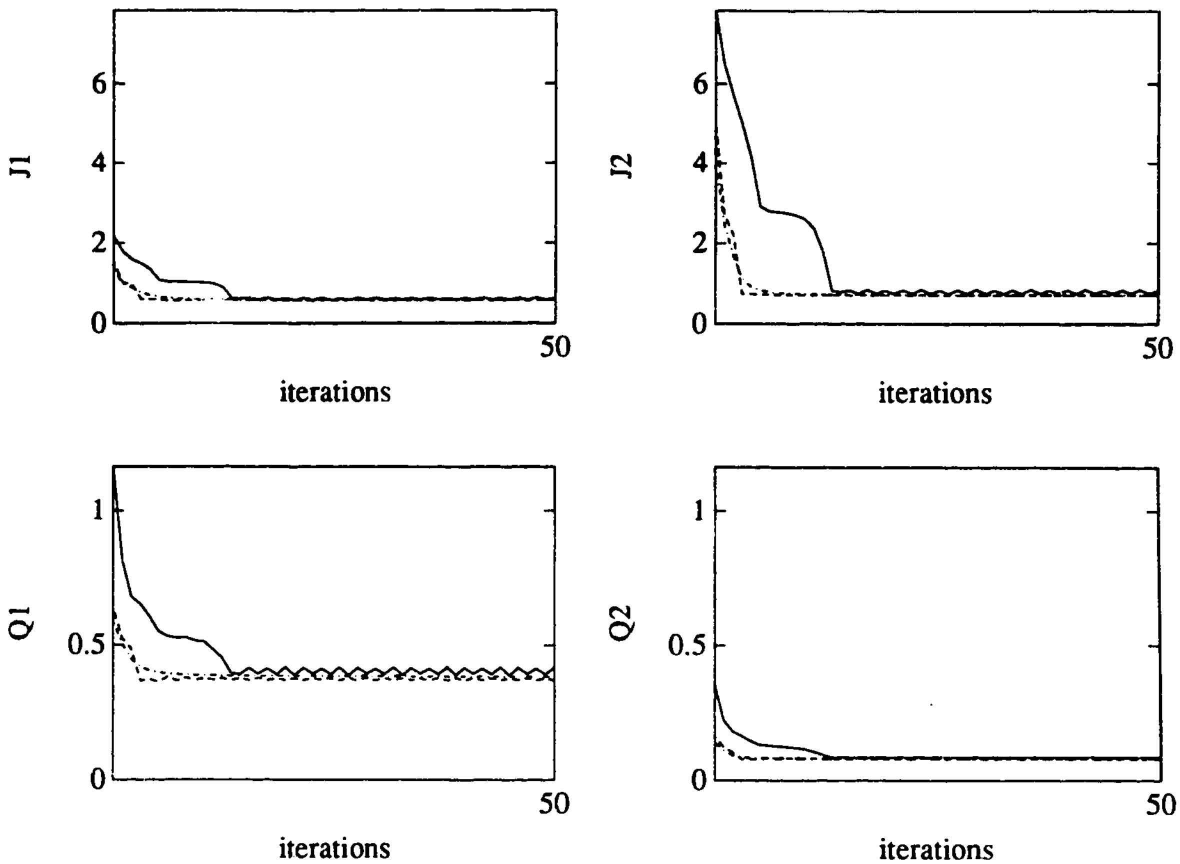


Figure 8.6: Trajectory of Indices for *Max-min* for all Methods for *R* = [.5]
Neural Learning ——; Quasi-Newton – · – · –; Probabilistic Descent – – – –

8.6 Comparison of Causal Minimization Results

The applicability of the optimization algorithms for a causal or control situation is considered in this section. Testing is undertaken using the probabilistic descent method [Ikoma *et al.* 1993], since the identification results using this algorithm were the *best*. The following overdamped, second order, unit gain process was used:

$$y_k = a_1\, y_{k-1} + a_2\, y_{k-2} + b_1\, u_{k-1} + b_2\, u_{k-2} \tag{8.98}$$

where $a_1 = 0.8519$

$a_2 = -0.0970$

$b_1 = 0.1672$

$b_2 = 0.0779$

Learning of the relational matrix, $\boldsymbol{R}$, is via the probabilistic descent method [Ikoma *et al.* 1993] for the first order model discussed in Chapter 7. The one step ahead (OSA) prediction of the output, without state estimation of the past output (i.e. using actual output values) is given in Figure 8.7 to illustrate the ability of the identified relational matrix.

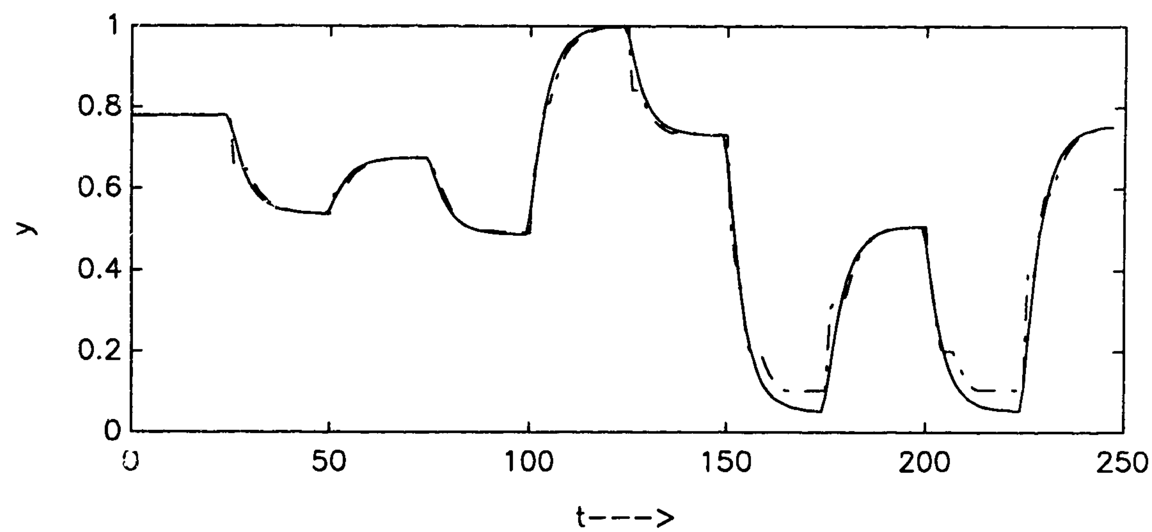


Figure 8.7: Plot of One Step Ahead Prediction of y using actual u
$J_1 = 0.0164$; $J_2 = 0.0008$;
(——— Actual; – · – – · – Model)

Figure 8.8 shows the performance of the algorithm when used to determine an optimized input in a control situation. The calculated process output is based on the input optimized by the current output setpoint and past output setpoint. The resulting process output, based on the optimized calculated input, $\boldsymbol{u}$, versus setpoint show extremely *poor* tracking using the optimization algorithm for the input calculation. So not only is the determination of the input for the control situation *poor* using the optimization algorithm but the algorithm requires extensive iteration and convergence tuning. Therefore using an optimization algorithm to perform the control calculation does not yield *good* results. This is in contrast to discrete, model-based controller formulations where optimization has been used with great success.

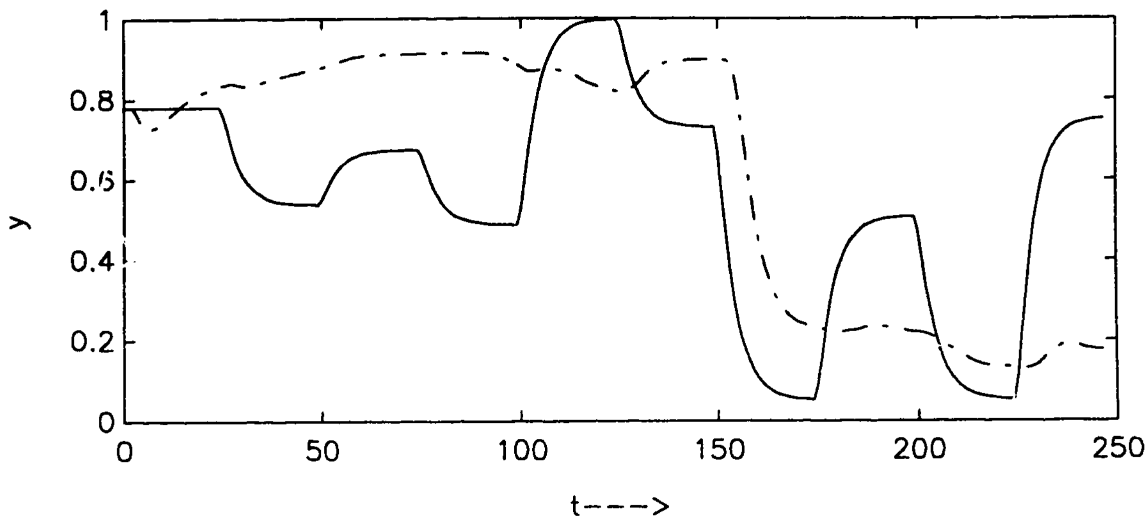


Figure 8.8: Plot of One Step Ahead Prediction of y using optimized u
$J_1 = 0.299$; $J_2 = 0.0760$;
(——— Actual; – · – · – · – Model)

8.7 Correlation between Q_q and J_q

For process systems that are inherently fuzzy, a performance objective based on Q_q is adequate, as its minimization should ensure a fuzzy minimum. However, if the actual output data is not inherently fuzzy, and a fuzzy structure is imposed mainly due to lack of an adequate deterministic model, the predicted output data must be *defuzzified* and then compared against the actual at the *non fuzzy* or numerical level. Thus a *minimization* criteria based the *defuzzified* data should be considered as follows:

$$J_q = |y - \tilde{y}|^q \tag{8.98}$$

where q is a integer.

For these *non-fuzzy* or discrete problems, *minimization* by a Q criteria **does not** guarantee minimization of a J criteria. The poor correlation of these indexes is shown in Figure 8.9, with the graphing of Q_q versus J_q for a representative number of simulations performed during the compilation of Chapter 7 and 8.

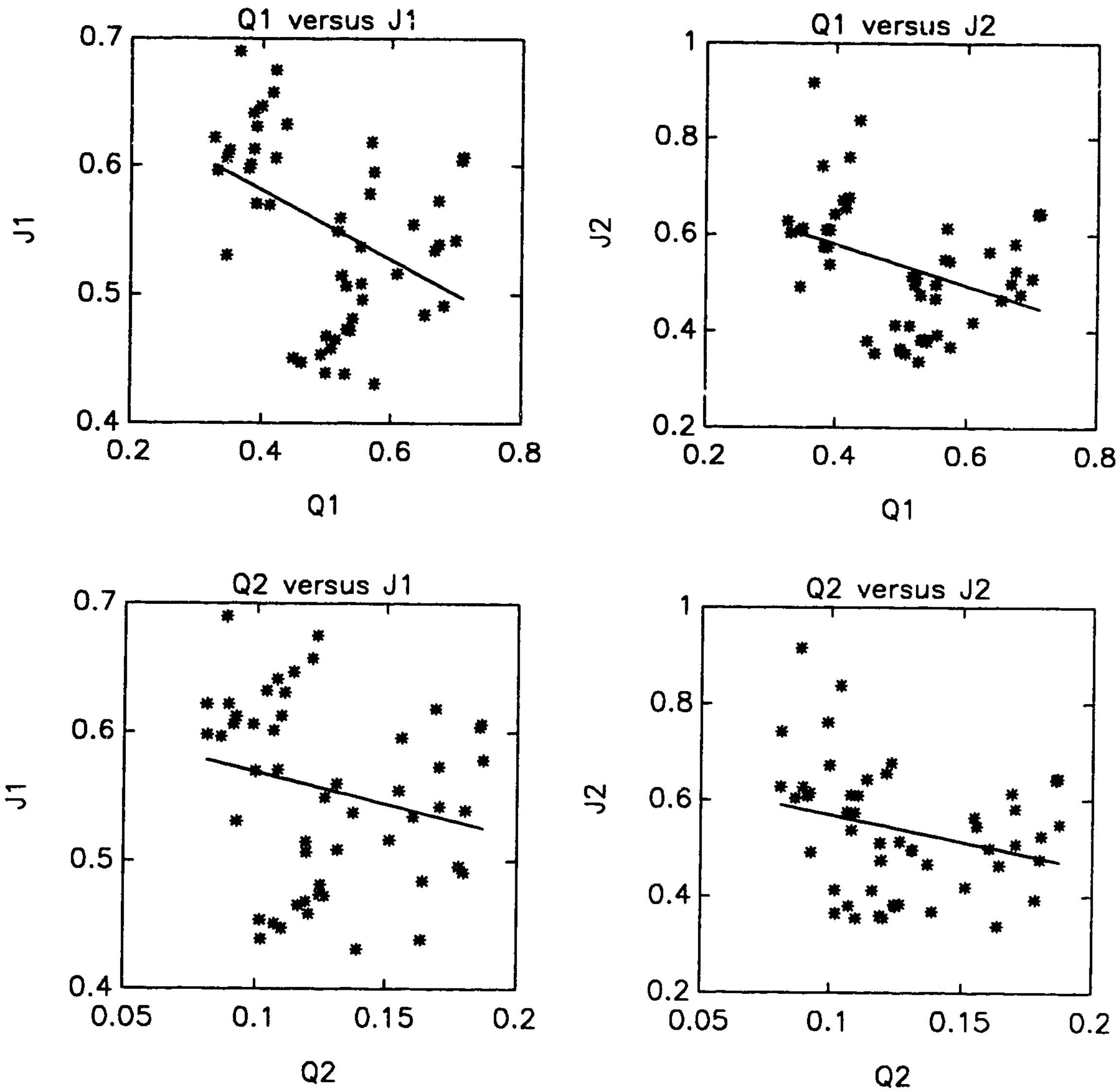


Figure 8.9: Comparison of Q_q versus J_q

Figure 8.9 clearly shows that the indexes of Q_q versus J_q tend toward an inverse relationship. Thus the objective function *minimization* of Q_q will not necessarily result in *minimum* J_q. The results of these graphs reinforce the need for a minimization criteria, as presented by Xu *et al.* [1987], where the *discrete* error is involved in the minimization.

8.8 Summary

The literature search for Chapter 8 provides a brief review of fuzzy analytical derivative theory which confirms the existence of a differential for a *fuzzy relational matrix* and validates this theory for application in the fuzzy domain. Application analysis of this theory is applied to the minimization problem:

$$Q_q = \sum_{j=1}^{n} (\tilde{y}_j - y_j)^q \tag{8.99}$$

where $q = 1 \text{ or } 2$

Results of this analysis confirms that under the conditions of the existence of an exact solution the derivative reduces to the inverse theory of Sanchez [1976, 1977]

The minimization of the control objective function or criterion is considered from an identification perspective and a causal perspective. For the work with identification it was necessary to extend the necessary and sufficient conditions from a solution of the single input-single output identification problem [Pedrycz, 1988 (*max-min*); Pedrycz, 1991(c) (*max-t-norm*)] to the single input-single output fuzzy cause problem for both *max-min* and *max-product*. These results, particularly for the *max-product* composition, demonstrated the need for numerical solution techniques.

Several optimization algorithms have been reviewed [Pedrycz, 1993; Ikoma *et al.*, 1993; Valente de Oliveira, 1994]. The algorithm for neural learning [Pedrycz, 1991(b); Valente de Oliveira, 1993] was extended to the fuzzy cause problem. The probabilistic descent algorithm [Ikoma *et al.*, 1993] was extended to both the fuzzy cause problem and the *max-product* domain. The suitability of these optimization algorithm for identification were compared using the same fuzzification and defuzzification methods, the same reference fuzzy sets and the Box-Jenkins gas furnace data [1970], as the identification algorithms compared in Chapter 7 [Bourke *et al.*, 1995]. The identification results showed that some non-optimized algorithms of Chapter 7 are capable of providing similar results with fewer calculations and less tuning.

A key result from a control system point of view is that the minimization of the fuzzy criteria of either Q_1 or Q_2 does not imply a minimum J, which is defined by a discrete criteria. This knowledge is particularly important for those discrete control systems that are handled in the fuzzy domain due to lack of an adequate deterministic model.

Although optimization algorithms are used effectively in many *discrete* model based control formulations, they are *not* recommended for fuzzy control formulation because the resulting controller output does not give *good* control performance.

REFERENCES

[1] Baboshin, N.P., Naryshkin, D.G., "On Identification of Multidimensional Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 35, pp. 325-331, 1990.

[2] Banks, H.T., Jacobs, M.Q., "A Differential Calculus for Multifunctions", *Journal of Mathematical Analysis and Applications*, Vol. 29, pp. 246-272, 1970.

[3] Bourke, M.M., Fisher, D.G., "A Comparative Analysis of Identification Algorithms for Fuzzy Relational Matrices", *Fuzzy Sets and Systems*, submitted 1995.

[4] Box, G.E.P., Jenkins, G.M., *Time Series Analysis, Forecasting and Control*, Holden Day, San Francisco, 1970.

[5] Buckley, J.J., Qu, Y., "Fuzzy Complex Analysis I: Differentiation", *Fuzzy Sets and Systems*, Vol. 41, pp. 269-284, 1991.

[6] Dubois, D., Prade, H., *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, Toronto, 1980.

[7] Dubois, D., Prade, H., "Towards Fuzzy Differential Calculus Part 1: Integration of Fuzzy Mappings", *Fuzzy Sets and Systems*, Vol. 8, pp. 1-17, 1982(a).

[8] Dubois, D., Prade, H., "Towards Fuzzy Differential Calculus Part 2: Integration on Fuzzy Intervals", *Fuzzy Sets and Systems*, Vol. 8, pp. 105-116, 1982(b).

[9] Dubois, D., Prade, H., "Towards Fuzzy Deferential Calculus Part 3: Differentiation", *Fuzzy Sets and Systems*, Vol. 8, pp. 225-233, 1982(c).

[10] Dubois, D., Prade, H., "On Several Definitions of the Differential of A Fuzzy Mapping", *Fuzzy Sets and Systems*, Vol. 24, pp. 117-120, 1987.

[11] Goetschel Jr., R., Voxman, W., "Elementary Fuzzy Calculus", *Fuzzy Sets and Systems*, Vol. 18, pp. 31-43, 1986.

[12] Hemes, H., "Calculus of Sets Valued Functions and Control", *Journal of Mathematics and Mechanics*, Vol. 18, pp. 47-59, 1968.

[13] Ikoma, N., Pedrycz, W., Hirota, K., "Estimation of Fuzzy Relational Matrix by using Probabilistic Descent Method", *Fuzzy Sets and Systems*, Vol. 57, pp. 335-349, 1993.

[14] Kaleva, O., "Fuzzy Differential Equations", *Fuzzy Sets and Systems*, Vol. 24, pp. 301-317, 1987.

[15] Kandel, A., *Fuzzy Mathematical Techniques with Applications*, Addison-Wesley Publishing Co. Inc., Don Mills, 1986.

[16] Kandel, A., Byatt, W.J., "Fuzzy Differential Equations", *Proceeding of the International Conference on Cybernetics and Society*, Tokyo, pp. 1213-1216, Nov. 1978.

[17] Kaufmann, A., *Introduction to the Theory of Fuzzy Subsets - Vol. 1*, Academic Press Inc., New York, 1975.

[18] Kaufmann, A, Gupta, M.M., *Introduction to Fuzzy Arithmetic*, Van Nostrand Reinhold Company, Inc., New York, 1985.

[19] Ouyang, H., "Topological Properties of the Spaces of Regular Fuzzy Sets", *Journal of Mathematical Analysis and Applications*, Vol. 129, pp. 346-361, 1988.

[20] Ouyang, H., Wu, Y., "On Fuzzy Differential Equations", *Fuzzy Sets and Systems*, Vol. 32, pp. 321-325, 1989.

[21] Pedrycz, W., "Numerical and Applicational Aspects of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 11, pp. 1-18, 1983.

[22] Pedrycz, W., "Approximate Solutions of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 28, pp. 183-202, 1988.

[23] Pedrycz, W., "Fuzzy Modeling: Fundamental, Construction and Evaluation", *Fuzzy Sets and Systems*, Vol. 41, pp. 1-15, 1991(a).

[24] Pedrycz, W., "Neurocomputations in Relational Systems", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, pp. 289-297, 1991(b).

[25] Pedrycz, W., "Processing in Relational Structures: Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 40, pp. 77-106, 1991(c).

[26] Puri, M.L., Ralescu, D.A., "Differentials of Fuzzy Functions", *Journal of Mathematical Analysis and Applications*, Vol. 91, pp. 552-558, 1983.

[27] Sanchez, E., "Resolution of Composite Fuzzy Relation Equations", *Information and Control*, Vol. 30, pp. 38-48, 1976.

[28] Sanchez, E., "Solutions in Composite Fuzzy Relation Equations: Application to Medical Diagnosis in Brouwerian Logic", In: *Fuzzy Automata and Decision Processes*, Gupta, M.M., Saridis, G.N., (Editors), Elsevier/North-Holland, Amsterdam, pp. 221-234, 1977.

[29] Tong, R.M., "Some Properties of Fuzzy Feedback Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 10, pp. 327-330, 1980.

[30] Wang, H.-F., "Numerical Analysis on Fuzzy Relation Equations with Various Operators", *Fuzzy Sets and Systems*, Vol. 53, pp. 155-166, 1993.

[31] Valente de Oliveira, J., "Neuron Inspired Learning Rules for Fuzzy Relational Structures", *Fuzzy Sets and Systems*, Vol. 57, pp. 41-53, 1993.

[32] Zadeh, L.A., "The Concept of a Linguistic Variable", *Information Sciences*, I: Vol. 8, pp. 199-249, II: Vol. 8, pp. 301-357, III: Vol. 9, pp. 43-80, 1975.

[33] Zimmermann, H.-J., *Fuzzy Sets Theory - and Its Applications*, Kluwer-Nijhoff Publishing, Boston, 1985.

CHAPTER 9 DESIGN OF SELF-LEARNING PREDICTIVE FUZZY CONTROLLER

The most effective way to improve the performance of a fuzzy controller is to optimize the fuzzy control rules.

[Tong, 1976]

9.1 Introduction

The equivalent of optimizing the control rules in self-learning relational-based fuzzy control systems translates to updating the relational matrix. This optimization or identification was discussed thoroughly in Chapters 7 and 8. Chapter 9 deals with the design of a predictive fuzzy logic controller and the incorporation of the previously discussed identification into this design so that the resulting controller is *self-learning*, as well as *predictive*.

The quality of a *fuzzy* prediction is a function of the mapping performed by the *fuzzy* model. In many cases the accuracy of the fuzzy model is limited due to imprecision of the input/output data and the vagueness of the model structure. As the length of the prediction horizon increases, so also does the degree of uncertainty of the results. Therefore, the longer the prediction horizon, the more approximate is the information obtained, until a point is reached where predictions from the model may be totally irrelevant [Pedrycz, 1993].

Graham *et al.* present the results of a fuzzy relational-based [1988] and a rule-based [1989] *one step ahead predictive fuzzy logic controller.* The relational-based controller [1988] is tested on a variable area liquid level rig and the paper compares favourably their adaptive algorithm against conventional PI and PI/feedforward controllers. The height of the water in the tank is regulated using 4 inputs, the past two height deviations and the past two differences between inlet and outlet flow rates. Since the relational model to be developed from this scenario proved to be too large (i.e. $21\times21\times11\times11\times21 = 1\,120\,581$) the authors combined a deterministic model of height deviations with a fuzzy model of flow differences, reducing the relational matrix to 2541 elements.

In the subsequent paper [Graham *et al.*, 1989] an adaptive rule-based fuzzy logic controller is applied to a first order process with varying gain and time constant. The process model developed by Graham *et al.* [1989] for this application uses a rule-based look-up table to predict the next change in error given the current change in error and the current process input. The aim of the controller was to reduce the error as much as possible over the next sampling period, so the control action that gives the smallest predicted error was chosen as the current change in control.

The fuzzy *PI* controller design by Song *et al.* [1993] is able to handle a system with a dead time of 2 sampling intervals. *Fuzzy models* are not strongly predictive due to the limited dynamics contained in the relational matrix, a result of the averaging effect of the identification methodologies. However, this does not mean of course that prediction using fuzzy models is impossible, as will be shown in this chapter.

9.2 *Fuzzy* Process Structures

One of the advantages of relational-based fuzzy logic is that the identification algorithms can learn directly from input/output data. Another advantage is predictive control, either 1-step or multi-step ahead, can be implemented in a manner similar to classical predictive controllers. This chapter will demonstrate the parallels between *classical* predictive control and *fuzzy* predictive control.

Starting with the basics, a process of interest can be represented by a standard input/output discrete model as illustrated in Figure 9.1. That is, a discrete (non-fuzzy) input, **u**, is fed to a deterministic process and a discrete (non-fuzzy) output, **y**, results. It should be noted here that the notation throughout this chapter have fuzzy variables (i.e. ***u***, ***y***) in *italics* and discrete (non-fuzzy) variables (i.e. **u**, **y**) are not italicized.

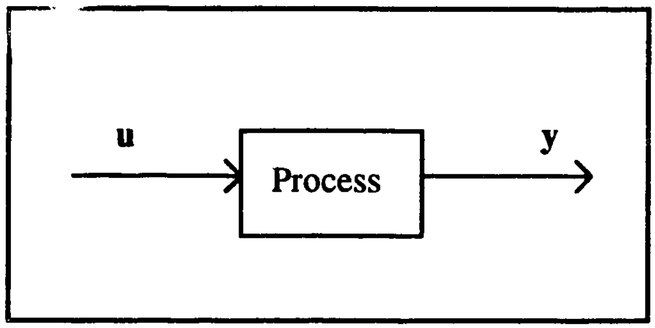


Figure 9.1: Standard Process Model
(u = input, y = output)

Fuzzy processes can be illustrated by a similar schematic which can also be used to describes the three basic problem scenarios for which *fuzzy logic* is ideally suited. These problem scenarios are:

(1) *Fuzzy* Process Dynamics (*fuzzy* model)

(2) *Fuzzy* Process Input (*fuzzy* input)

(3) *Fuzzy* Process Measurement (*fuzzy* output)

(1) *Fuzzy* Process Dynamics

The *fuzzy* dynamic system model is the one most often encountered in the literature. This is the case of discrete input and output data, but the process is considered too complicated to be modeled by conventional techniques. Thus interface techniques of *fuzzification* and *defuzzification* are introduced in order to model the process from a *fuzzy* perspective. The system model for this scenario is illustrated in Figure 9.2, with the hatched box marking the *fuzzy* boundaries.

In most instances in the literature, clearly deterministic processes are artificially fuzzified to demonstrate the capabilities of the various *fuzzy* controllers. This artificial application of fuzzy logic leads to some confusion as to the need for the fuzzy technology. However, there are several application areas where the process dynamics are inherently fuzzy in the sense that the process input, output and/or the model are uncertain (e.g. tar sands extraction, bio-treatment and mineral processing). These example processes are all being controlled today using conventional technologies with discrete data. However, fuzzy logic can compliment these areas thereby permitting the use of vague or indexed data which conventional control systems can not include. When discrete and fuzzy data are to be considered together, it is customary to fuzzify the discrete data so that all information is evaluated on the same basis.

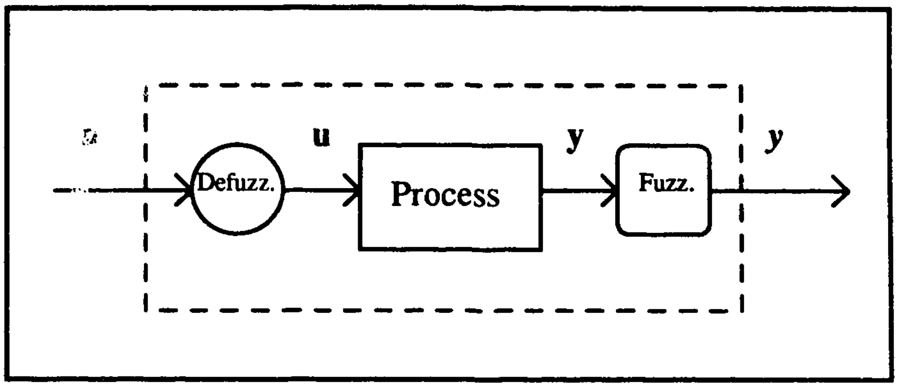


Figure 9.2: *Fuzzy* Process Dynamics Model
(u = discrete input, y = discrete output)
(*u* = *fuzzy* input, *y* = *fuzzy* output)

(2) *Fuzzy* Process Input

The *fuzzy* process input scenario is encountered when the process is inherently fuzzy due to the control law. In these situations the output measurement is discrete and accurate, and it is possible to set the discrete process input to any desired value. However, the problem is that the control law or relationship between the input and output of the controller is only approximate (e.g. whether the error is 1.234 or 1.888 the control law may still call for only a *small* change in input). Although the error is known precisely there is not enough information available to determine a precise value for the manipulated variable, u. An industrial example would be mineral processing using flotation. The current error (or setpoint change) may be known exactly but it is not possible to calculate precise values for the flotation chemicals required to drive the error to zero. Another example is where the manipulated variable can be accurately set to any discrete value but the key component (i.e. composition) of the manipulated variable changes with time and is currently unknown. Thus the manipulated variable can be set accurately to a discrete value but the effect of the process output is *approximate* or *fuzzy*. This scenario is illustrated in Figure 9.3.

(3) *Fuzzy* Process Measurement

The final *fuzzy* process scenario is *fuzzy* process measurements. In this situation the process measurement is inherently fuzzy (i.e. qualitative or indexed). The manipulated input into the process is discrete, but the controller is fuzzy because of the *fuzziness* inherent in the process measurement. An industrial example would be in tar sands processing where the clay content or oil content can only be estimated approximately on-line. This process model is illustrated in Figure 9.4.

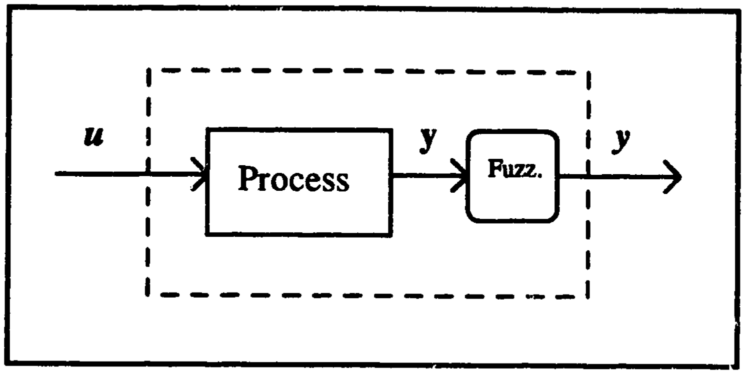


Figure 9.3: *Fuzzy* Process Input Model
(y = discrete output; *y* = *fuzzy* output)
(*u* = *fuzzy* input,)

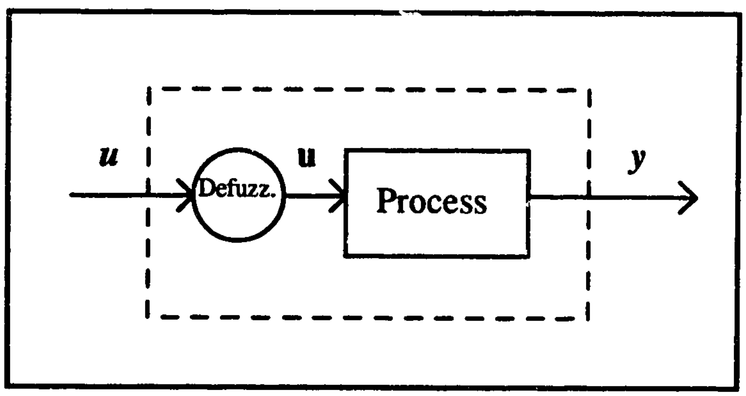


Figure 9.4: *Fuzzy* Process Measurement Model
(u = discrete input)
(*u* = *fuzzy* input, *y* = *fuzzy* output)

The three process situations illustrated by Figures 9.2, 9.3 and 9.4 can be combined to produce other process situations, such as a totally *fuzzy* process where the dynamics, inputs and measurements are all *fuzzy*. However, the modeling and control would be executed the same as the other problem scenarios, without the need to artificially fuzzify or defuzzify.

Although the *fuzzification* and *defuzzification* may seem artificial in some instances, the power or benefit of the technique is that it permits the handling of both fuzzy and non-fuzzy data in the same model.

Since it is clear that the fuzzy model can handle both discrete and fuzzy data, via fuzzification and defuzzification interfaces, these interfaces will be assumed to be an implicit part of the modeling and control structure. Therefore these procedures will not be shown explicitly in the block diagrams during the following controller development. This will simplify the controller diagrams that follow permitting a one-to-one comparison between the conventional predictive control structure and the fuzzy logic predictive control structure being developed in this thesis. The fuzzification and defuzzification interfaces will be included, however, in the final controller design schematic (Figure 9.18).

9.3 *Fuzzy* Control vs. Classical Predictive Control

This section will show that the development of *fuzzy* model-based control parallels the development of conventional model-based control. The starting point is simple servo and regulatory control leading to the Smith Predictor model, closed-loop stability results and finally to model-based predictive control.

9.3.1 Simple Servo Control

The *fuzzy* controller development starts with the basic control problem of the discrete system. The first structure considered is the simple *servo control* system, illustrated in Figure 9.5.

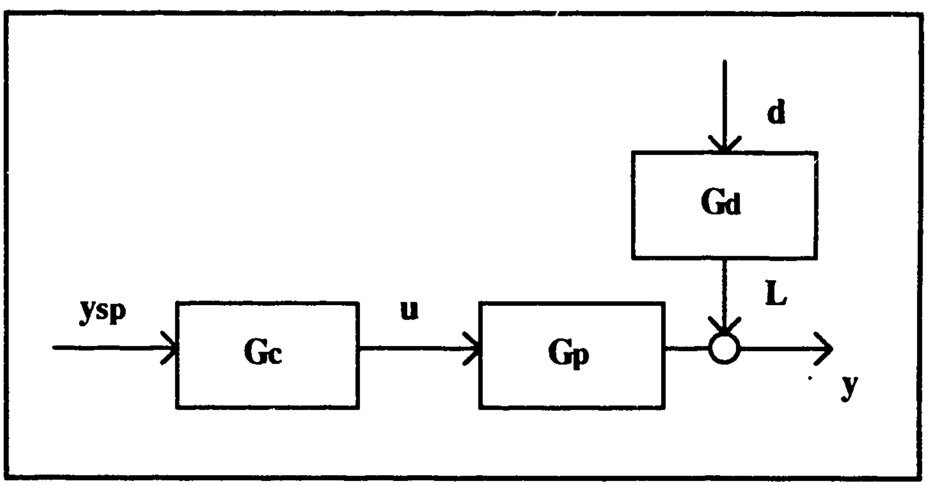


Figure 9.5: Servo Control

If $\mathbf{G_p}$ is known exactly, the *best model-based* controller is simply the inverse of the process model:

$$\mathbf{G_c} = \mathbf{G_p}^{-1} \tag{9.1}$$

and perfect control is obtained in the sense that $y = y_{sp}$ for all time. However, $\mathbf{G_p}$ does not always exist or if it does the direct inverse in not always practical.

In terms of a fuzzy model a similar analogy can be made.

Process Model: $$y = u \circ G_p \tag{9.2}$$

Control Law: $$u = y_{sp} \circ G_c \tag{9.3}$$

For perfect control,

$$y = y_{sp}. \tag{9.4}$$

Combining equations (9.2), (9.3) and (9.4).

$$y_{sp} = y_{sp} \circ G_c \circ G_p \tag{9.5}$$

Clearly equation (9.5) represents the eigen fuzzy problem discussed in Chapter 6.

Let

$$E = G_c \circ G_p \tag{9.6}$$

then

$$G_c = E \, \textcircled{\psi} \, G_p \tag{9.7}$$

where $\textcircled{\psi}$ is the inverse operator defined in Chapter 5. As with conventional model-based control, the inverse calculation of equation (9.7) may not always exist.

9.3.2 Feedforward Control

The servo based control system in Figure 9.5 can not handle disturbances. Regulation of the system faced with these challenges must also be considered. So consider the structure of *regulatory feedforward control*, illustrated in Figure 9.6.

For perfect feedforward control, the conventional feedforward control law is:

$$\mathbf{G_{ff}} = -\frac{\mathbf{G_d}}{\mathbf{G_p}} \tag{9.8}$$

Again this design has a fuzzy equivalent. Let $\boldsymbol{d} = \{d_i \mid i = 1, 2, \ldots n\}$. Then

$$\sum_i \{d \circ \mathbf{G_{ff}} \circ \mathbf{G_p} + d \circ \mathbf{G_d}\} = 0 \tag{9.9}$$

$$\sum_i d \circ [\mathbf{G_{ff}} \circ \mathbf{G_p} + \mathbf{G_d}] = 0 \tag{9.10}$$

Therefore $\mathbf{G_{ff}} \circ \mathbf{G_p} + \mathbf{G_d}$ must be such that it either produces a zero sum for all values of $\boldsymbol{d}$ (i.e. $\boldsymbol{d}$ is orthogonal to $\mathbf{G_{ff}} \circ \mathbf{G_p} + \mathbf{G_d}$ is some fuzzy sense) or it must equal zero.

$$\mathbf{G_{ff}} \circ \mathbf{G_p} + \mathbf{G_d} = 0 \tag{9.11}$$

so

$$\mathbf{G_{ff}} = -\mathbf{G_d} \circledS \mathbf{G_p} \tag{9.12}$$

The negative sign (–) in equation (9.12) can be interpreted as increasing the universe of discourse to include the negative or opposite domain, if the universe is absolute, or inverting the results if the domain uses data that is differenced (i.e. $\boldsymbol{u}$ or $\Delta \boldsymbol{u}$, respectively).

9.3.3 Modified Feedforward Control

For conventional feedforward control the problem is formulated in terms of the disturbance, **d**, measured at the input to the process. However, for later comparison with model-based predictive controllers, it is convenient to formulate feedforward control in terms of **L**, (i.e. the effect of the disturbance, **d**, on the process output, **y**). *Regulatory feedforward control* can be represented, as shown in Figure 9.7, even though in many process applications it is not possible to measure **L**.

For perfect feedforward control for the modified case, the conventional feedforward control law is:

$$G_{ff} = -\frac{1}{G_p} \tag{9.13}$$

Note that the feedforward controller is simply the process inverse, as in servo control.

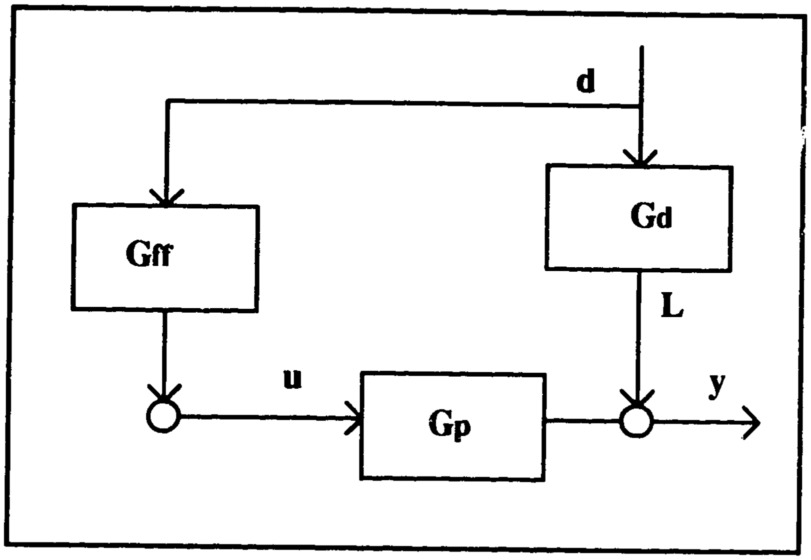


Figure 9.6: Regulatory Feedforward Control

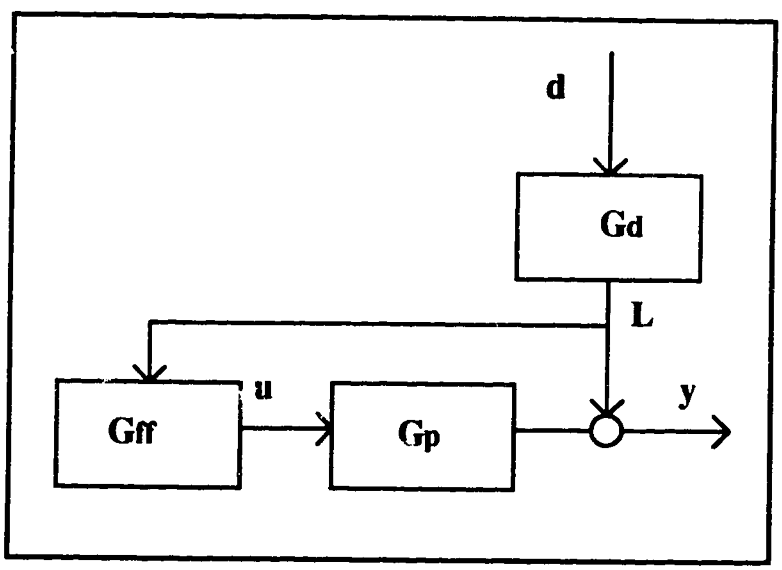


Figure 9.7: Modified Regulatory Feedforward Control

For the fuzzy equivalent, let $L = \{L_i \mid i = 1, 2, \ldots n\}$. Then

$$\sum_i [L \circ G_{ff} \circ G_p + L] = 0 \tag{9.14}$$

$$\sum_i L \circ [G_{ff} \circ G_p + E] = 0 \tag{9.15}$$

where E is the eigen fuzzy relation such that $L = E \circ L$

Again assume that $L \circ [G_{ff} \circ G_p + E]$ does not produce a zero result for all L, then

$$G_{ff} \circ G_p + E = 0 \tag{9.16}$$

and

$$G_{ff} = -E \circledast G_p \tag{9.17}$$

9.3.4 Combined Servo-Regulatory Control

Now both the servo and the modified feedforward regulatory control schemes, from Figure 9.5 and 9.5, can be combined to produce a servo-regulatory control representation. The schematic for this scenario is illustrated in Figure 9.8

For perfect control for the servo-regulatory system, the conventional control law is:

$$G_c = \frac{1}{G_p} \tag{9.18}$$

For the fuzzy equivalent, let $y = \{y_i \mid i = 1, 2, \ldots n\}$ and $L = \{L_i \mid i = 1, 2, \ldots n\}$. Then

$$\sum_i [y - y \circ G_c \circ G_p + L - L \circ G_c \circ G_p] = 0 \tag{9.19}$$

$$\sum_i y \circ [E_y - G_c \circ G_p] + \sum_i L \circ [E_L - G_c \circ G_p] = 0 \tag{9.20}$$

where E_y is the eigen fuzzy relation such that $y = E_y \circ y$

E_L is the eigen fuzzy relation such that $L = E_L \circ L$

So for equation (9.20) to be true

$$G_c = E_y \circledcirc G_p \cap E_y \circledcirc G_p \tag{9.21}$$

where $\cap$ is the intersection.

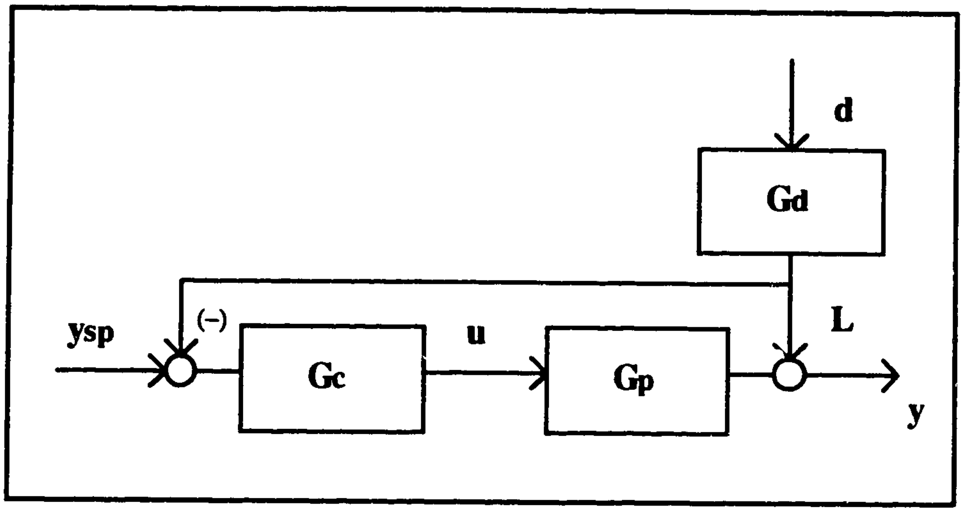


Figure 9.8: Servo–Regulatory Control

9.3.5 Model-Based Control

As has been shown, discrete and *fuzzy* systems can be represented by the same schematic diagrams for servo-regulatory control. Model-based control follows directly from standard feedback control, and is equivalent to servo-regulatory control, as will be shown next during the model-based control development. The following controller development is valid for both discrete and *fuzzy* systems. The validity of this design for *fuzzy* systems will be confirmed by the *fuzzy predictive controller* design outlined later in this chapter.

For the predictive development, start with the standard feedback control model, shown in Figure 9.9. This feedback control strategy can be translated into a model-based control strategy by adding and subtracting a process model, G_m, as shown in Figure 9.10, which is mathematically equivalent to Figure 9.9.

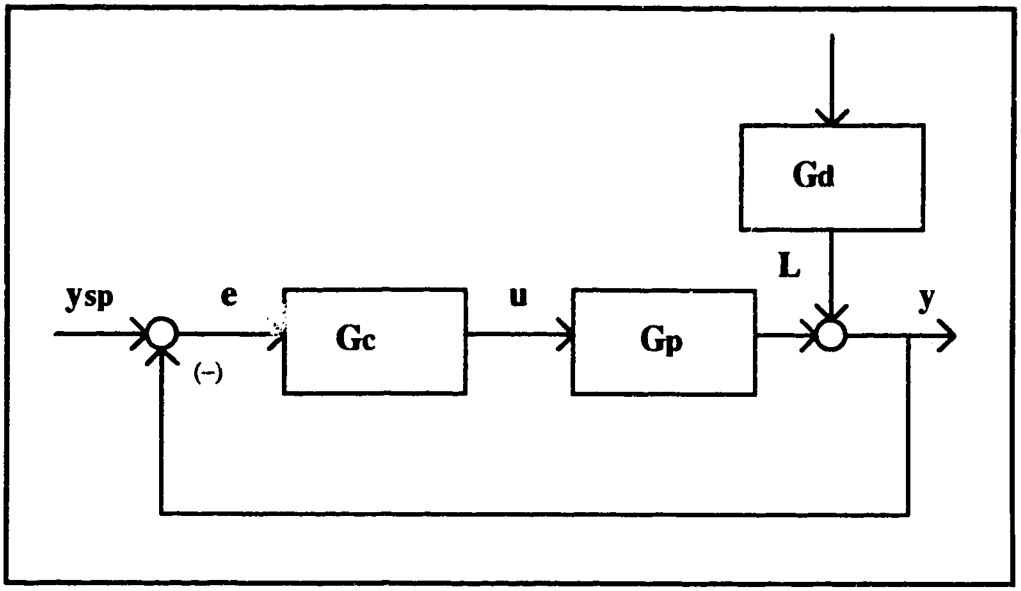


Figure 9.9: Standard Feedback Control

The process model, G_m, is then combined with the controller, G_c, and a new controller, G_c^*, developed. The controller, G_c^*, is now model-based. This new control scenario is illustrated in Figure 9.11. Note that G_c^* is simply the closed-loop transfer function for a standard feedback loop containing G_c and G_m. That is:

$$G_c^* = \frac{G_c}{1 + G_c G_m} \tag{9.22}$$

or

$$G_c = \frac{G_c^*}{1 - G_c^* G_m} \tag{9.23}$$

If $G_m = G_p$ then $\hat{L} = L$ and Figure 9.11 reduces to the servo and regulatory control schematic of Figure 9.8. A simplified schematic of this reduced system is shown in Figure 9.12. Using the same arguments used in conjunction with Figures 9.5 and 9.8, it follows that for perfect control $G_c^* = G_p^{-1}$.

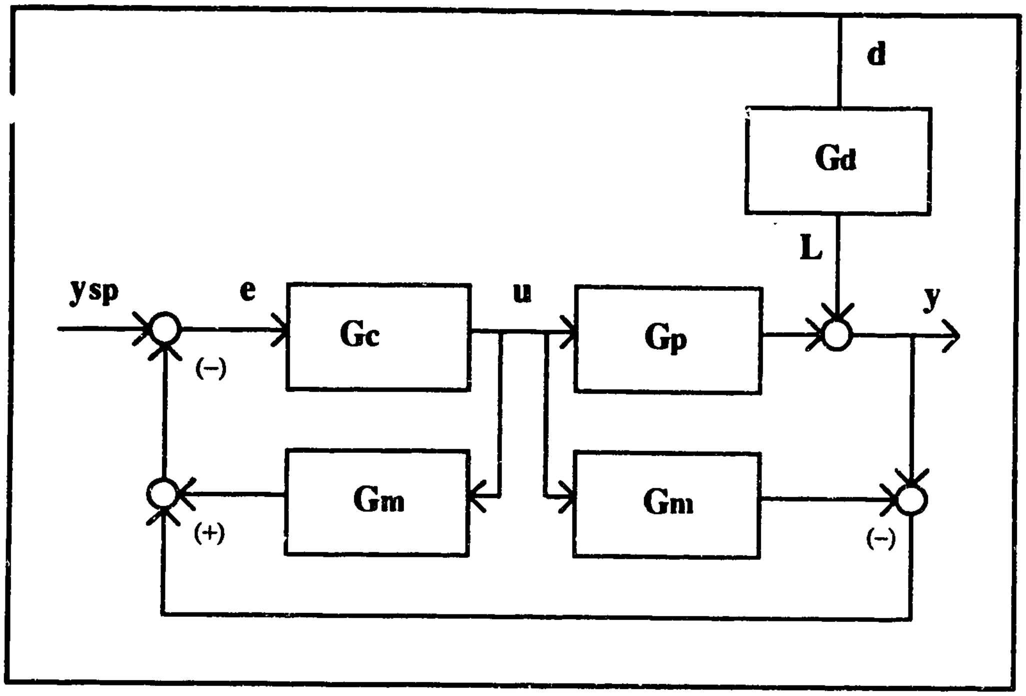


Figure 9.10: Feedback Control +/– Process Model

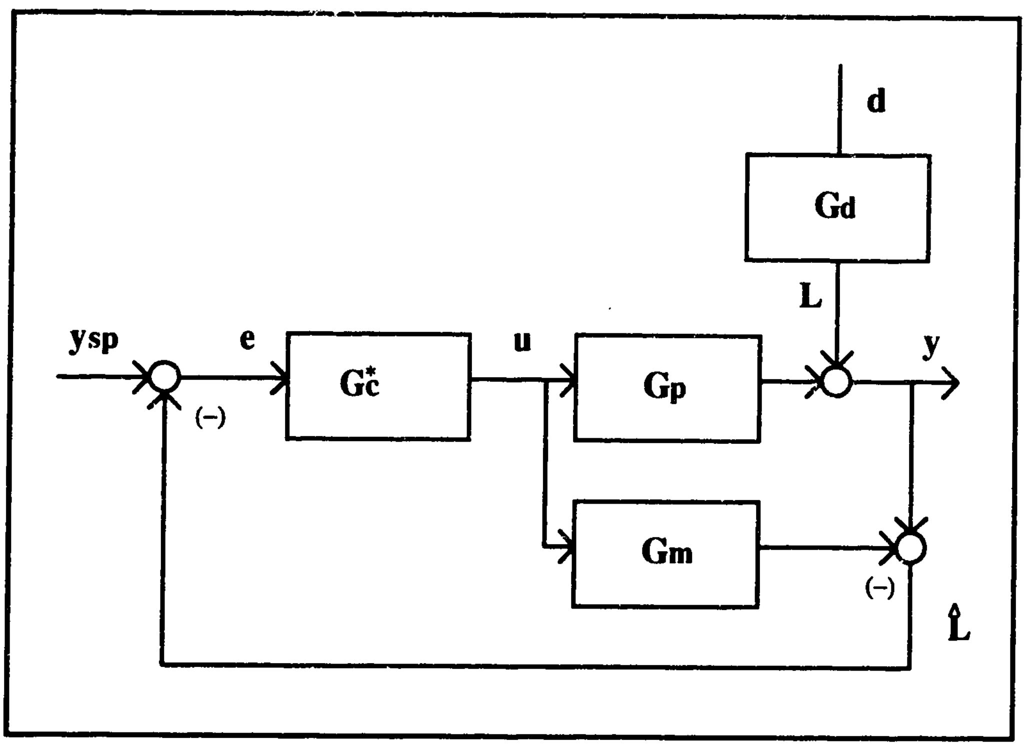


Figure 9.11: Model-Based Feedback Control

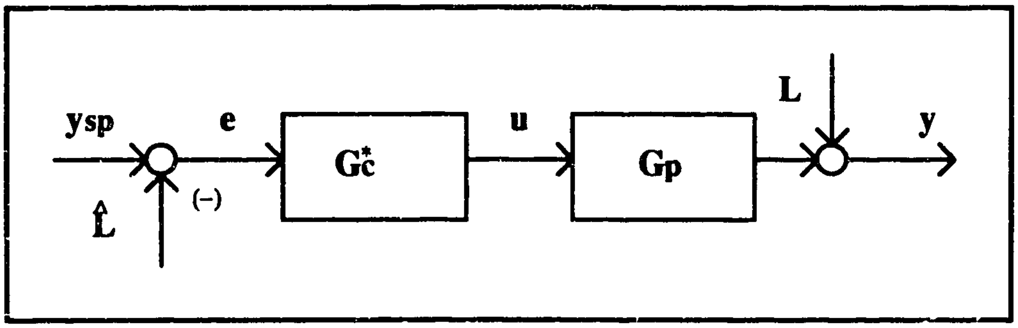


Figure 9.12: Simplification of Servo-Regulator and Model-Based Control

9.3.6 The Advantages of Model-Based Control

There are several advantages to using model-based control, which follow from the preceding developments.

- the basic principles of *open-loop servo* and *feedforward* control led to Figure 9.8 with a model-inverse controller (i.e. $G_c = G_p^{-1}$)
- the standard *feedback* control scheme in Figure 9.9 can be re-arranged to the form shown in Figure 9.11 which for the ideal case of perfect modeling (i.e. $G_m = G_p$) reduces to the simplified form of Figure 9.12.
- Figure 9.8 and 9.12 are equivalent so it can be concluded that the basic principles of conventional servo, feedforward and feedback control are incorporated in Figure 9.12 for the ideal case of perfect modeling.

The simplicity of Figure 9.12 make it easy to derive the following characteristics for this control scheme:

(1) The response of the controlled variable, $y(k)$, to changes in the setpoint, $y_{sp}(k)$ and/or the disturbance/load, $L(k)$, is given (in transfer function notation) by:

$$y = [G_p G_c^*] y_{sp} + [1 - G_p G_c^*] L \qquad (9.24)$$

Note that (9.24) is much easier to derive and interpret than the *closed-loop* relationships that apply to the conventional feedback control scheme of Figure 9.9.

(2) The basis for designing *the best possible controller* is obvious, (i.e. make $G_c^* = G_p^{-1}$). The design procedures for conventional feedback (Figure 9.9) are more complex and the controller for *best possible* control performance is not as obvious. Also note that model-inverse control, $G_c^* = G_p^{-1}$, give perfect control in the sense that:

$$y = y_{sp} \qquad \forall\, y_{sp} \text{ and } \forall\, L \tag{9.25}$$

(3) The system in Figure 9.12 is obviously stable if both G_c^* and G_p are stable. There is no need to use stability analysis procedures typically applied to conventional feedback control systems.

(4) The stability and performance analysis for Figure 9.12 is not complicated if constraints are added to the controller output, $u(k)$. This is important because in practice, $u(k)$ always has finite upper and lower bounds (i.e. a control valve must operate between the limits of 0% and 100%).

The ideal case of perfect modeling has been assumed throughout this discussion. However, in reality this is seldom the case. When Model-Process-Mismatch (MPM) is present (i.e. $G_m \neq G_p$) then the model-based controller of Figure 9.11 can be unstable even if G_c^* and G_p are stable. However, it can be shown that for most systems a filter, f, exists such that replacing G_c^* by $G_c^* f$ in Figure 9.11 will result in stable, but less than perfect control. Because fuzzy models are not as amenable to analytical analysis as transfer functions, it is recommended that *the design of the proposed controller be made on the basis of perfect modeling and that the final tuning of the controller and the analysis of control performance be done by simulation and field trials.*

Although the arguments presented in this chapter are mainly heuristic, it is reasonable to conclude that:

> *the model-based control structure shown in Figures 9.11 and 9.12 is appropriate and desirable for fuzzy, as well as conventional control systems.*

The next step in this development is to show that the model-based control system of Figure 9.11 is *better* than conventional feedback control for processes containing time delays. This will be done in the next subsection by analyzing the Smith predictor control scheme. Extending the model-based control system of Figure 9.11, to include *predictive control systems* will be done following the discussion of Smith predictors.

9.3.7 Time Delay Compensation using a Smith Predictor

Now consider the model-based feedback control illustrated in Figure 9.11, again. Perfect control for setpoint and load disturbances requires *model-inverse control* (i.e. $\mathbf{G_c^*} = \mathbf{G_p^{-1}}$). This is not physically possible if the process contains a time delay since a time delay is not invertable.

The Smith predictor technique, shown in Figure 9.13, handles dead time processes by using two process models, one ***with*** dead time, $\mathbf{G_m}$, and other ***without***, $\mathbf{G_m^*}$. Note that for the ideal case with perfect modeling $\mathbf{r}(k) = \mathbf{L}(k)$ (as in Figure 9.11) and Figure 9.13 reduces to the standard feedback control scheme of Figure 9.9 with the controller, $\mathbf{G_c}$, designed for a process ***without*** delay, $\mathbf{G_m^*}$. The controller output is sent to the actual process, $\mathbf{G_p}$, as well as to the delay-free model, $\mathbf{G_m^*}$. Therefore the actual process output is a delayed version of the output of $\mathbf{G_m^*}$ or conversely the output of $\mathbf{G_m^*}$ is a *prediction* of the actual plant output (assuming $\mathbf{L}$ = constant over the period of prediction).

To see the similarity of the Smith predictor and the model-based controllers, discussed earlier, compare Figure 9.13 versus Figure 9.10. The two figures are the same except that in Figure 9.10 both $\mathbf{G_m}$ transfer functions are assumed to be identical. In model-based control (Figure 9.11) perfect control required that $\mathbf{G_c^*} = \mathbf{G_p^{-1}}$. However, as stated earlier, if the process contains a pure time delay the inverse is not realizable. The simplest approach for model-based control (Figure 9.11) is to neglect the delay and simply invert the process transfer function without the delay (i.e. $\mathbf{G_c^*} = [\mathbf{G_m^*}]^{-1}$). The Smith predictor design is equivalent. This is shown by combining $\mathbf{G_c}$ and $\mathbf{G_m^*}$ in Figure 9.13 into a standard feedback loop as was done when reducing Figure 9.10 to 9.11. The performance of a standard feedback loop improves as the controller gain increases. As the controller gain approaches infinity the control becomes *perfect* in the sense that $y = y_{sp}$. This means that Figure 9.13 reduces to Figure 9.11 with $\mathbf{G_c^*} = [\mathbf{G_m^*}]^{-1}$ and hence the Smith predictor and the model-based controller are identical. The importance of this discussion is to show how model-based controllers compensate for process time delays through *prediction*.

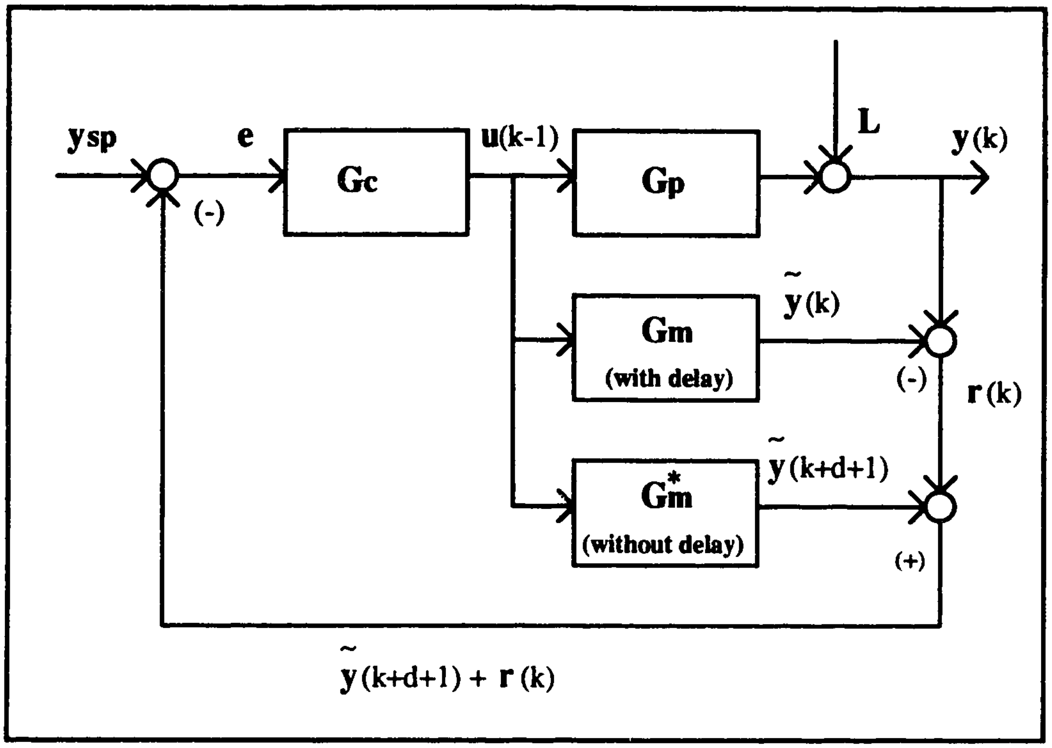


Figure 9.13: Smith Predictor
(~ = estimate; d = time delay; k = k-th sample instant)

9.3.8 Model-Based Predictive Control

Model-based predictive control (MPC) includes a very broad classification of control techniques that have evolved in the control literature over the past 2 or 3 decades. These include Model Predictive Heuristic Control (MPHC) (Richalet *et al.*, 1978), Dynamic Matrix Control (DMC) (Cutler, 1980), Internal Model Control (IMC) (Rivera *et al.*, 1986) and Generalized Predictive control (GPC) (Clarke *et al.*, 1987). However, the basic concepts required for derivation of the fuzzy controller in this thesis can be deduced from the simplified representation shown in Figure 9.14.

The basic structure of Figure 9.14 is equivalent to that of Figure 9.13 (Smith Predictor) and Figure 9.10 (which is equivalent to the conventional feedback of Figure 9.9 and the model-based controller in Figure 9.11) *except that a single block with two outputs is used to represent the model rather that two separate blocks.* (The filter is discussed below) This leads to the correct *functional* interpretation but is obviously not consistent with the conventions for *block diagrams.*

The key difference in predictive control is that the control error $e(k+d+1)$ used by the controller is based on future and/or *predicted* values of the setpoint, $y_{sp}(k+d+1)$, and the process output, $\tilde{y}(k+d+1)$, rather that the *current* values $y_{sp}(k)$ and $\tilde{y}(k)$ used in the figures preceding Figure 9.13. (Note that k represents the current time or current control interval).

A very simplified explanation, of predictive control is as follows:

(1) One step ahead predictive control based on $\mathbf{e}(k+1)$ is required for *discrete* control systems (e.g. those implemented using a digital control computer) because the output, $\mathbf{y}(k)$, of the practical process does not respond *instantly* to changes in the input $\mathbf{u}(k)$. More specifically, the effect of a change in the process input at the k-th control interval is not observed (measured or predicted) until the $(k+1)$-th control interval. Therefore, for perfect control the controller must calculated, $\mathbf{u}(k)$, such that $\tilde{\mathbf{y}}(k+1)$ will equal $\mathbf{y}_{sp}(k+1)$.

(2) Multi-step-ahead prediction is used for systems with time delays since by definition the effect of a change (e.g. step) in the process input does not affect the output until after the delay, as discussed in the previous section on Smith predictors (Figure 9.13). In terms of discrete controllers this means that the *best possible* output control is to calculate $\mathbf{u}(k)$ so that $\tilde{\mathbf{y}}(k+d+1) = \mathbf{y}_{sp}(k+d+1)$.

(3) Multi-step ahead predictive control is also used to give more practical (detuned, sub-optimal) control. Consider a step change in setpoint at time k on a process without delay operating at steady state. The best possible output control would be achieved with *one-step-ahead* optimal or deadbeat control such that $\mathbf{y}(k+1) = \mathbf{y}_{sp}(k+1)$. However, this type of control is often impractical. For example, the required control action, $\mathbf{u}(k)$, could be greater than available (i.e. exceeds the upper limit on the input) or simply larger than desired for operational reasons. Obviously a much smaller control change, $\mathbf{u}(k)$, would be required to achieve a given setpoint change in $(d+1)$ control intervals rather that in a single control interval and the output would change more slowly. The objective of the controller is therefore to calculate $\mathbf{u}(k)$ such that $\tilde{\mathbf{y}}(k+d+1) = \mathbf{y}(k+d+1)$, where d is the tuning parameter.

Note that more advanced controllers can be formulated that make it possible to specify the *path or trajectory* that the output follows as it moves from the current value, $\mathbf{y}(k)$, to the desired value, $\mathbf{y}(k+d+1)$. Similarly, it is possible to formulate *optimal* controllers that calculate a series of control moves $\{\mathbf{u}(k), \mathbf{u}(k+1), \ldots, \mathbf{u}(k+\mathrm{Nu})\}$ such that the process output follows the desired trajectory. Most of these multi-step and/or multivariable *discrete* controllers use an optimization algorithm to calculate the control action, $\mathbf{u}(k)$. However, as noted in the concluding section to Chapter 8, numerical optimization or matrix inversion does not appear to work well in the fuzzy domain. This thesis therefore focuses on the (single-point) d-step ahead predictive controller.

The filter shown in the feedback path is added primarily to achieve the following two objectives:

(1) To convert the scalar residual, $\mathbf{r}(k)$ (which under ideal conditions of perfect modeling is equal to $\mathbf{L}(k)$) into a prediction of the residual at time $(k+d+1)$. In principle this could be done using *single series forecasting* (*time series*) *techniques* based on the known and past values $\{\mathbf{r}(k-i+1), i = 1, 2, 3, \ldots\}$. However, in practice it is common to simply assume that $\mathbf{r}(k+d+1) = \mathbf{r}(k)$.

(2) To prevent the controller from over-reacting to noise in the measured output and/or to the effect of model process mismatch. For this purpose a simple low-pass filter or averaging operation is usually adequate.

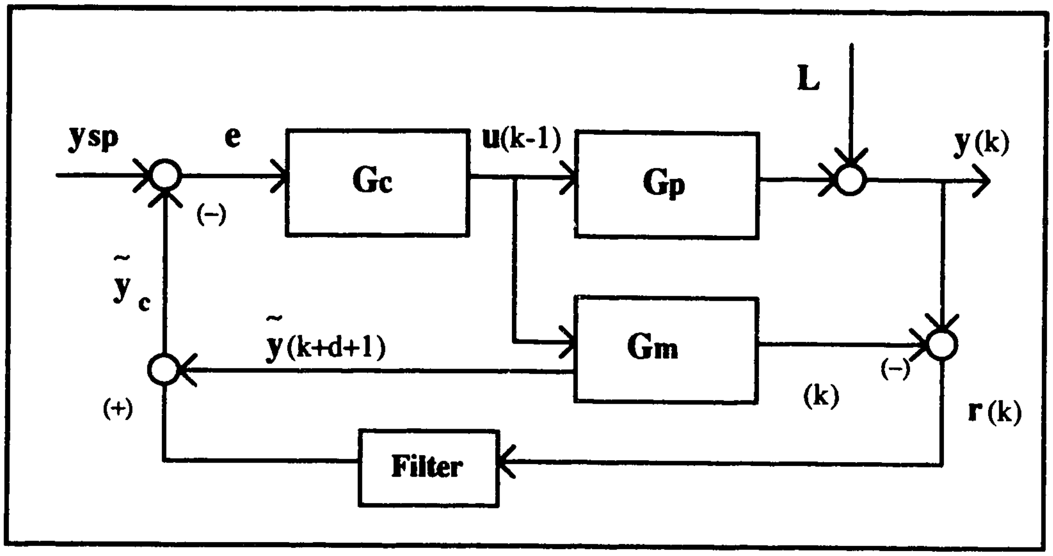


Figure 9.14: Predictive Control at Control Interval *k*
(~ = estimate; d = time delay; k = k-th sample instant)

From the preceding discussion it is obvious that model-based control techniques require a process model for use in the output prediction and/or control calculation steps. For simple, *time invariant* processes it may be sufficient to derive a suitable model *a prior* and include it, for example in the G_m block of Figure 9.14. However, in many practical applications it is necessary to update the model *on-line* due to changes in operating conditions, raw material, product specification, etc. In principle, this is easily accomplished by adding an *identification* block to Figure 9.14. As shown in Figure 9.15, the identification block supplies an updated process model to the prediction and control blocks, This can be done at some regular multiple of the control interval or on an *as required* basis.

Figure 9.15 is the basis for the formulation of the *Fuzzy Predictive Control* algorithm developed as part of this thesis. The next section describes the fuzzy control algorithm and leads to Figure 9.17 which represents the *fuzzy-equivalent* of discrete controllers described by Figure 9.15.

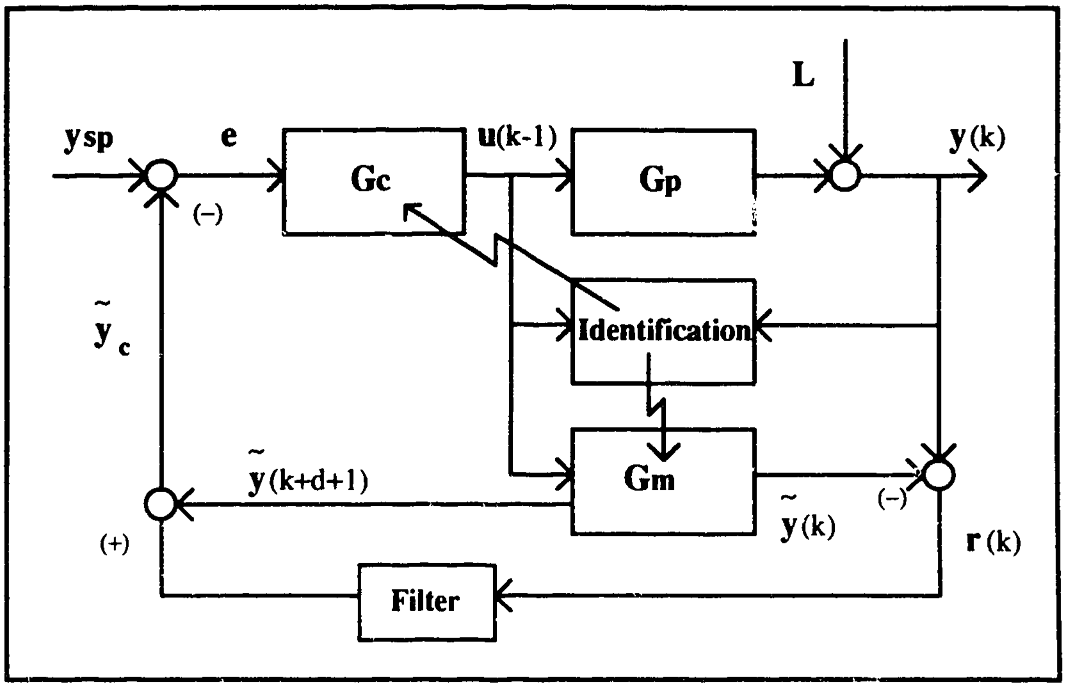


Figure 9.15: Self-Learning Predictive Control
(~ = estimate; d = time delay; k = k-th sample instant)

9.4 *Fuzzy* Predictive Controller Design

9.4.1 Introduction

A brief functional description, in conventional terms, of the *fuzzy controller* that is developed in this section is:

> *An adaptive, SISO, d-step ahead predictive controller based on a first order plus time delay relational fuzzy model.*

The design of the proposed *Self-Learning Predictive Fuzzy Logic Controller* of this thesis parallels that of classical and modern control techniques as discussed in the previous sections. The specific design of the *fuzzy logic* controller developed here assumes that the process variables, both input and output, are discrete and control optimization is performed through the minimization of the *discrete scalar distance*:

$$|y_{sp} - \tilde{y}_c| \tag{9.26}$$

This controller is for applications with *Fuzzy Process Dynamics*, as defined in Section 9.2. However, this does not preclude the use of the proposed controller for systems with *Fuzzy Process Input* and/or *Fuzzy Process Measurement.*

The self-learning portion of the control scheme can be accomplished through the use of any of the identification algorithms discussed in Chapters 7 and 8. The identification algorithm is a separate module from the controller and should be chosen for model accuracy for the specific system being controlled. However, learning of the process information should be in the form of a *fuzzy first order plus delay model* consistent with this controller development.

9.4.2 Process and Prediction Models

For this discussion of the controller design, define $\mathbf{u} \in \mathcal{U}$ and $\mathbf{y} \in \mathcal{Y}$ as the discrete input and output (i.e. **u** and **y** are non-italic), respectively. And, let $u = \{u_i \mid i = \{1, 2, \ldots, m\}\} \in \mathcal{U}$ and $y = \{y_l \mid l = \{1, 2, \ldots, n\}\} \in \mathcal{Y}$ be the *fuzzy* spaces of input and output (i.e. *u* and *y* are *italic*), respectively, all defined on the finite fuzzy universes of discourses indicated.

The *Process Model* is assumed to be the first order plus delay, τ, fuzzy state model

$$y(k+1) = \boldsymbol{R} \circ y(k) \circ u(k-\tau) \tag{9.27}$$

where $\circ \in O$

O stands for the family of composition (i.e. *max-min*, *max-product*, etc.)

τ is the time delay

The *Prediction Model* is also a first order plus delay, τ, fuzzy state model which, since there is no explicit disturbance modeling, is obtained by simply shifting the time index in (9.27).

$$y(k+\tau+1) = \boldsymbol{R} \circ y(k+\tau) \circ u(k) \tag{9.28}$$

9.4.3 Calculation of the Controller Output

For a deterministic system, a 1-step ahead predictive model is a function of the current input and the past known inputs.

$$\mathbf{y}(k+1) = f(\mathbf{u}(k), \mathbf{u}(k-1), \mathbf{u}(k-2), \ldots) \tag{9.29}$$

The basic approach used in model-based predictive control is to develop a model such as (9.29) that relates the process input and output values. The future *predicted* output value is then set equal to the desired or setpoint value at that future time. The current control action, $\mathbf{u}(k)$, is calculated so that the predicted output will be equal to or *close* to the desired value. For a N_2-step ahead predictive model, the number of unknown inputs increases to N_2,

$$\mathbf{y}(k+N_2) = f(\mathbf{u}(k+N_2-1), \mathbf{u}(k+N_2-2), \ldots, \mathbf{u}(k), \mathbf{u}(k-1), \ldots)) \tag{9.30}$$

and at the k-th time instant, $\mathbf{u}(k+N_2-1), \mathbf{u}(k+N_2-2), \ldots, \mathbf{u}(k)$ are all unknown.

Some model-based predictive controller solve for all the elements of the future control vector $\{u(k+N_2-1), u(k+N_2-2), \ldots, u(k)\}$ using a search or estimation algorithm. However, the simplest approach, and the one used here, is to calculate $u(k)$ assuming that the controller output is held constant, $\{u(k+i) = u(k), i = 1, 2, \ldots, N_2-1\}$. In practical applications, the control calculation is repeated at every control interval and hence the actual controller output may change at each interval.

In practice, N_2 represents that actual process delay, τ or d, plus the unit delay associated with the discrete process, plus an additional prediction added to detune the controller. In order to calculated ongest possible control action $N_2 = \tau + 1$ in the following controller development.

In the *ideal* model-based controllers, discussed in Section 9.3, if $\mathbf{G}_c = \mathbf{G}_p^{-1}$ then $y = y_{sp}$ for all time which represents perfect output control. However, in *practical* control algorithms it is necessary to include a tuning parameter to accommodate individual user preferences, model-process-mismatch, noise, robustness etc. The approach adopted for the *fuzzy predictive controller* being developed was:

(1) to calculate the *mean-level* or *steady state* control action, $u_{gain}(k)$, which will result in $y(k+i) \rightarrow y_{sp}(k+i)$ for large values of i, (i.e. at steady state). This equates to the *smallest* single-step control action that will achieve the desired setpoint.

(2) to calculate the *one-step-ahead* or *deadbeat* control action, $u_{dync}(k)$, that will drive the output from its current state to the desired state in one control interval, $y(k+1) = y_{sp}(k+1)$. This is the *strongest* possible control action.

(3) to calculate the *actual* controller output as a linear combination of $u_{gain}(k)$ and $u_{dync}(k)$

$$u(k) = \alpha \cdot u_{gain}(k) + (1-\alpha) \cdot u_{dync}(k) \tag{9.31}$$

so that the user can specify conservative, mean-level control ($\alpha = 1$), aggressive deadbeat control ($\alpha = 0$) or any combination ($0 < \alpha < 1$).

The procedure for calculating the controller output is illustrated by Figure 9.16 and discussed in the following three subsections.

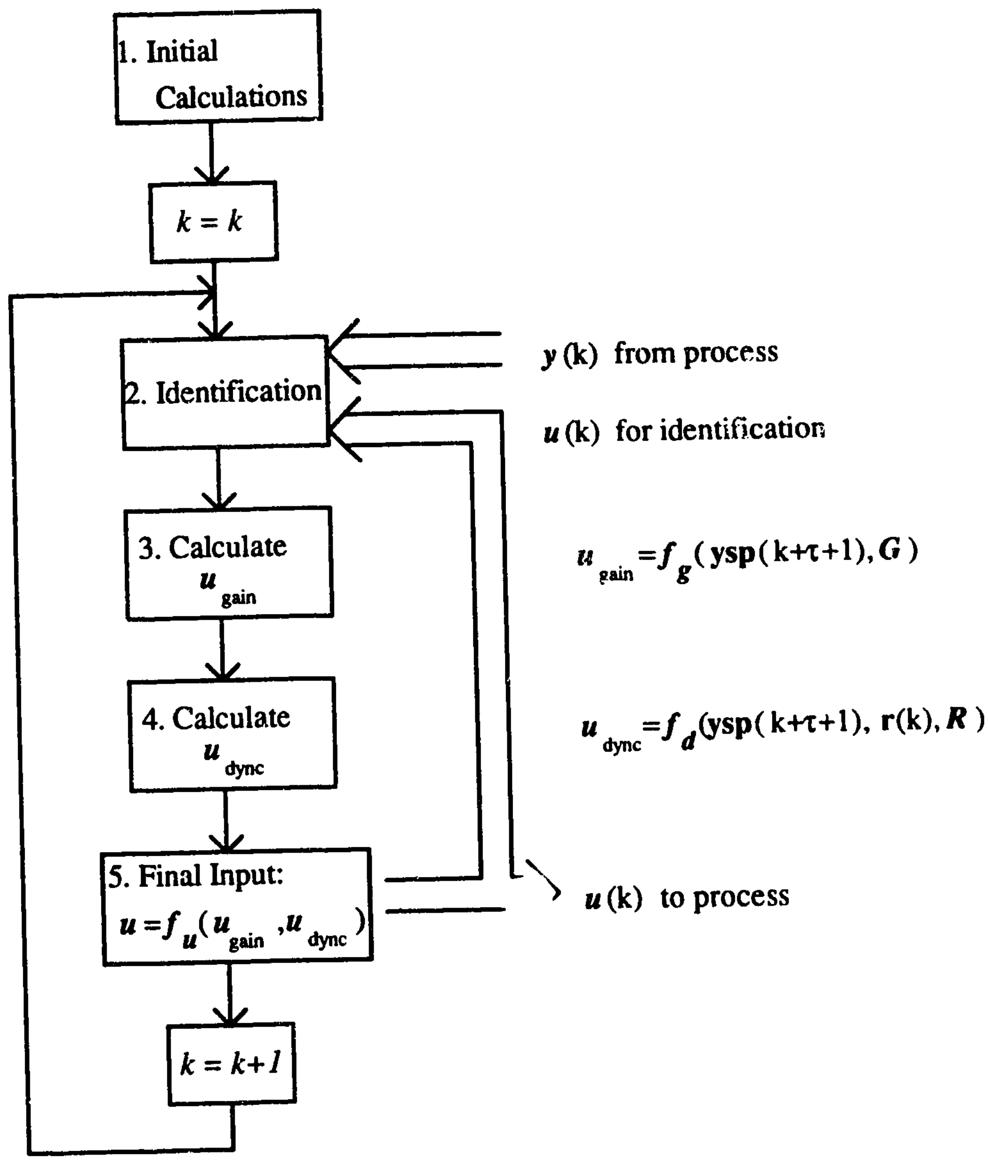


Figure 9.16: Flow Diagram of the Predictive Controller

9.4.3.1 Calculation of the Mean-Level Control Action, $\mathbf{u}_{gain}$

The mean-level control action, $\mathbf{u}_{gain}(k)$, is calculated using a relational ***gain*** matrix, $\boldsymbol{G}(y(k+\tau+1), \boldsymbol{u}(k))$:

$$\therefore \quad \boldsymbol{u}_{gain}(k) = \boldsymbol{G} \circ \boldsymbol{y}_{sp}(k+\tau+1) \tag{9.32}$$

Thus the discrete gain input, $\mathbf{u}_{gain}(k) = \text{defuzz}(\boldsymbol{u}_{gain}(k))$, results in an output trajectory that is equal to the natural open-loop step response of a stable process to reach the setpoint. The output response from this input represents the slowest or most conservative control move and is usually the lower limit of acceptable responses.

9.4.3.2 Calculation of the One-Step-Ahead or Deadbeat Control Action, $\mathbf{u}_{dync}$

A second value for current unknown input, $\mathbf{u}(k)$, is calculated using a predictive fuzzy algorithm based on the identified relational matrix, $\boldsymbol{R}$, so these calculations take into account the process ***dynamics***. The strategy for the development of this dynamic input calculation, $\mathbf{u}_{dync}(k)$, is that the output should move from the current state to the desired setpoint state within one control interval. Thus the output response is considered to be fast or *deadbeat*, and is therefore the upper limit of possible responses. The development of these calculations is as follows.

Information available at time k includes the process model and the predictive model [Pedrycz, 1981]:

Process Model: $$\boldsymbol{y}(k) = \boldsymbol{R}_{k-\tau-1} \circ \boldsymbol{y}(k-1) \circ \boldsymbol{u}(k-\tau-1) \tag{9.33}$$

Predictive Model: $$\boldsymbol{y}(k+\tau+1) = \boldsymbol{R}_{k-\tau-1} \circ \boldsymbol{y}(k+\tau) \circ \boldsymbol{u}(k) \tag{9.34}$$

where $\boldsymbol{R}_{k-\tau-1}$ is the model identified at the k-th time instant using $\boldsymbol{y}(k)$, $\boldsymbol{y}(k-1)$, and $\boldsymbol{u}(k-\tau-1)$. The predicted output in equation (9.34) is set to the required setpoint value,

$$\boldsymbol{y}(k+\tau+1) = \boldsymbol{y}_{sp}(k+\tau+1) \tag{9.35}$$

and $\boldsymbol{u}(k)$ is determined. Since $\boldsymbol{y}(k+\tau)$ is unknown, equation (9.34) must be re-written such that $\boldsymbol{y}(k+\tau+1)$ is dependent on known values, that is:

$$\boldsymbol{y}(k+\tau+1) = f(\boldsymbol{y}(k), \boldsymbol{u}(k)). \tag{9.36}$$

The required calculations are:

$$y(k+\tau) = \boldsymbol{R}_{k\text{-}1} \circ y(k+\tau-1) \circ \boldsymbol{u}(k-1) \tag{9.37}$$

$$y(k+\tau-1) = \boldsymbol{R}_{k\text{-}2} \circ y(k+\tau-2) \circ \boldsymbol{u}(k-2) \tag{9.38}$$

$$y(k+\tau-2) = \boldsymbol{R}_{k\text{-}3} \circ y(k+\tau-3) \circ \boldsymbol{u}(k-3) \tag{9.39}$$

$$\vdots$$

$$y(k-1) = \boldsymbol{R}_{k\text{-}\tau} \circ y(k) \circ \boldsymbol{u}(k-\tau) \tag{9.40}$$

Substituting equations (9.38), (9.39) etc. into equation (9.37):

$$y(k+\tau) = [\boldsymbol{R}_{k\text{-}1} \circ \boldsymbol{u}(k-1)] \circ [\boldsymbol{R}_{k\text{-}2} \circ \boldsymbol{u}(k-2)] \quad \circ \ldots \circ [\boldsymbol{R}_{k\text{-}\tau} \circ \boldsymbol{u}(k-\tau)] \circ \; y(k) \tag{9.41}$$

$$\therefore \quad y(k+\tau) = \boldsymbol{K}_{k\text{-}1} \circ \boldsymbol{K}_{k\text{-}2} \circ \ldots \circ \boldsymbol{K}_{k\text{-}\tau} \circ y(k) \tag{9.42}$$

where $\boldsymbol{K}_{k\text{-}1} = \boldsymbol{R}_{k\text{-}1} \circ \boldsymbol{u}(k-1)$, etc. are all known values.

Combining equation (9.42) with equation (9.34):

$$y(k+\tau+1) = \boldsymbol{K}_{k\text{-}1} \circ \boldsymbol{K}_{k\text{-}2} \circ \ldots \circ \boldsymbol{K}_{k\text{-}\tau} \circ y(k) \circ \boldsymbol{u}(k) \tag{9.43}$$

as required. Since $y(k)$ and the $\boldsymbol{K}$'s are all known at the k-th time instant, efficient on-line calculations can be obtained by combining these values once at the start of the sampling interval such that:

$$y(k+\tau+1) = \boldsymbol{Q} \circ \boldsymbol{u}(k) \tag{9.44}$$

where $\boldsymbol{Q} = \boldsymbol{K}_{k\text{-}1} \circ \boldsymbol{K}_{k\text{-}2} \circ \ldots \circ \boldsymbol{K}_{k\text{-}\tau} \circ y(k)$

The results from Chapter 8 show that minimizing the fuzzy control error does not minimize the discrete control error:

$$\min | y_{sp}(k) - \tilde{y}(k) | \not\Rightarrow \min | \mathrm{y}_{sp}(k) - \tilde{\mathrm{y}}(k) | \tag{9.45}$$

Because of this, the calculated value of the fuzzy dynamic input is adjusted iteratively by minimizing the discrete difference, $|\mathrm{y}_{sp}(k+\tau+1) - \tilde{\mathrm{y}}(k+\tau+1) - \mathrm{r}(k)|$ at each step. The residual, $\mathrm{r}(k)$, as implied by Figure 9.14 and 9.15 includes the structured disturbance, $\mathrm{L}(k)$, the measurement noise, and the model-process-mismatch. In the proposed fuzzy controller, the *filtered* value of $\mathrm{r}(k)$ is calculated from:

$$\mathbf{r}(k) = \eta\cdot(\mathbf{y}(k) - \tilde{\mathbf{y}}(k)) + (1-\eta)\cdot(\mathbf{y}_{sp}(k) - \tilde{\mathbf{y}}(k)) - (\mathbf{y}_{sp}(k+\tau+1) - \tilde{\mathbf{y}}(k+\tau+1)) \qquad (9.46)$$

In equation (9.46), the first difference, $\mathbf{y}(k) - \tilde{\mathbf{y}}(k)$, would be equal to the disturbance, $\mathbf{L}(k)$, if the modeling were exact and the second difference, $\mathbf{y}_{sp}(k) - \tilde{\mathbf{y}}(k)$, is a measure of the *prediction* or *modeling* error. The parameter $0 < \eta < 1$ is chosen by the control system designer to determine the weighting of the disturbance versus the modeling error in the calculation of $\mathbf{r}(k)$. The third difference is added to improve the prediction capacity of the fuzzy logic controller. The value $\tilde{\mathbf{y}}(k+\tau+1)$ is the defuzzified value of the prediction estimate:

$$\tilde{\mathbf{y}}(k+\tau+1) = \boldsymbol{R}_{k-\tau-1}\circ\mathbf{y}(k)\circ\boldsymbol{u}(k-1) \qquad (9.47)$$

The fuzzy input $\boldsymbol{u}_{dync}(k)$ is calculated by iteratively searching for the process input that minimizes the discrete error, $|\mathbf{y}_{sp}(k+\tau+1) - \tilde{\mathbf{y}}(k+\tau+1) - \mathbf{r}(k)|$. The search is made by adjusting the values of the individual components of the fuzzy input vector as a function of the discrete error. This calculation is performed as described below.

Assume that the control application has a fuzzy input vector of dimension 5 with an initial value of $[\,0\;\; a\;\; b\;\; c\;\; 0]$. The initial value could be equal to $\boldsymbol{u}_{gain}(k)$ or the previous $\boldsymbol{u}_{dync}(k-1)$. Let the discrete error be, $\mathbf{e} = \mathbf{y}_{sp}(k+\tau+1) - \tilde{\mathbf{y}}(k+\tau+1) + \mathbf{r}(k)$. Then:

$$\boldsymbol{u}(k) = \{0\;\; \mathrm{a}-f(\mathbf{e})\;\; \mathrm{b}\;\; \mathrm{c}+f(\mathbf{e})\;\; 0\} \qquad \text{if } \mathbf{e} > 0 \qquad (9.48)$$

$$\boldsymbol{u}(k) = \{0\;\; \mathrm{a}+f(\mathbf{e})\;\; \mathrm{b}\;\; \mathrm{c}-f(\mathbf{e})\;\; 0\} \qquad \text{if } \mathbf{e} < 0 \qquad (9.49)$$

where $$f(\mathbf{e}) = s\cdot\gamma\cdot|\mathbf{e}| \qquad (9.50)$$

and $$s = \begin{cases} +1 & \text{if } \mathbf{u}\uparrow \Rightarrow \mathbf{y}\uparrow \\ -1 & \text{if } \mathbf{u}\uparrow \Rightarrow \mathbf{y}\downarrow \end{cases} \qquad (9.51)$$

$\gamma \geq 1$ is a tuning parameter for the convergence rate.

The adjustment of the fuzzy input vector, $\boldsymbol{u}_{dync}(k)$, by equations 9.47 to 9.49 is carried out iteratively until the error, $\mathbf{e}$, is sufficiently small but with a maximum on the iterations per control interval (set to 10 for examples in this thesis).

The process input calculated by this *iterative algorithm* may not be aggressive enough for the process to reach the setpoint by the next step. This is mainly due to the *poor* predi ive capabilities of the relational matrix for large N_2. To compensate for any shortfall that may r ult from the calculated discrete input, $\mathbf{u}(k) = \text{defuzz}(\boldsymbol{u}(k))$, an input tuning factor, $a(k)$, is in oduced, based on the final error, **e**, of the iterative input procedure.

$$a(k) = 1 + \beta \cdot |e| \tag{9.52}$$

where $0 \leq \beta \leq 1$ determines the aggressiveness of the input tuning factor.

If the final predictive error, e, is very small, the tuning factor $a(k) \approx 1$, and there is no adjustment to the input value that was obtained from the iterative search algorithm.

The input tuning factor, $a(k)$, is applied to the change in the input, $\Delta\mathbf{u}(k)$, defined by:

$$\Delta\mathbf{u}(k) = \mathbf{u}(k) - \mathbf{u}(k-1) \tag{9.53}$$

The final calculated dynamic input, $\mathbf{u}_{dync}(k)$, is calculated:

$$\mathbf{u}_{dync}(k) = \mathbf{u}(k-1) + a(k)\cdot\Delta\mathbf{u}(k) \tag{9.54}$$

where $\mathbf{u}(k-1)$ is the control action actually sent to the process at the previous control interval. The calculated controller output value, $\mathbf{u}_{dync}(k)$, represents an aggressive 1-step ahead deadbeat type control move.

Song *et al.* [1993] also use an aggression or reinforcement factor similar to $a(k)$ which is applied to the control action in the next cycle. For their *PI* controller design, these authors compute the reinforcement from a fixed look-up table based on *error* and *change of error*. The final *PI* design is tested with a second order plus dead time process, where $\tau = 1$, with good results.

9.4.3.3 Calculation of the Final Controller Output

The calculations of $\mathbf{u}_{gain}(k)$ and $\mathbf{u}_{dync}(k)$ are now cor plete. The final input value to be implemented in the process is a weighted average of the ***gain*** (mean-level) input, $\mathbf{u}_{gain}(k)$, and the ***dynamic*** (deadbeat) input, $\mathbf{u}_{dync}(k)$ and is therefore a balance between the two input extremes.

$$\mathbf{u}(k) = \alpha\cdot\boldsymbol{u}_{gain}(k) + (1-\alpha)\cdot\mathbf{u}_{dync}(k) \tag{9.55}$$

where $0 < \alpha < 1$ and is a function of the *accuracy* of the predictive model.

9.4.4 Schematic of the Fuzzy Predictive Controller

A simplified block diagram of the iterative algorithm to determine the process input is illustrated in Figure 9.17. The iterative calculations take place inside the dashed block at each control interval.

The next section details the steps of the entire *fuzzy predictive controller*.

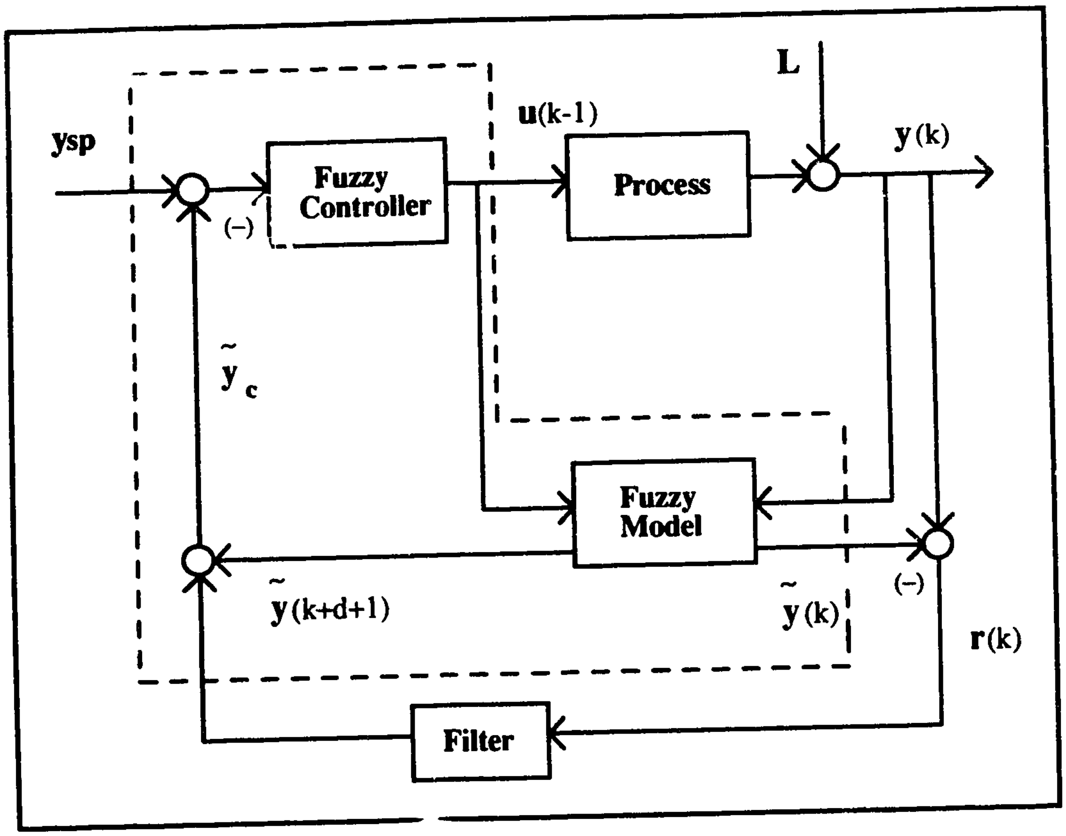


Figure 9.17: Iterative Control Algorithm
(discrete / *fuzzy*, *sp ≡ setpoint*, *~ ≡ estimate*)
(d = τ = delay)

9.5 *Fuzzy* Predictive Controller Algorithm

The final block diagram of the *fuzzy predictive controller* is illustrated in Figure 9.18. Figure 9.18 is virtually identical to the self-learning predictive control scheme of Figure 9.15, except that the modules for fuzzification and defuzzification have been include for completeness.

Each component of this controller design has been investigated separately, in the various chapters of this thesis, and analyzed for theoretical soundness and applicational practicality. These investigations include:

Fuzzification
- a fuzzification methodology to ensure referential set completeness and compatibility between the input and output fuzzy data

Identification
- an on-line identification algorithm that is computationally efficient and able to detect and adapt to process changes

Modeling
- a first order plus time delay model developed using relational equations as the basis for the controller design
- the *max-product* composition operator is used as the fuzzy inferencing mechanism based on its superior performance

Controller
- the controller design is that of a *fuzzy k-step predictive controller*
- it is composed of 5 parts:

  (1) Initial calculations

  (2) Identification

  (3) Gain Input Calculations

  (4) Dynamic Input Calculations

  (5) Weighted Average Input Calculation

Defuzzification
- the algorithm for defuzzification is the commonly used *fuzzy area method*

Step-by-step detail of the controller algorithm is available in Appendix 4. Application of this controller algorithm, under various process situations, is demonstrated in Chapter 10.

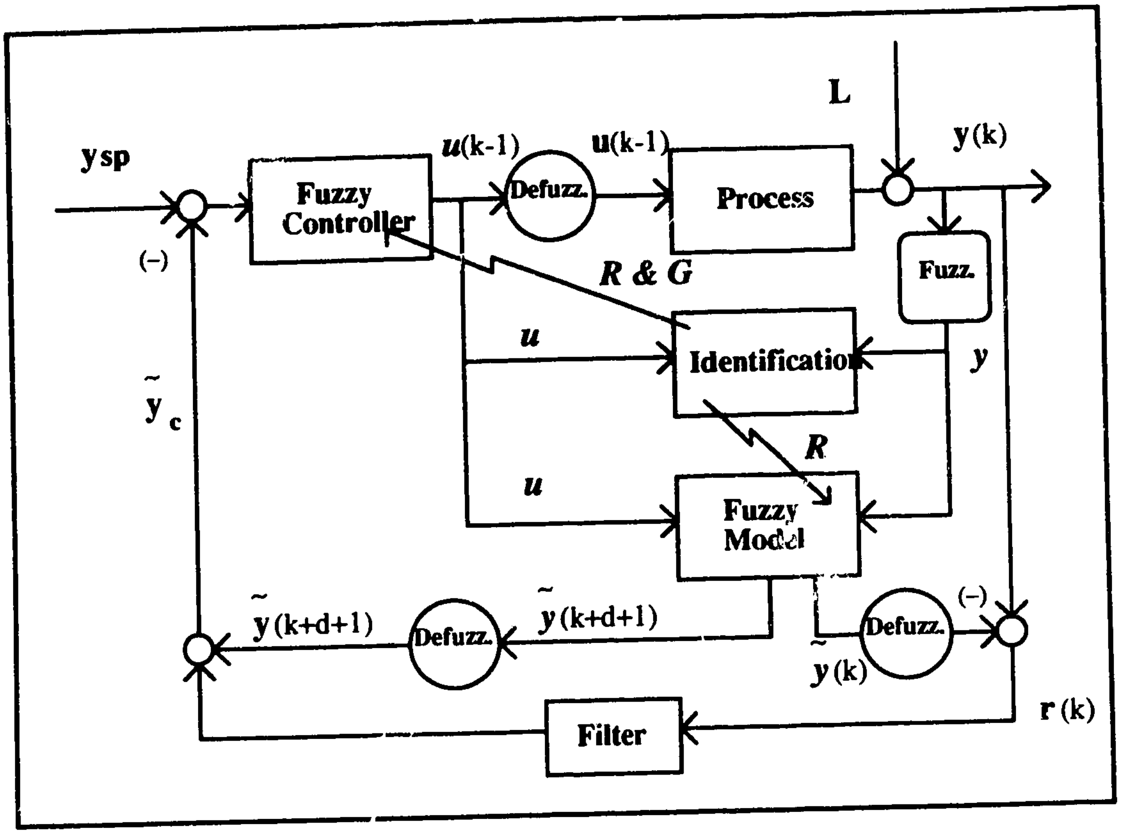


Figure 9.18: Self Learning Predictive Fuzzy Logic Controller
(discrete / *fuzzy*, *sp ≡ setpoint*, ~ ≡ *estimate*)
(d = τ = delay)

9.6 Controller Contributions

This proposed controller design is non-application specific and therefore applicable to a wide range of open-loop stable industrial problems. The main contribution associated with the completion of this thesis on ***self-learning predictive fuzzy control*** is an industrially viable controller design, applicable to areas of *fuzzy* technology, such as water quality control, [illegible] processing, or building environmental controls, where goals are not quantitatively defined. The distinguishing features of this design which characterize the contribution are:

(*i*) use of a relational-based model which is superior to the traditional rule-based model in that it permits numerical and analytical analysis

(*ii*) employment of the *max-product* compositional operator which has been shown to provide *better* results, in a least squares sense, than the traditional *max-min* compositional operator

(*iii*) incorporation of self learning or adaptive identification capabilities

(*iv*) predictive model-based controller structure

(*v*) a generalized design which is not application specific and contains enough flexibility so that it can be easily modified for different classes of applications

The development and integration of these features results in an original fuzzy controller design based on relational matrices. This work is a contribution to the area of *fuzzy control* in that it is model-based (as opposed to a relational matrix formulation of *PI* control) and parallels the development of discrete model-based control.

REFERENCES

[1] Clarke, D.W., Mohtadi, C., "Generalized Predictive Control: Part I - The Basic Algorithm; Part II - Extensions and Interpretations", *Automatica*, Vol. 23, pp. 137-160, 1987.

[2] Cutler, C.R., Ramaker, B.L., "Dynamic Matrix Control - A Computer Control Algorithm", *Proceedings of the 1980 Joint Automatic Control Conference*, pp. WP5-B, San Francisco, 1980.

[3] Gottwald, S., Pedrycz, W., "Problems of the Design of Fuzzy Controllers", In: *Approximate Reasoning in Expert Systems*, Gupta, M.M., Kandel, A., Bandler, W., Kiszka, J.B. (Editors), Elsevier Science Publishers B.V., North-Holland, pp. 393-405, 1985.

[4] Graham, B.P., Newel, R.B., "Fuzzy Identification and Control of a Liquid Level Rig", *Fuzzy Sets and Systems*, Vol. 26, pp. 255-273, 1988.

[5] Graham, B.P., Newell, R.B., "Fuzzy Adaptive Control of a First-Order Process", *Fuzzy Sets and Systems*, Vol. 31, pp. 47-65, 1989.

[6] Kickert, W.J.M., Mamdani, E.H., "Analysis of a Fuzzy Logic Controller", *Fuzzy Sets and Systems*, Vol. 1, pp. 29-44, 1978.

[7] Maeda, M., Murakami, S., "A Design for a Fuzzy Logic Controller", *Information Sciences*, Vol. 45, pp. 315-330, 1988.

[8] Mamdani, E.H., Assilian, S., "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", *International Journal of Man-Machine Studies*, Vol. 7, pp. 1-13, 1975.

[9] Pedrycz, W., "An Approach to the Analysis of Fuzzy Systems", *International Journal of Control*, Vol. 34, pp. 403-421, 1981.

[10] Pedrycz, W., "Design of Fuzzy Control Algorithms with the Aid of Fuzzy Models", In: *Industrial Applications of Fuzzy Control*, Sugeno, M. (Editor), Elsevier Science Publishers, North-Holland, pp. 153-173, 1985.

[11] Pedrycz, W., *Fuzzy Control and Fuzzy Systems, Second Extended Edition*, John Wiley & Sons, Inc., New York, N.Y., 1993.

[12] Richalet, J., Rault, A., Testud, J.L., Papon, J., "Model Predictive Heuristic Control: Application to Industrial Processes", *Automatica*, Vol. 14, pp. 413-417, 1978.

[13] Rivera, D.E., Morari, M., "Internal Model Control: PID Controller Design", *Ind. Eng. Chem. Process Des. Dev.*, Vol. 25, pp. 252-265, 1086.

[14] Song, J.J., Park, W., "A Fuzzy Dynamic learning Controller for Chemical Process Control". *Fuzzy Sets and Systems*, Vol. 54, pp. 121-133, 1993.

[15] Tong, R.M., "Analysis of Fuzzy Control Algorithms using the Relation Matrix". *International Journal of Man-Machine Studies*, Vol. 8, pp. 679-686, 1976.

CHAPTER 10 SIMULATION RESULTS

In our view, it is this fuzzy, and as yet not well-understood, logic that plays a basic role in what may well be one of the most important facets of human thinking, namely, the ability to summarize information.

[Zadeh, 1973]

10.1 Introduction

This Chapter presents the simulation results from applications of the predictive fuzzy logic controller developed in Chapter 9. The controller is applied to several processes situations (i.e. high order overdamped and underdamped; inverse non-linear and inverse overdamped with dead time systems). Results of these simulations clearly show that the predictive controller is capable of *good* control, in the least squares sense, for the test cases used. Additionally, the predictive fuzzy logic controller is compare against a PI controller for a highly non-linear process. The results of this comparison show that fuzzy logic is most applicable to areas where conventional control fails.

10.2 The *Fuzzy* Model

A first order fuzzy state space model is used for the identification and modeling regardless of the order of the actual process. Let $u=\{u_i \mid i = \{1,2, ..., m\}\} \in \mathcal{U}$, $x=\{x_j \mid j = \{1,2, ..., n\}\} \in \mathcal{X}$, and $y=\{y_l \mid l = \{1,2, ..., n\}\} \in \mathcal{Y}$ be the *fuzzy* spaces of input, state and output, respectively, all defined on the finite fuzzy universes of discourses indicated. Then for a series of N data points:

$$y_k = x_{k-1} \circ u_{k-\tau-1} \circ R \tag{10.1}$$

where $x_{k-1} = y_{k-1}$

and τ is the time delay.

Since the data generated from the simulation studies is discrete, it must be *fuzzified* and the resulting controller output *defuzzified*. Partitioning of the data is subjective [Pedrycz, 1983, Xu *et al.* 1987], however $\rho = 5$ referential fuzzy sets (or clusters) appeared adequate for both the input and the output. The shape of the referential fuzzy sets was chosen to be isosceles triangles, which satisfy the requirements of being normal and convex. These fuzzy membership functions were place evenly over the input-output space so that the space was *completely covered* and the referential sets had a 50% overlap.

The *defuzzification* formula was chosen to be the weighted average of the center of area method which uses all the information in the *fuzzy* output vector.

$$y = \frac{\sum_i c_i \cdot \mu_i}{\sum_i \mu_i} \tag{10.2}$$

where c_i is the center of area of the i-th reference fuzzy set

μ_i is the degree of membership in the i-th reference fuzzy set

The minimization criterion used to judge the performance of the controller is based on the difference between the *defuzzified* or discrete output and the output setpoint and is defined in equation 10.3.

$$J_q = \frac{\sum_{k=\tau+2}^{N} |ysp_k - \tilde{y}_k|^q}{N - \tau - 1} \tag{10.3}$$

where $q = 1, 2$

10.3 The Simulation Data

Four process test cases are used to evaluate the fuzzy logic predictive control algorithm. These include an overdamped process, an underdamped process, a non-linear negative steady state gain process and a negative steady state gain process with dead time. The discrete models used for these simulations are listed in this section along with additional information about the individual processes (i.e. noise, disturbances). The simulation results are discussed later in Section 10.5 with Figure 10.5 to 10.21.

10.3.1 Overdamped Model

Since most chemical processes are high order and over-damped, the test data for this scenario has been generated from an arbitrary second order over-damped process with unit gain. A unit gain was chosen for this process to simplify fuzzification and defuzzification since the same membership function can be used for each calculation. There was no time delay for this test data. The overdamped process model is:

$$y_k = a_1 y_{k-1} + a_2 y_{k-2} + b_1 u_{k-1} + b_2 u_{k-2} \qquad (10.4)$$

where $a_1 = 0.8518$

$a_2 = -0.097$

$b_1 = 0.1672$

$b_2 = 0.0779$

The open-loop step response of process (10.4) is shown in Figure 10.1

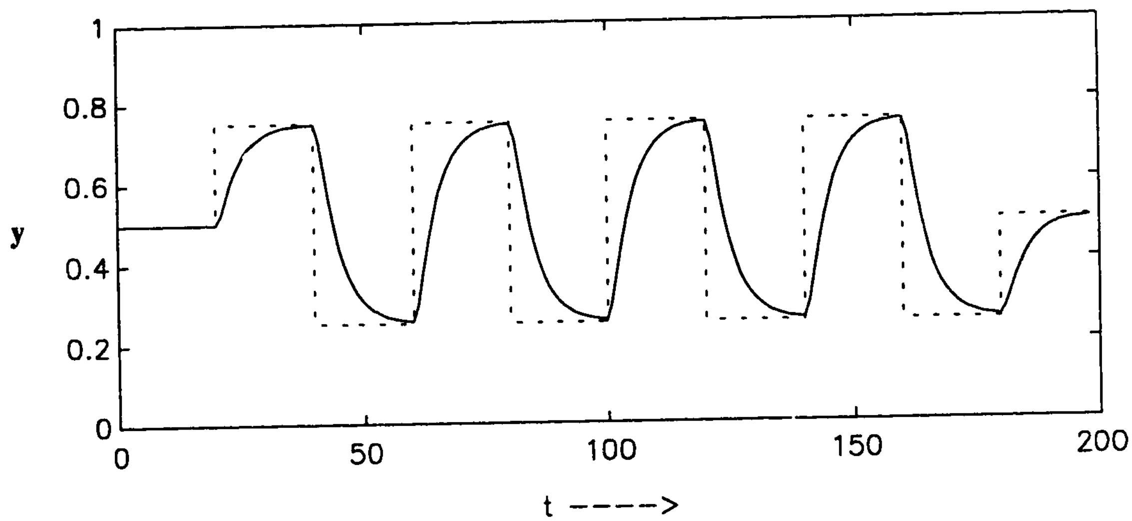


Figure 10.1: Open-Loop Step Response of the Overdamped Process
(Actual ____ ; Setpoint - - - -)

10.3.2 Underdamped Model

In order to show the flexibility of the controller, a fast-responding underdamped second order process with unit gain was then tested. Again, no time delay was added to this data. The underdamped process model is:

$$\mathbf{y}_k = \mathbf{a}_1\,\mathbf{y}_{k-1} + \mathbf{a}_2\,\mathbf{y}_{k-2} + \mathbf{b}_1\,\mathbf{u}_{k-1} + \mathbf{b}_2\,\mathbf{u}_{k-2} \tag{10.5}$$

where $\mathbf{a}_1 = 1.5$

$\mathbf{a}_2 = -0.5$

$\mathbf{b}_1 = -0.5$

$\mathbf{b}_2 = 0.5$

The open-loop step response of process (10.5) is shown in Figure 10.2

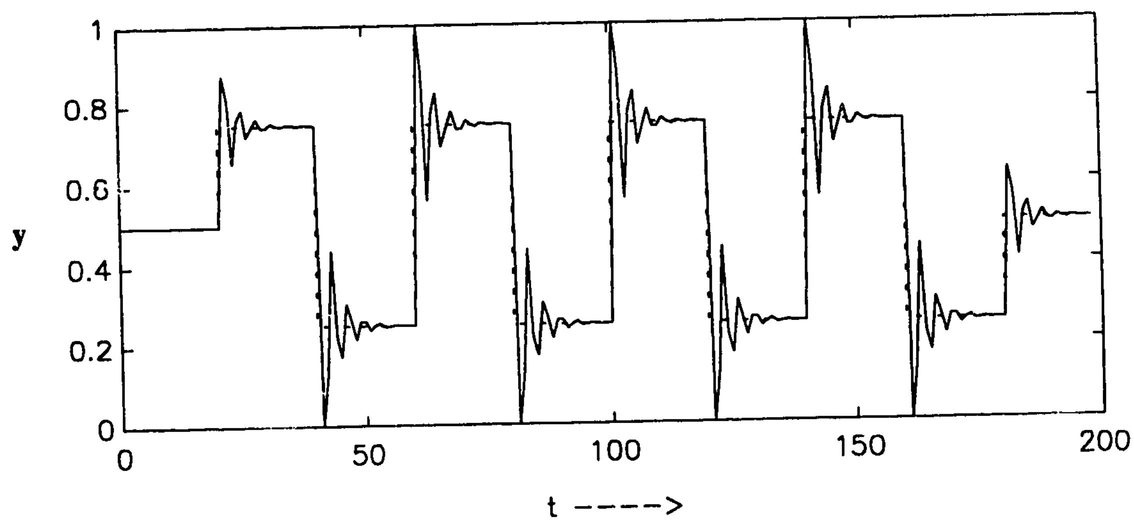


Figure 10.2: Open-Loop Step Response of the Underdamped Process (Actual ____ ; Setpoint - - - -)

10.3.3 Nonlinear Negative Gain Model

A highly nonlinear negative steady state gain model was used for the third example. The model is a first order model with a hyperbolic input, followed by a sigmoid function [Hernández *et al.*, 1991] so that the system is physically bounded in [0, 1].

$$y(k) = \sigma(a_1 y(k\text{-}1) + b_1 \sinh(u(k\text{-}1))^4) \tag{10.6}$$

where $a_1 = 0.5$
$b_1 = 0.75$

The sigmoid function is:

$$\sigma(x) = \frac{1}{1+\exp(-x)} \tag{10.7}$$

The open-loop step response of process (10.6) is shown in Figure 10.3

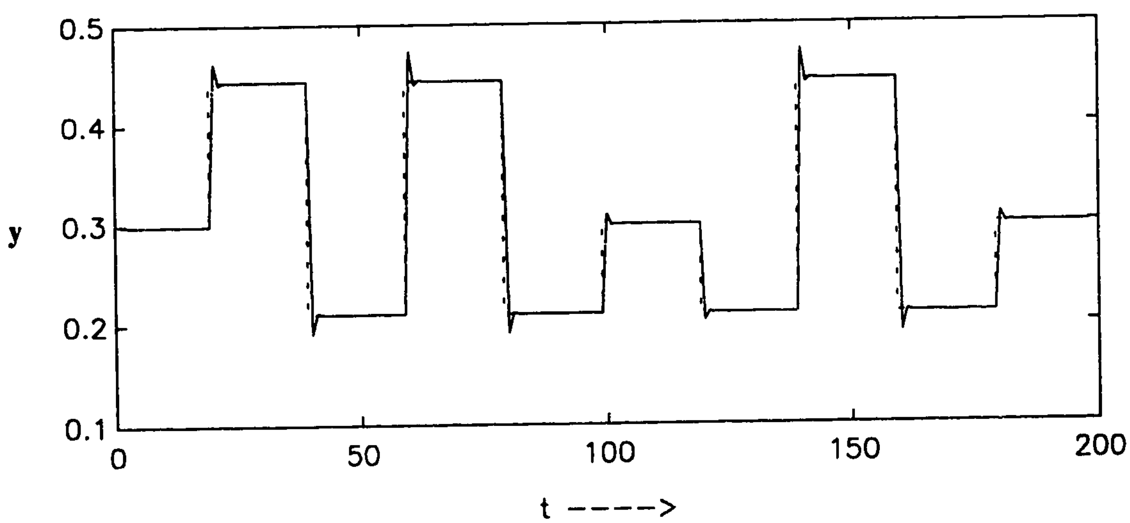


Figure 10.3: Open-Loop Step Response of the Negative Gain Process (Actual ____ ; Setpoint - - - -)

No noise or disturbances were added to this system in order to demonstrate the accuracy of the fuzzy logic controller.

10.5.4 Shell Process Model

The Shell process data for the distillate bottoms impurity, made available for the 1992 Canadian Chemical Engineering Process Identification Workshop [Cott, 1995(a); Arifin *et al.*, 1995; Bailey, 1995; Chan, 1995, Cott, 1995(b); Banerjee *et al.*, 1995], was also used for the control application. This data is particular applicable since the problem has proven to be realistic and relevant from a process point of view. The significant features of this data are that the process has a negative, mildly non-linear steady-state gain relationship between the input and output data and the length of the time delay is 6 sampling instances. The process model is

$$\mathbf{X}(k) = 0.0765 \cdot \frac{500000}{\mathbf{Q}(k-7)-1500} + 0.9235 \cdot \mathbf{X}(k\text{-}1) \tag{10.8}$$

where **X** is bottoms impurity

Q is the reboiler duty

The open-loop step response of process (10.8) is shown in Figure 10.4

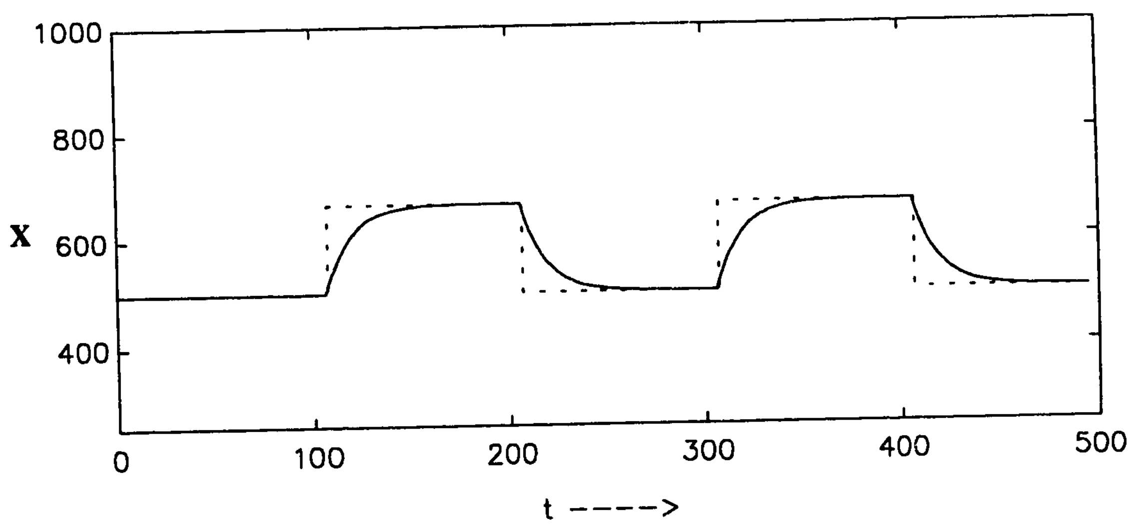


Figure 10.4: Open-Loop Step Response of Shell Process
(Actual ____ ; Setpoint - - - -)

10.4 Tuning and Implementation

There are six tuning parameter associated with the *fuzzy* predictive control.

- α varies the weighting between u_{gain} and u_{dync} for the final value of u [Eqn. 9.31].
- β varies the intensity of the tuning factor *a* for u_{dync} [Eqn. 9.52].
- η a filter parameter that varies the weighting between current actual error and predicted error [Eqn. 9.46].
- ε is the difference tolerance (i.e. $|y_{sp} - \tilde{y}| \leq \varepsilon$) for testing the convergence of the u_{dync} search algorithm and identification of G [See Appendix 4 - Section (4) - Dynamic Input Calculations and Section (2) - Identification, respectively].
- γ is the tuning factor for the iterative input search algorithm [Eqn. 9.50].
- ω a filter parameter that varies the length of the window over which the errors are averaged [See Appendix 4 - Section (3) - Gain Input Calculations].

The first three parameters regulate the smoothness and accuracy of the control. These parameters required the most tuning in order to obtain acceptable control. The last three parameters were arbitrarily set and maintained at the same value for all the simulations performed. In other words, no tuning was performed on the parameters ε, γ, and ω. That is not to say that they can not be tuned, just that the values for these parameters, determined during preliminary testing, were found to be adequate for all the simulations presented.

Two relational matrix models are required for this predictive controller, the dynamic matrix model, $\boldsymbol{R}$, and the gain matrix model, $\boldsymbol{G}$. Since the gain model is a steady state model which relates steady state input to steady state output, the dimensions of this relational matrix can be larger than the dynamic matrix model. For example an 11×11 two dimensional gain model requires less computation than a 5×5×5 three dimensional model. With the increased size of the gain model there is an increase in of accuracy for the controller, particularly when the process is close to the setpoint.

Additionally, when the gain model, $\boldsymbol{G}$, is identified by an averaging technique then the same model is valid for prediction of either input or output. The averaging identification technique for the gain matrix is:

$$G(u_{k-\tau-1}, y_k) = \sum_{k=\tau+2}^{N} \prod_{\substack{1\le i\le m \\ 1\le l\le p}} (u_i^{(k-\tau-1)}, y_l^{(k)}) \Bigg/ \sum_{k=\tau+2}^{N} \mathrm{K}_{\substack{1\le i\le m \\ 1\le p\le l}} (u_i^{(k-\tau-1)}, y_l^{(k)}) \tag{10.9}$$

where
$$\mathrm{K}(u,y) = \begin{cases} 1 & if \quad u \cdot y > 0 \\ 0 & otherwise \end{cases} \tag{10.10}$$

Because of the commutative property of the product operation:

$$G(u, y) = G^{-1}(y, u) \tag{10.11}$$

which is consistent with Definition 6 of Chapter 5. Thus the same model can be used to predict the steady state input:

$$u = G(u, y)\circ y \tag{10.12}$$

or predict output:

$$y = G^{-1}(u\ y)\circ u \tag{10.13}$$

The *product* operation in equation (10.10) can be replaced with the *minimum* operation and the preceding analysis would still be valid since the *minimum* operation is also commutative..

Finally, for tuning processes with a fast response the tuning factor $a(k)$ for the dynamic input calculation, $\mathbf{u}_{gain}(k)$, can be adjusted so that it is not as aggressive as required for overdamped process. Thus the control algorithm line:

$$a(k) = 1 + \beta|e_d(i)| \tag{10.14}$$

can be redefined for underdamped processed by replacing the 1 by 0. So that:

$$a(k) = \beta|e_d(i)| \tag{10.15}$$

This adjustment reduces the aggressiveness of the calculated input and therefore reduces overshoot.

10.5 Simulation Results

The following graphical results are presented for the 4 test cases discussed in Section 10.2. The 4 test cases were evaluated for both the *max-product* and *max-min* compositions. The tuning objective for these simulations was to minimize the J_q (q = 1 or 2) values, for each composition, while maintaining a *smooth* (i.e. non-oscillatory) control action.

For completeness and reproducibility of this work, the complete solution for the Shell Process Model (Section 10.5.4) is included in Appendix 5.

10.5.1 Overdamped Process

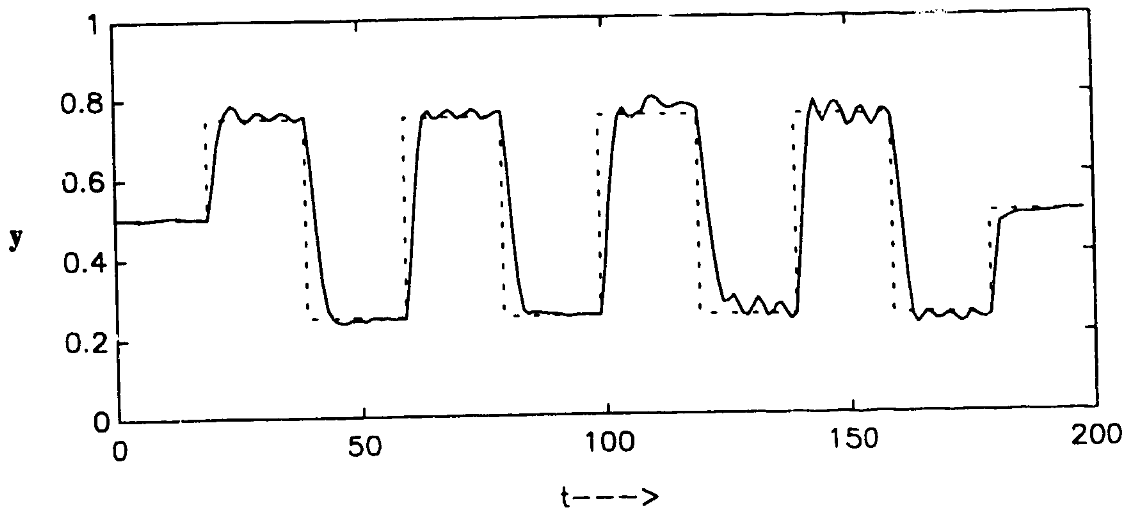
Using the overdamped process model of equation (10.4), white noise with standard deviation $\sigma = 0.002$ was added to the process output. A step disturbance of 0.03 output units was added at simulation time instances 110 and removed at 150. The disturbance profile for the overdamped process is shown in Figure 10.7

Simulation results using the overdamped process disturbance model show that the *max-product* and *max-min* compositions produce similar results, with the *max-product* composition being slightly *better*. The minimization criteria results for the *max-product* composition are J_1 = 0.0348 and J_2 = 0.0074 and for the *max-min* composition are J_1 = 0.0354 and J_2 = 0.0078. This simulation example was tuned so that the process reached the setpoint *quickly* (i.e. < open-loop response) while minimizing the overshoot. Control of the overdamped process encountered some difficulty with the disturbance step change, as shown in Figures 10.5 and 10.6, by a slight oscillatory response during the period of this disturbance (i.e. sampling instances 110-150). However, the self-learning during this process disturbance resulted in the reduction of the oscillatory behavior with time.

10.5.2 Underdamped Process

Using the underdamped process model of equation (10.5), white noise with standard deviation $\sigma = 0.002$ was added to the process output. A step disturbance of 0.05 output units was added at time instance 110 and removed at 150. The disturbance profile for the underdamped process is shown in Figure 10.10

For the underdamped process disturbance model the *max-product* composition results were again *better* (i.e. fewer/smaller overshoots) with J_1 = 0.0226 and J_2 = 0.0015 for the *max-product* composition versus J_1 = 0.0260 and J_2 = 0.0018 for the *max-min* composition. Tuning of this model, as shown in Figures 10.8 and 10.9, was conservative in order to minimize the overshoot of this very responsive process. The controller response to the step disturbances (at time 110 and 150) was less oscillatory than for the overdamped process discussed in Section 10.5.1. This improvement in the oscillatory response is attributed to the less aggressive controller tuning required for this process.



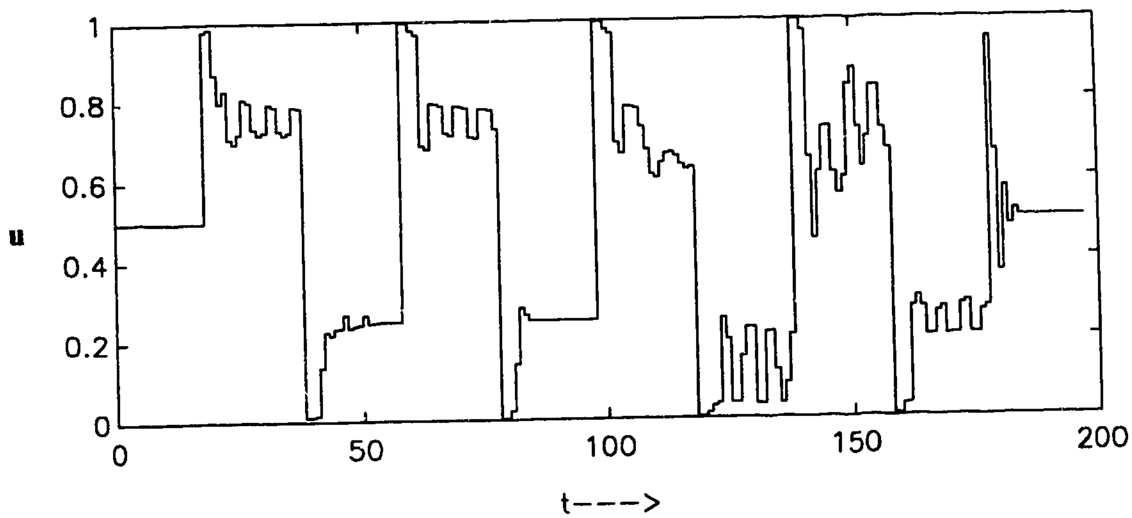


Figure 10.5: Control of Overdamped Process with *Max-product* Composition
(α = 0.15; β = 0.1; η = 0.95; ε = 0.01; γ = 3.0; ω = 1.0)
(J_1 = 0.0348; J_2 = 0.0074)
(Actual ____ ; Setpoint - - - -)

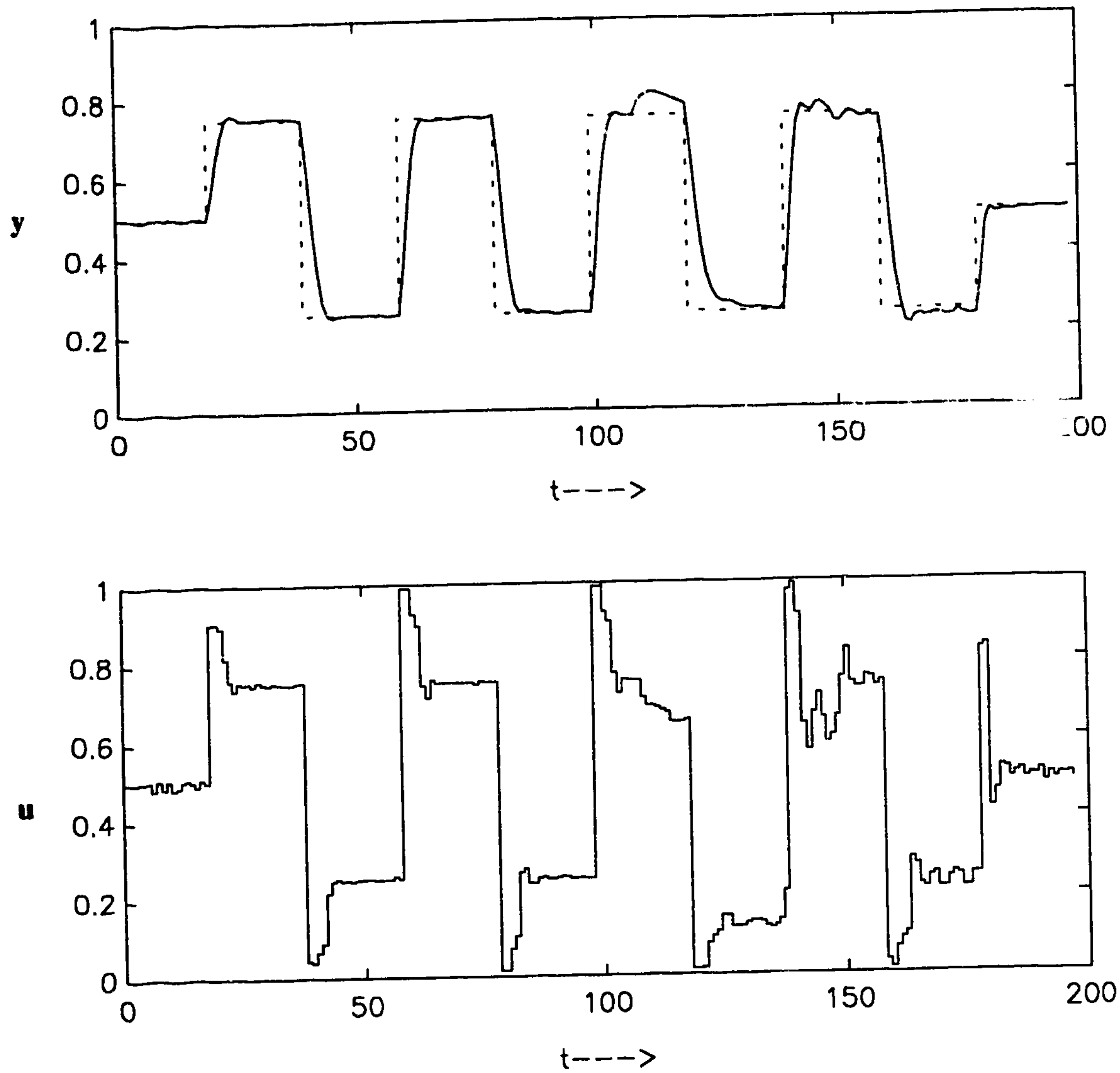


Figure 10.6: Control of Overdamped Process with *Max-min* Composition
($\alpha = 0.5$; $\beta = 0.5$; $\eta = 1.0$; $\varepsilon = 0.01$; $\gamma = 3.0$; $\omega = 1.0$)
($J_1 = 0.0354$; $J_2 = 0.0078$)
(Actual ____ ; Setpoint - - - -)

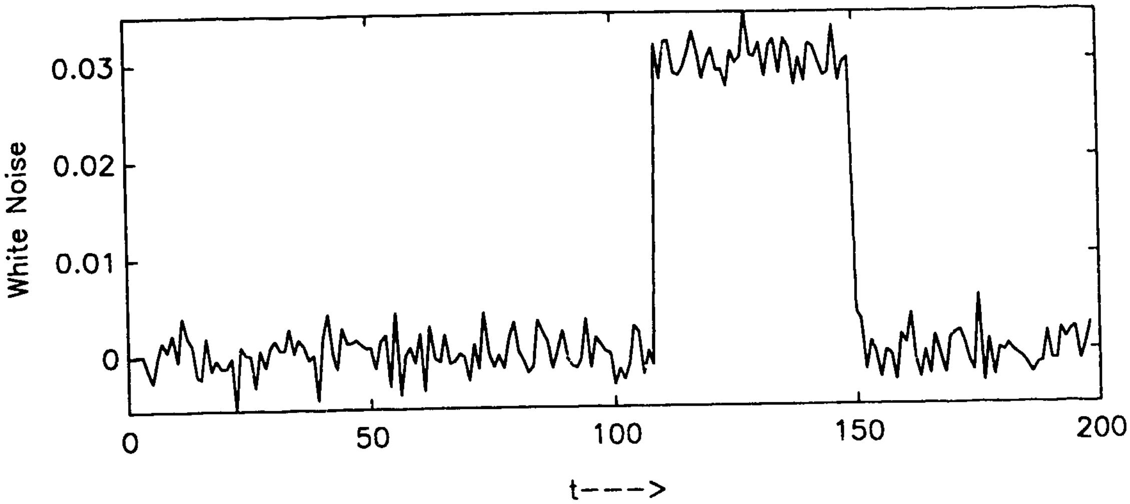


Figure 10.7: Noise for Overdamped Process
(White with change in mean)
(σ = 0.002; step = 0.03)

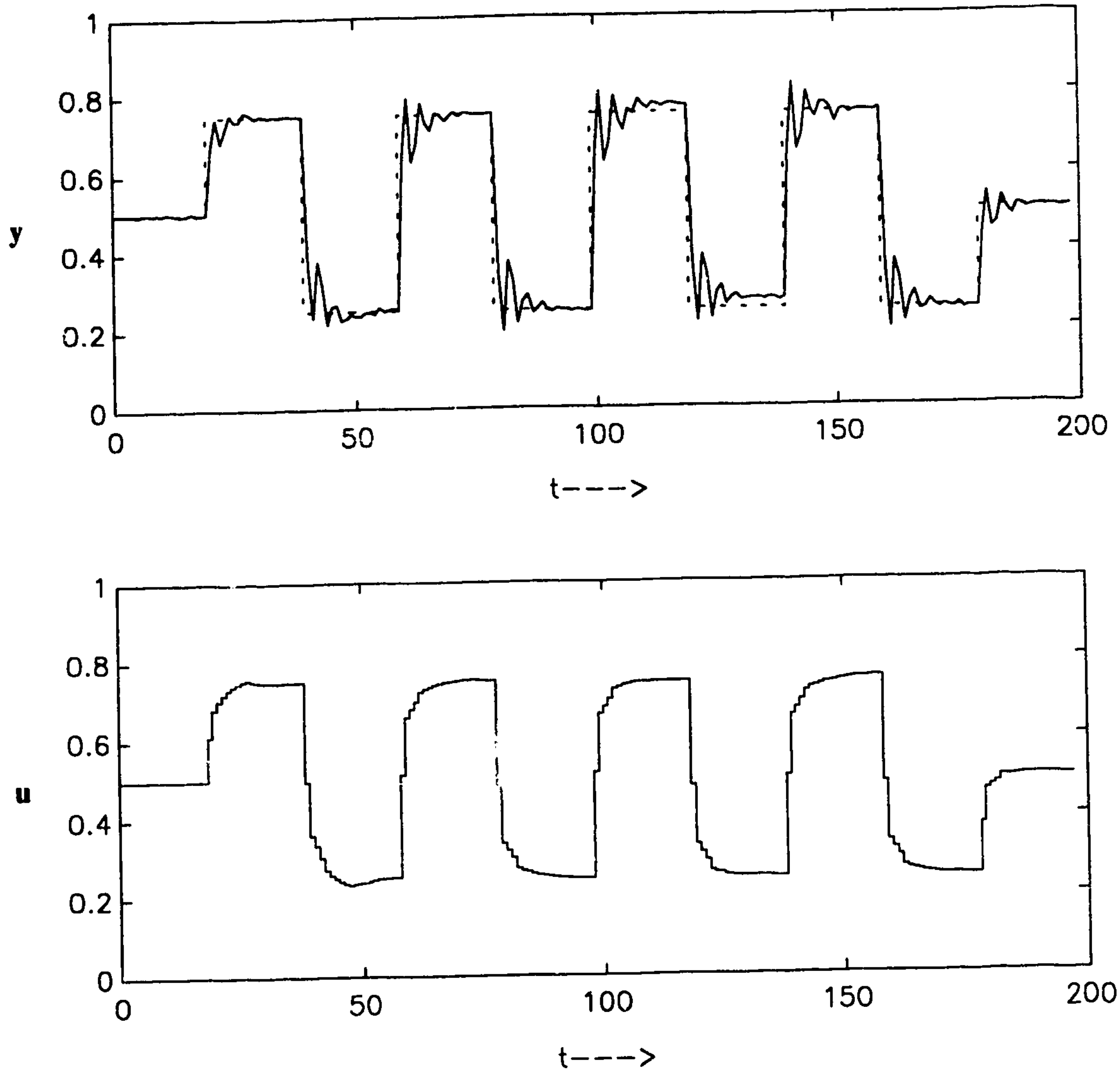


Figure 10.8: Control of Underdamped Process with *Max-product* Composition
($\alpha = 0.25$; $\beta = 0.25$; $\eta = 1.0$; $\varepsilon = 0.01$; $\gamma = 3.0$; $\omega = 1.0$)
($J_1 = 0.0226$; $J_2 = 0.0015$)
(Actual ____ ; Setpoint - - - -)

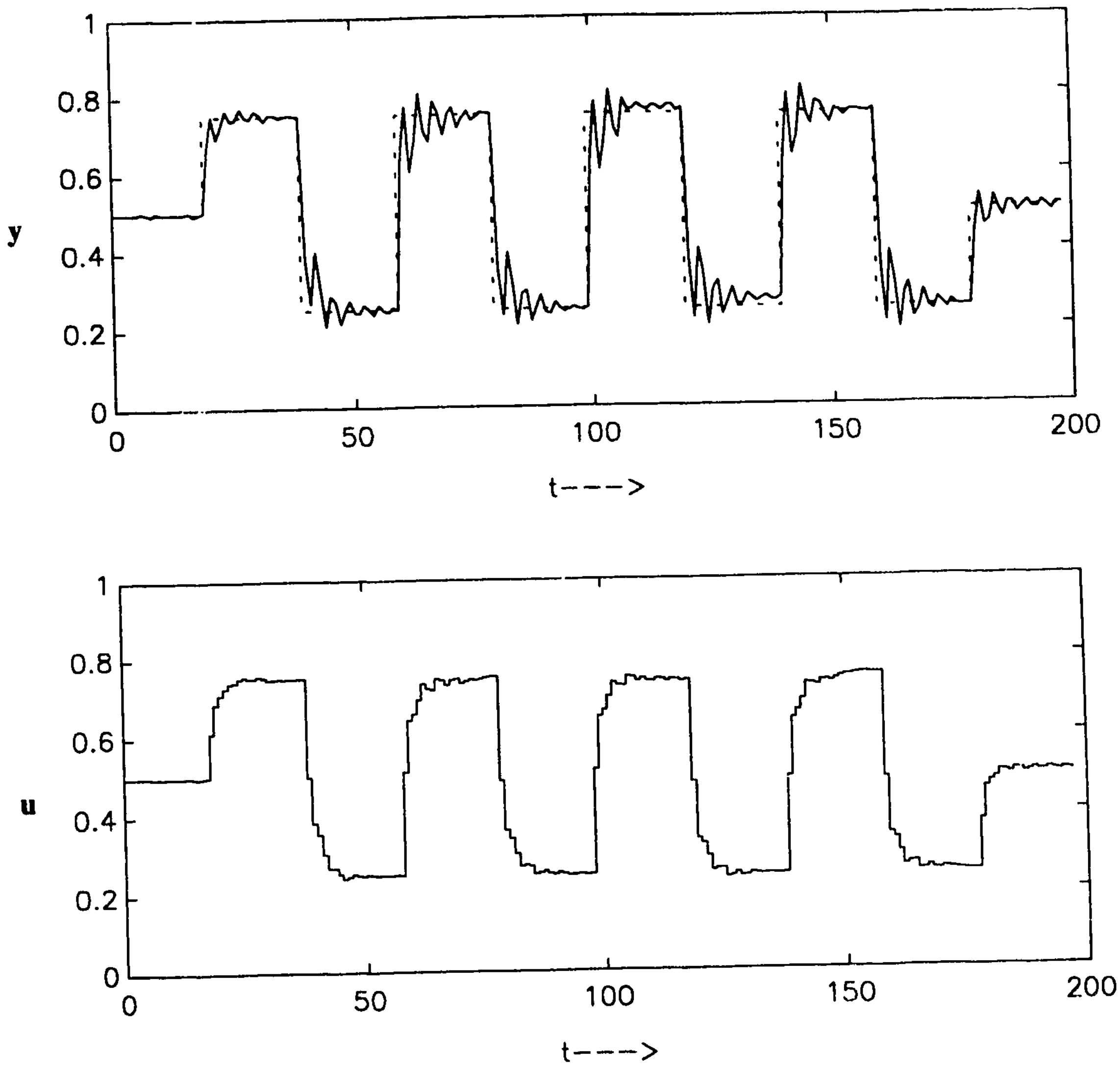


Figure 10.9: Control of Underdamped Process with *Max-min* Composition
($\alpha = 0.25$; $\beta = 0.25$; $\eta = 0.75$; $\varepsilon = 0.01$; $\gamma = 3.0$; $\omega = 1.0$)
($J_1 = 0.0260$; $J_2 = 0.0018$)
(Actual ____ ; Setpoint - - - -)

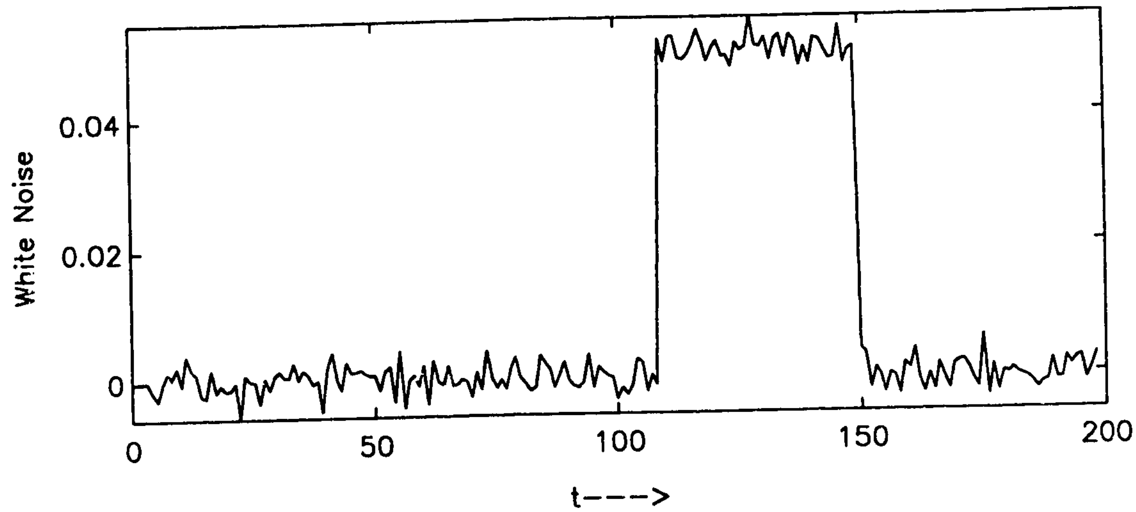


Figure 10.10: Noise for Underdamped Process
(White with change in mean)
(σ = 0.002; step = 0.05)

10.5.3 Non-linear Negative Gain Process

The strength of fuzzy logic controllers is that they can control processes which cause difficulties for conventional controllers. The highly non-linear inverse relation of the process model tested in this section is illustrated in Figure 10.11. This process was controlled without noise so that the accuracy of the controller could be demonstrated.

For the non-linear process model of equation (10.6 and 10.7) the results of the *max-product* controller are *better* than for the *max-min* controller, as shown in Figures 10.8 and 10.9. For the *max-product* composition, J_1 = 0.0069 and J_2 = 3.632×10^{-4} versus J_1 = 0.0079 and J_2 = 4.140× 10^{-4} for the *max-min* composition. The tuning for this test case was not aggressive as indicated by the lack of overshoot and a closed-loop process response slower than the open-loop response of Figure 10.3.

During steady state operation the *gain* model, $\boldsymbol{G}$, and *dynamic* model, $\boldsymbol{R}$, are required to calculate a process input that minimizes the process output error (i.e. $|y_{sp} - y|$). As shown during the steady state intervals in Figures 10.12 and 10.13, the calculated process input does not result in the *exact* process output required to track the setpoint. This is indicated by the offset between the process output and the setpoint. However, this discrepancy is to be expected since the model is *fuzzy* or *non-exact*. Yet even for this highly non-linear process, the fuzzy model was able to predict and control over the entire operating range, shown in Figure 10.11.

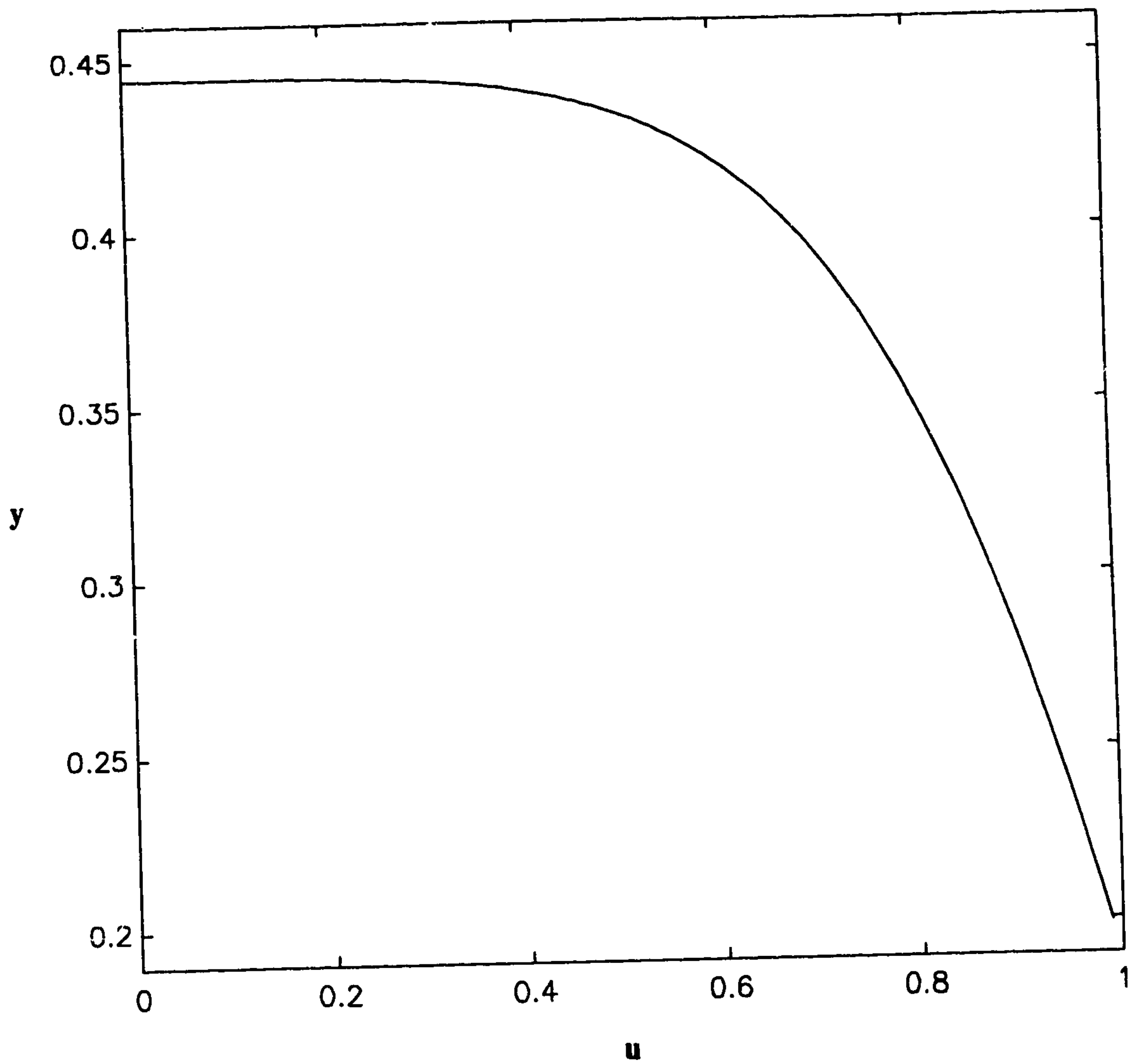


Figure 10.11: Non-linear Negative Gain Process Model

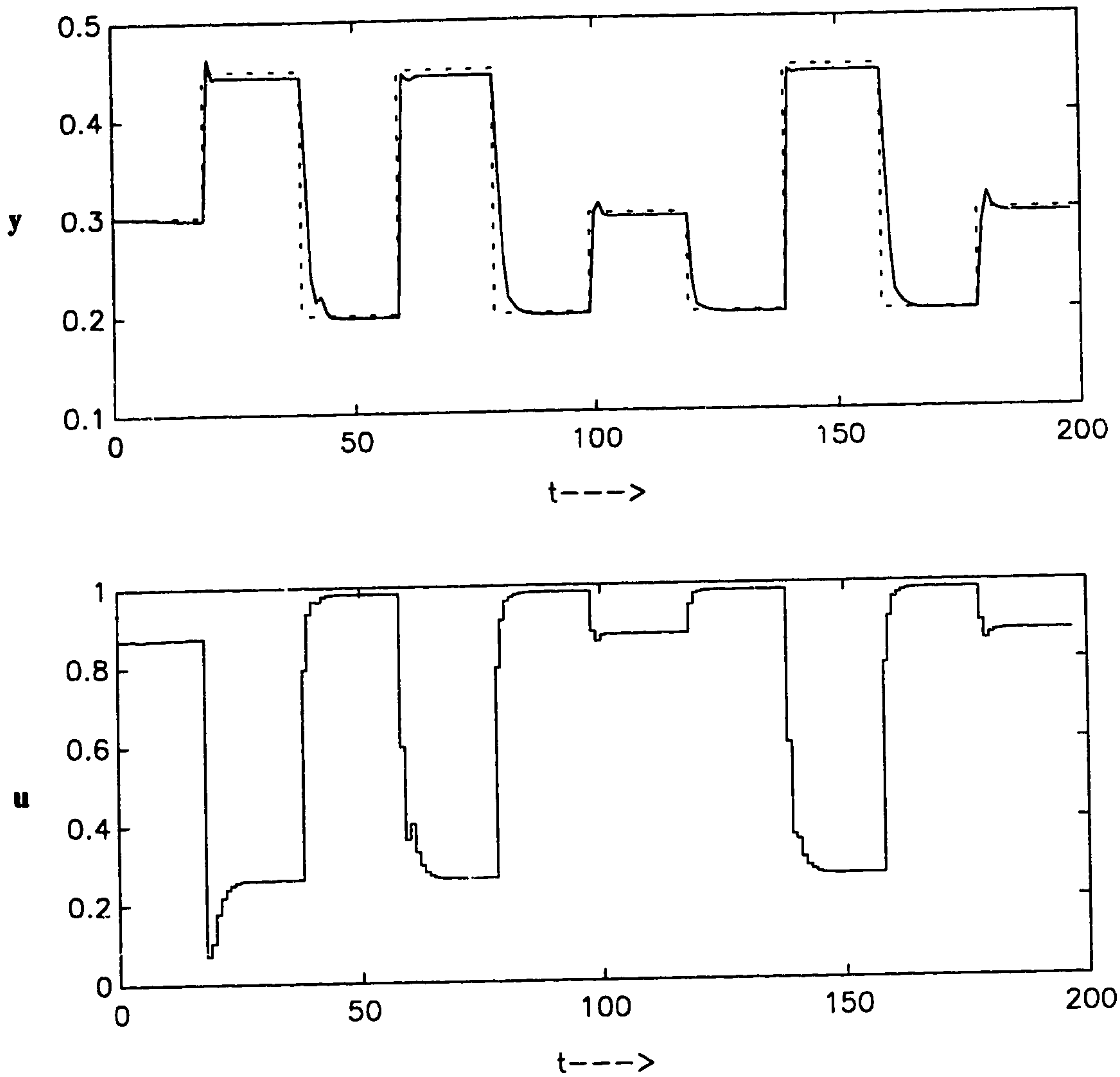


Figure 10.12: Control of Non-linear Negative Process with *Max-product* Composition
($\alpha = 0.45$; $\beta = 0.95$; $\eta = 0.9$; $\varepsilon = 0.01$; $\gamma = 3.0$; $\omega = 1.0$)
($J_1 = 0.0045$; $J_2 = 3.632\times10^{-4}$)
(Actual ____ ; Setpoint - - - -)

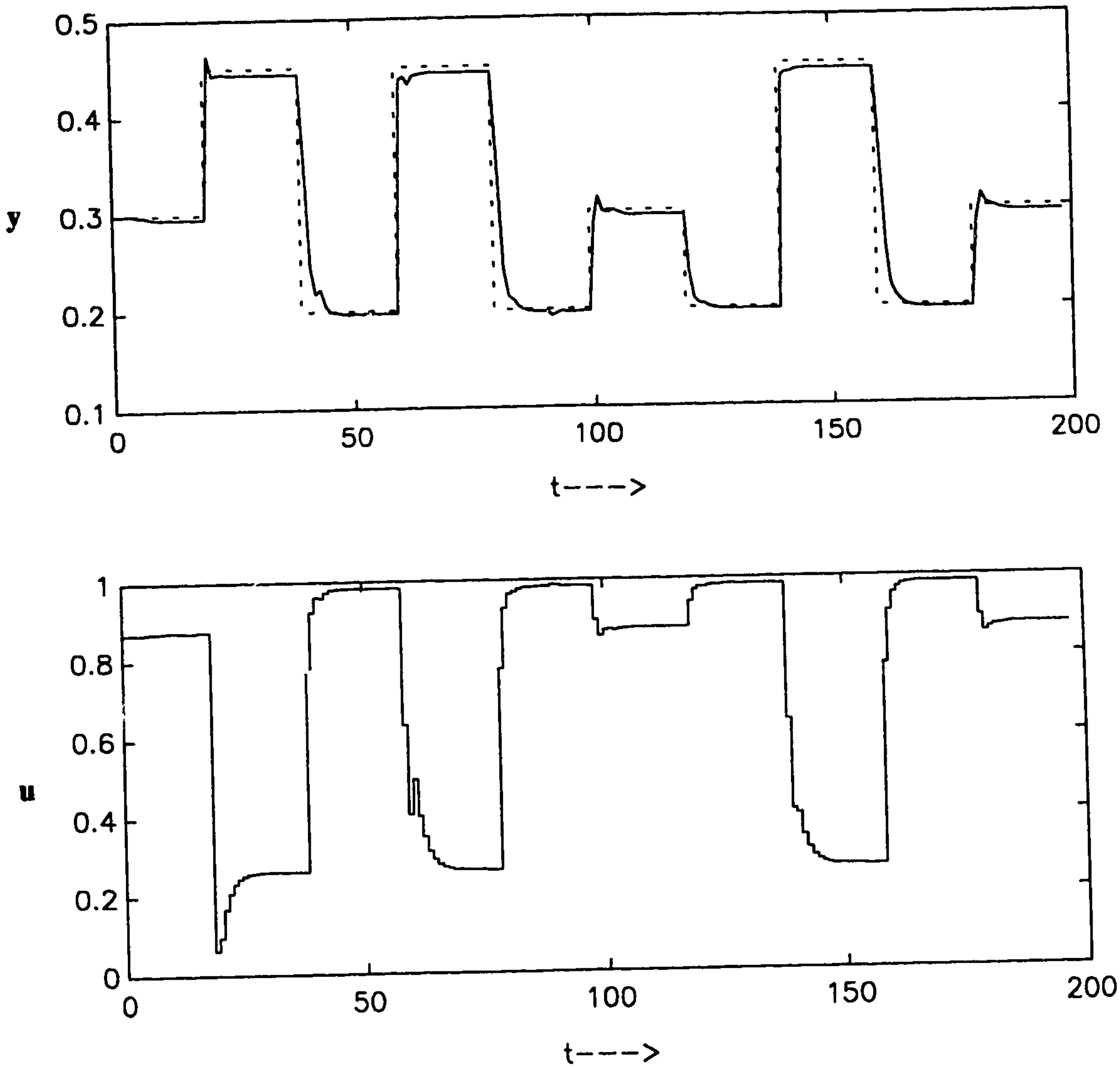


Figure 10.13: Control of Non-linear Negative Gain Process with *Max-min* Composition
($\alpha = 0.4$; $\beta = 0.95$; $\eta = 0.9$; $\varepsilon = 0.01$; $\gamma = 3.0$; $\omega = 1.0$)
($J_1 = 0.0079$; $J_2 = 4.140\times10^{-4}$)
(Actual ____ ; Setpoint - - - -)

10.5.4 Shell Process

For the Shell process model [Cott, 1995(b)] the process coloured noise, described in the paper by Cott [1995(b)] was not used as the distortion from the noise did not allow for adequate assessment of the capability of the controller. Instead white noise was added with a standard deviation, $\sigma = 1.0$. A disturbance step of 5 output units is added at simulation time instance 75 and removed at instance 325. The disturbance profile is shown in Figure 10.14.

For the Shell process disturbance model, results based on the performance indices again show that the *max-product* composition is *better* than the *max-min* composition, with $J_1 = 25.445$ and $J_2 = 1.289\times10^3$ for the *max-product* composition and $J_1 = 26.979$ and $J_2 = 1.450\times10^3$ for the *max-min* composition. The controller performance for the *max-min* composition is more oscillatory compared to the *max-product* composition, as shown in Figures 10.15 and 10.16. The *max-min* controller also has more difficulty controlling the step disturbance (i.e. sampling instances 75-325).

Although the process output does not track the setpoint *exactly*, for the two approaches using the Shell Data, the closed-loop response time has been *significantly reduced* versus the open-loop response of Figure 10.4. Offset from setpoint is due to the step disturbance and slow learning of the changed operating conditions on the part of the identification algorithm, as well as reflecting *inaccuracies* in the gain matrix, **G**.

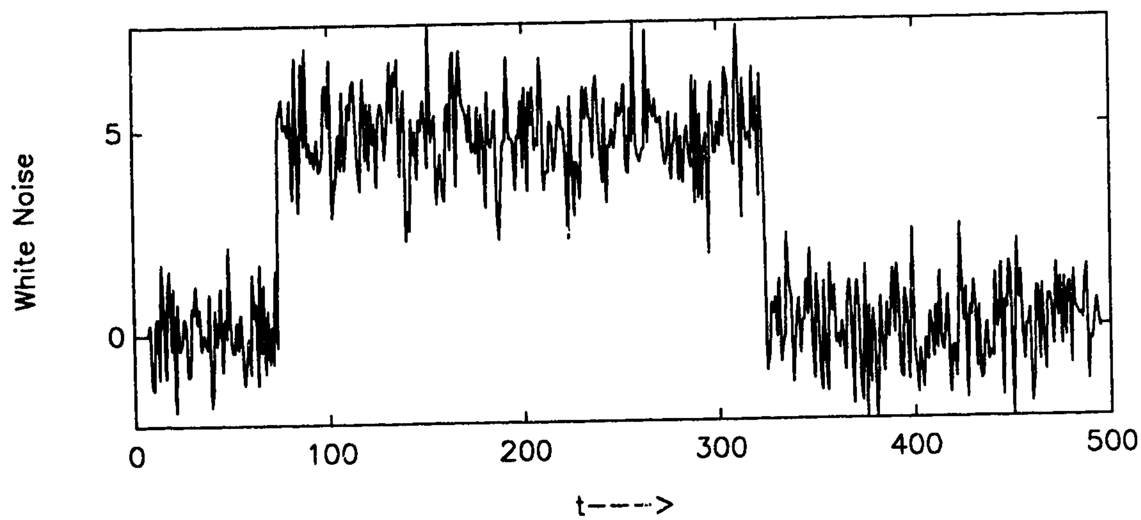


Figure 10.14: Noise for Shell Process (White with change in mean) (σ = 1.0; step = 5.0)

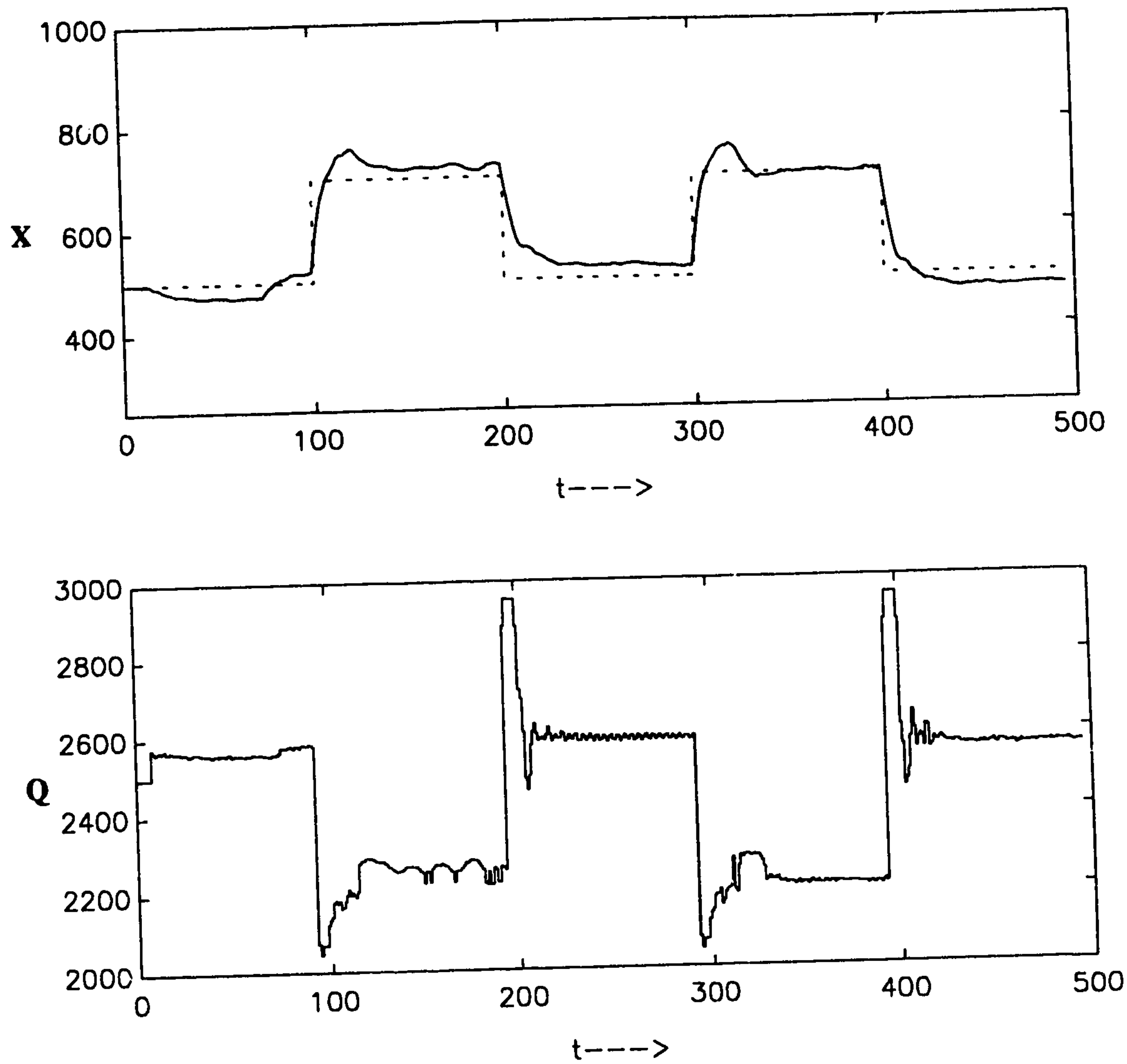
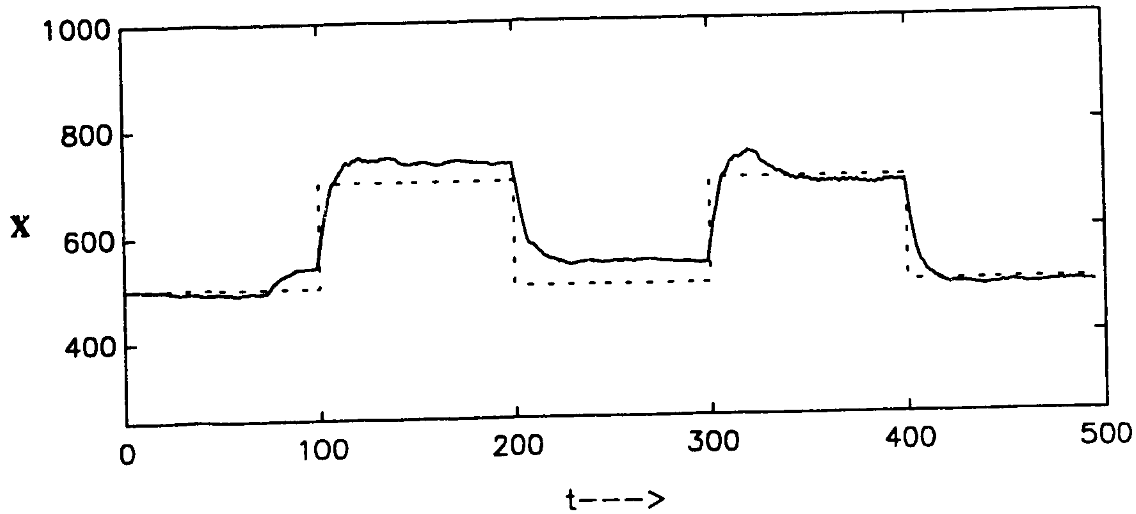


Figure 10.15: Control of Shell Process with *Max-product* Composition
($\alpha = 0.5$; $\beta = 2.0$; $\eta = 0.3$; $\varepsilon = 0.01$; $\gamma = 3.0$; $\omega = 1.0$)
($J_1 = 25.445$; $J_2 = 1.289\times10^3$)
(Actual ____ ; Setpoint - - - -)



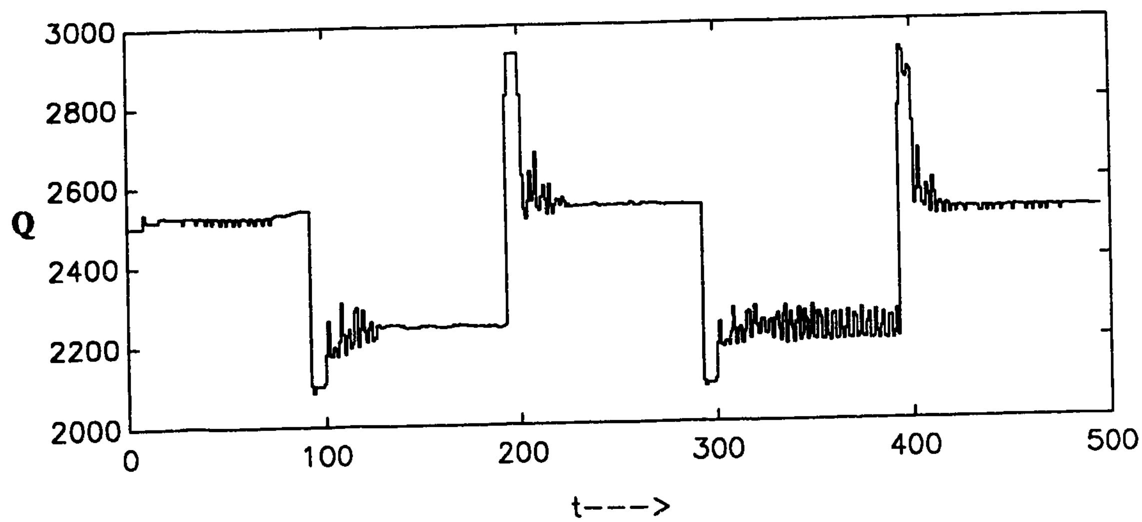


Figure 10.16: Control of Shell Process with *Max-min* Composition
($\alpha = 0.75$; $\beta = 2.0$; $\eta = 0.25$; $\varepsilon = 0.01$; $\gamma = 3.0$; $\omega = 1.0$)
($J_1 = 26.979$; $J_2 = 1.450\times10^3$)
(Actual ____ ; Setpoint - - - -)

10.6 *Fuzzy* Control vs. Conventional PI Control

As mentioned previously, fuzzy logic control is ideally suited to those situations when conventional control methodologies fail. An example of this would be control of a highly non-linear process. This section compares the control results of a non-linear process using the fuzzy logic controller developed in this thesis versus a PI controller.

The highly non-linear process chosen for this comparison is such that there are 3 areas of gain, low, medium and high, as shown in Figure 10.17. The actual process model is defined by:

$$y(k) = \sigma(a_1 y(k-1) + b_1 \sinh(0.8 \cdot u(k-1))^4) \quad (10.16)$$

where $a_1 = 0.01$

$b_1 = 10.0$

The sigmoid function is as defined as:

$$\sigma(x) = \frac{2}{0.995 \cdot (1 + \exp(-x))} \quad (10.17)$$

The PI controller configuration used for these studies is:

$$\Delta u = K_c\left[(e_n - e_{n-1}) + \frac{\Delta t}{\tau_i} e_n\right] \quad (10.18)$$

where $\Delta t \equiv$ one sampling interval and K_c and τ_i are the proportional and integral tuning constants respectively.

For this study, the PI controller was initially tuned for the ***medium gain*** region of the given process and the controller was then required to provide servo control over the entire process range. Figure 10.18 shows the results of this application. Clearly from Figure 10.18(c), *good* control is evident in the *medium gain* region. Figure 10.18(b) illustrates the slower yet stable control in the *low gain* region. However, control in the ***high gain*** region is unstable, as shown in Figure 10.19(a). Thus PI controllers produce poorer and/or unstable results in regions outside the tuned range.

The obvious direction to take, based on the results of Figure 10.18, is to reduce the controller gain. Thus the PI controller as retuned for the *high gain* region and servo control was again tested over the entire process range. Figure 10.19 shows the results of the retuning. The PI controller produces *good* results in the high gain region, however, setpoint tracking is slow in both the *medium* and *low* gain regions.

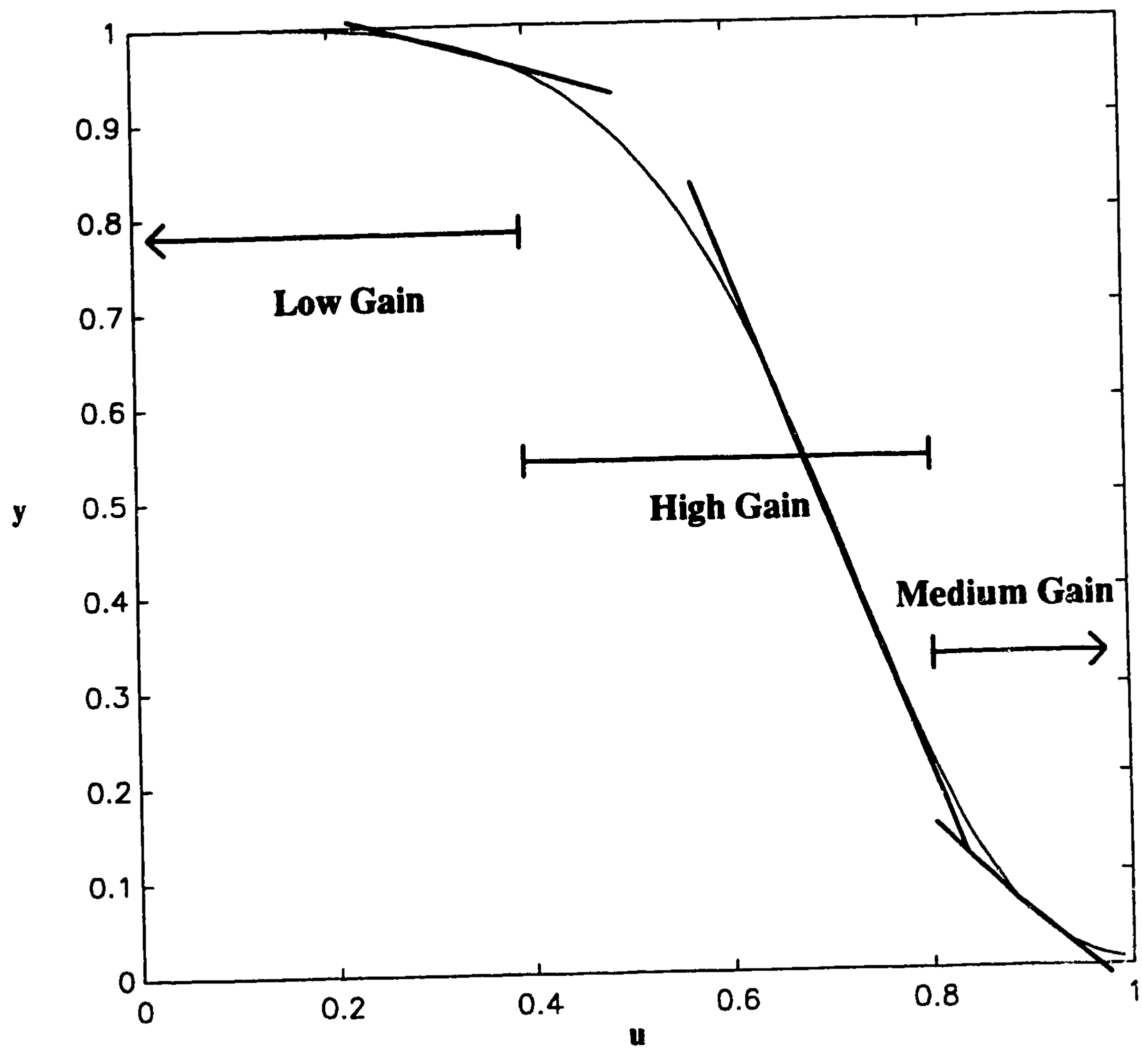


Figure 10.17: Non-linear Process Featuring Low, Medium and High Gain

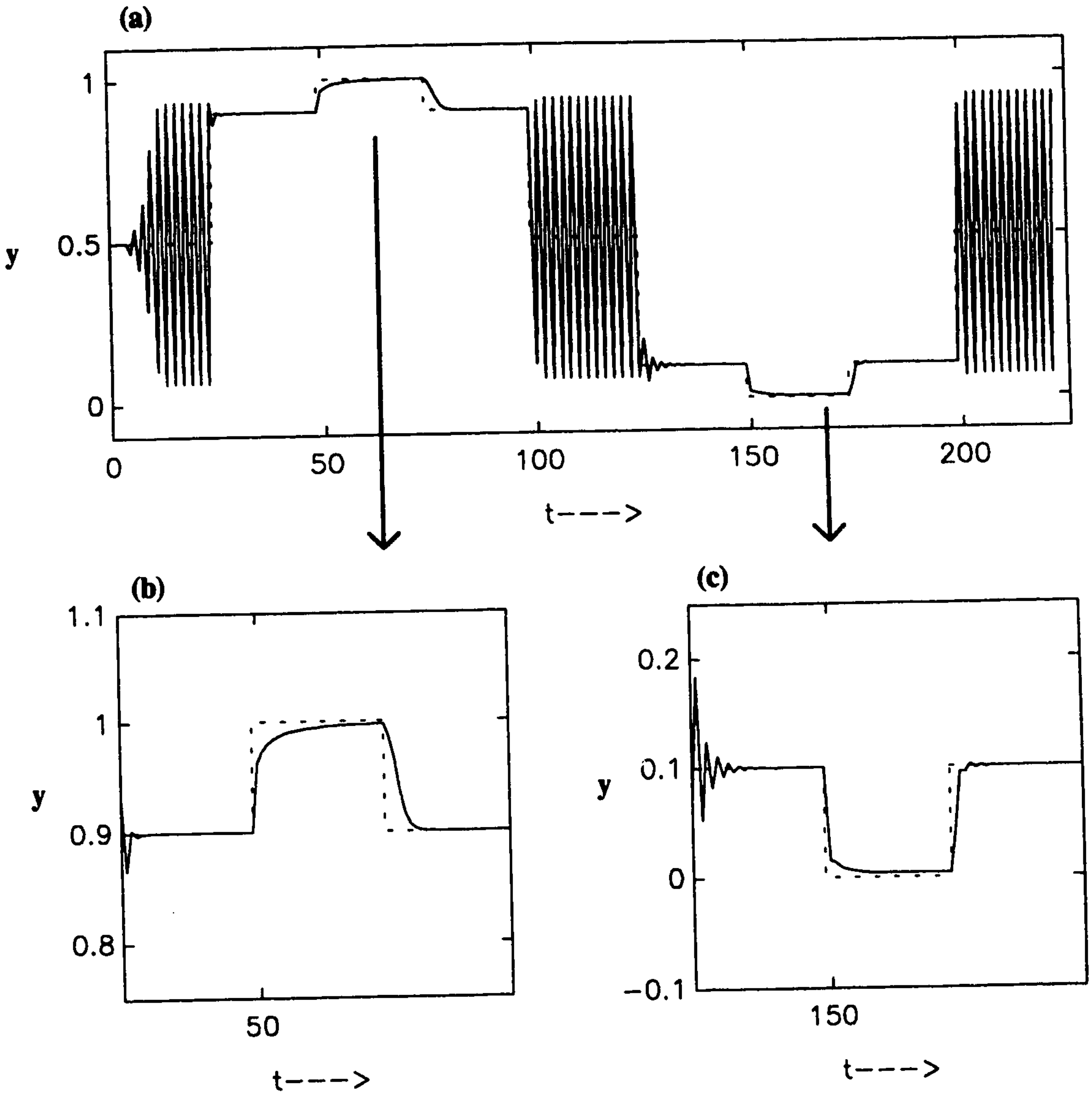


Figure 10.18: PI Control Tuned for Low Gain Region
(a) Output Profile; (b) Magnification of Low Gain; (c) Magnification of Medium Gain
(K_c = -0.0875; τ_i = 0.1)
(J_1 = 0.1332; J_2 = 0.0544)
(Actual ____ ; Setpoint - - - -)

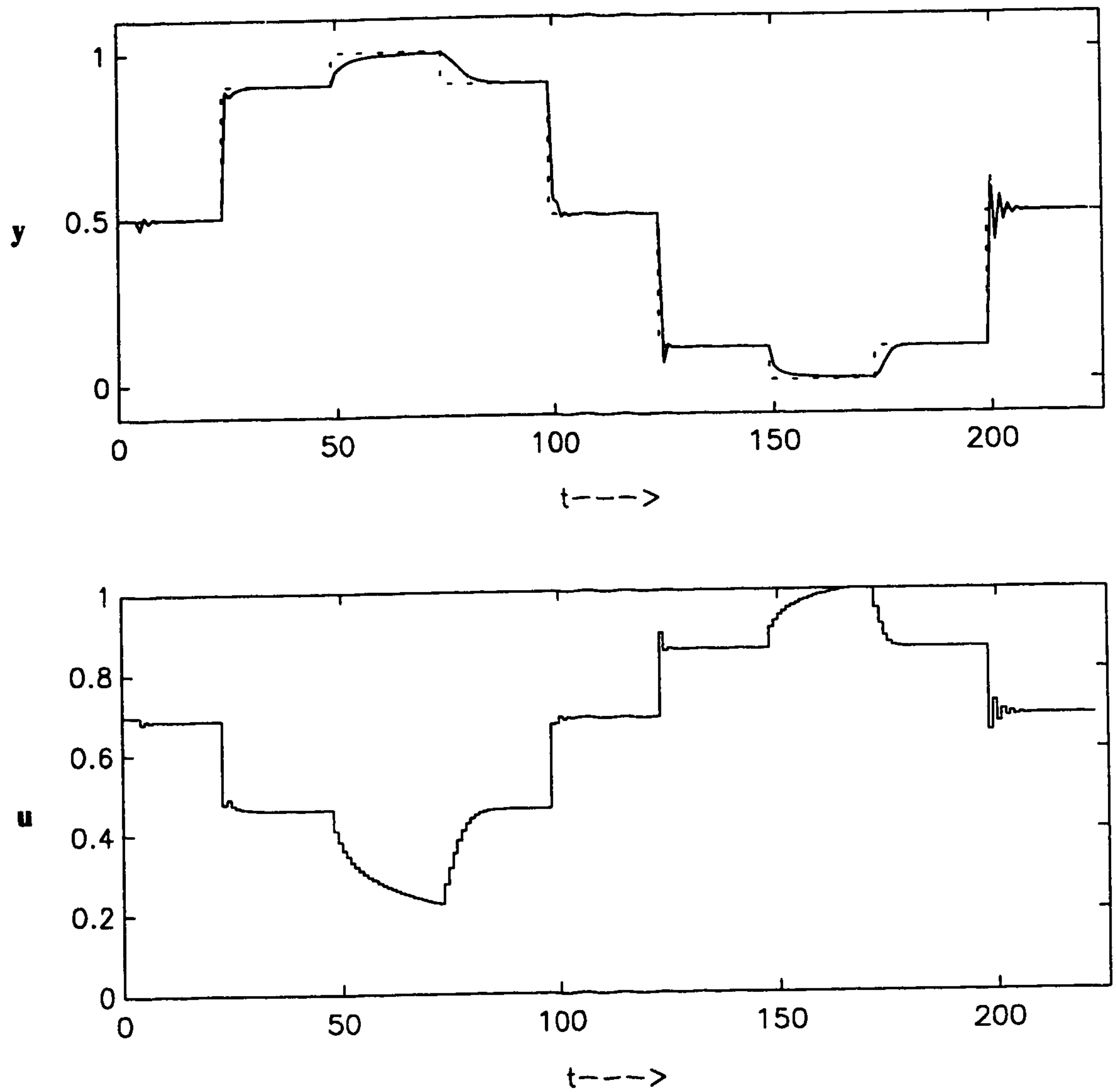
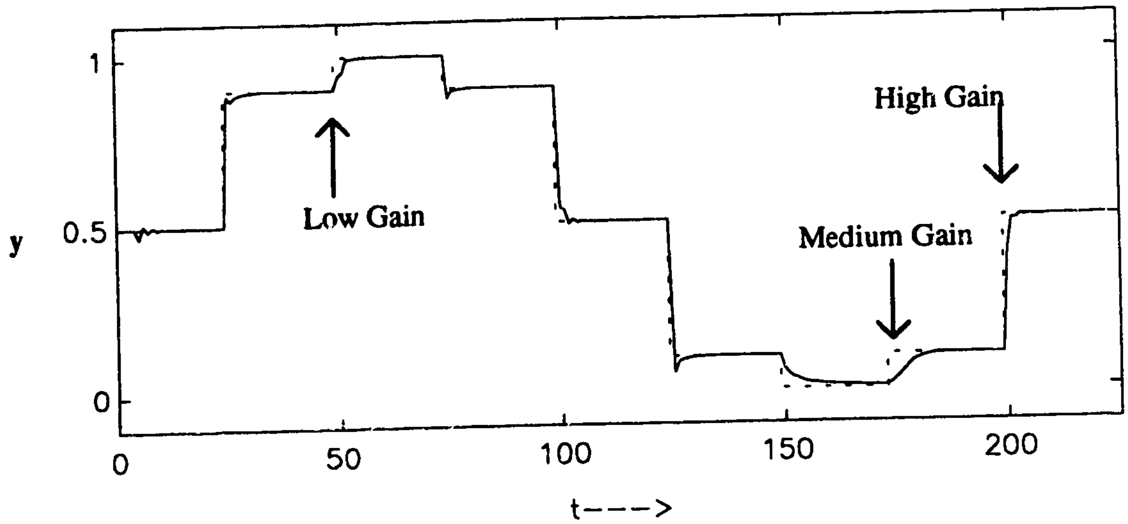


Figure 10.19: PI Control Tuned for High Gain Region
(K_c = -0.0472; τ_i = 0.1)
(J_1 = 0.0170; J_2 = 0.0034)
(Actual ____ ; Setpoint - - - -)

Gain scheduling of the PI controller was then applied in an attempt to obtain *good* control response in all regions. As shown in Figure 10.20, response has improved in the *low gain* region and the change in gain criterion is quite evident from the input response figure. However, response is *poor* in the *medium gain* region due to the detuning required to improved control in the ***high gain*** region. This improved control is the elimination of the oscillatory response at the 200 sampling instance. Overall control results of PI gain scheduling have improved, with $J_1 = 0.0150$, over the results of tuning for only the ***high gain*** region.

The proposed fuzzy logic controller was then applied to the same non-linear process problem. Tuning of the fuzzy logic controller was based on *good* overall control, with minimum overshoot. Figure 10.21 illustrates the results of the fuzzy logic controller. *Clearly good overall control is obtained over the entire process range*. Comparing the J_1 results of the fuzzy logic controller, $J_1 = 0.0131$, and the *best* PI controller, tuned for with gain scheduling, $J_1 = 0.0150$, show that the fuzzy logic controller operates *better* over the entire process range, based on this minimum distance criteria.



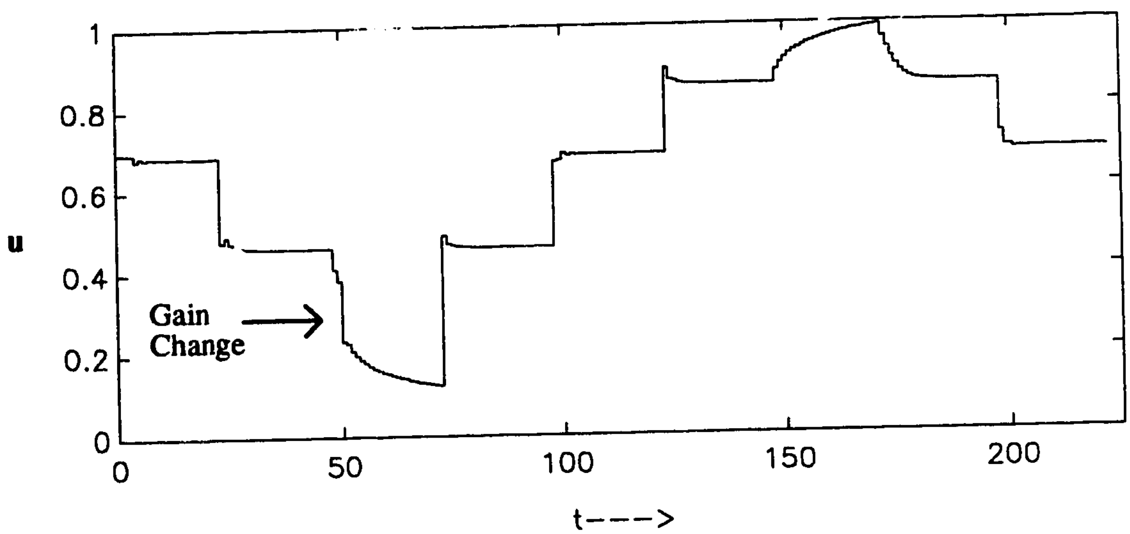


Figure 10.20: PI Control with Gain Scheduling
(K_c(Low) = -0.3416; K_c(High) = -0.0098; K_c(Medium) = -0.0294; τ_i = 0.1)
(J_1 = 0.0150; J_2 = 0.0033)
(Actual ____ ; Setpoint - - - -)

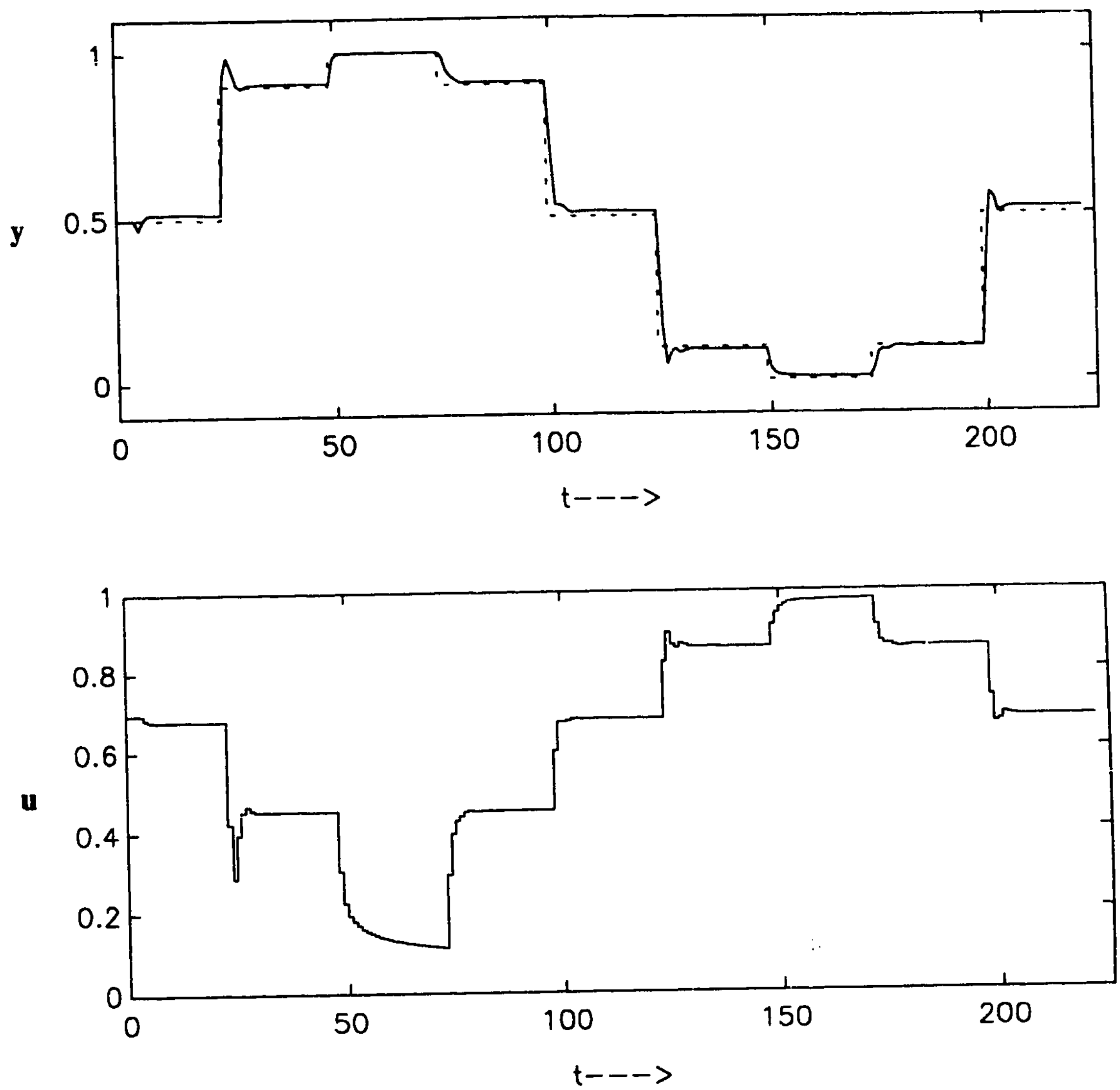


Figure 10.21: Fuzzy Control with *Max-Product* Composition
($\alpha = 0.45$; $\beta = 0.25$; $\eta = 0.95$; $\epsilon = 0.01$; $\gamma = 3.0$; $\omega = 1.0$)
($J_1 = 0.0131$; $J_2 = 0.0005$)
(Actual ____ ; Setpoint - - - -)

10.7 Summary

With conventional control it is assumed that a process is represented by the linear model G_m in a small region about the operating point. Basically G_m is the same for all operating regions. With *fuzzy control*, the input-output relationship is divided into a user chosen number of regions that define a functional form and the relational model provides the *best fit in each region*. With judicious choice of the *fuzzy* membership functions, the transition from region to region is smooth. Thus there is continuity in the modeling and predictive capabilities of the *fuzzy* model.

Chapter 10 shows that the fuzzy controller design in this thesis produces *good* control for a variety of realistic processes (i.e. underdamped process, overdamped process, negative gain process, non-linear process, large system delay, large step disturbance). Additionally, the *max-product* composition provided consistently *better* control than the *max-min* composition, based on the minimum distance criterion.

For all the test cases the absolute error, J_1, was less than 5% of the process output. Thus the fuzzy predictive control algorithm has demonstrated that it is applicable to process applications. The test simulation with the Shell data also demonstrated that the algorithm can handle a large delay (**d** = 6 sampling instances). The non-linear test case comparing the *fuzzy controller* to the PI controller clearly demonstrates the ability of the *fuzzy controller* to provided *good* control over the entire operation range of a highly non-linear process.

REFERENCES

[1] Arifin, N., Wang, L., Goberdhansingh, E., Cluett, W.R., "Identification of the Shell Distillation Column using the Frequency Sampling Filter Model", *Journal of Process Control*, Vol. 5, pp. 71-76, 1995.

[2] Bailey, J.K., "Process Identification using Finite Impulse Response Models", *Journal of Process Control*, Vol. 5, pp. 77-84, 1995.

[3] Banerjee, P., Shah, S.L., Niu, S., Fisher, D.G., "Identification of Dynamic Models for the Shell Benchmark Problem", *Journal of Process Control*, Vol. 5, pp. 85-87, 1995.

[4] Chan, M.D., "Identification of a Simulated Shell Distillation column using Time Series Analysis Methods", *Journal of Process Control*, Vol. 5, pp. 99-103, 1995.

[5] Cott, B.J., "Introduction to the Process Identification Workshop at the 1992 Canadian Chemical Engineering Conference", *Journal of Process Control*, Vol. 5, pp. 67-69, 1995(a).

[6] Cott, B.J., "Summary of the Process Identification Workshop at the 1992 Canadian Chemical Engineering Conference", *Journal of Process Control*, Vol. 5, pp. 109-113, 1995(b).

[7] Hernández, E., Arkun, Y., "Control of Nonlinear Systems Using Polynomial ARMA Models", *AIChE Journal*, Vol. 39, pp. 446-460, March 1993.

[8] Pedrycz, W., "Numerical and Applicational Aspects of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 11, pp. 1-18, 1983.

[9] Xu, C.-W., Lu, Y.-Z., "Fuzzy Model Identification and Self-Learning for Dynamic Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol., 17, pp. 683-689, 1987.

[10] Zadeh, L.A., "Outline of a New Approach to the Analysis of Complex Systems and Decision Process", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-3, No. 1, pp. 28-44, 1973.

CHAPTER 11 CONCLUSIONS

"The universe is strange. Not only is the universe stranger the we think, it may be stranger than we can think. One result is that the physicist, who must deal with the real world, has at least three logics: one for ordinary state variables, classical mechanics, one for the microscope, quantum mechanics, and one for the macroscopic, relativity theory.

Thus consistency gives way to utility. It is possible that one theory would handle all three cases, but it would probably be too complex to use.

We definitely need more logics to deal with uncertainty."

[Bellman and Zadeh, 1977]

11.1 Thesis Overview

Fuzzy logic is a powerful technique for control applications that are not deterministic enough to be handled by traditional model-based control systems. It can address ill-defined systems with vague data specifications. In many instances complex processes are controlled manually due to the inappropriateness of traditional control methodologies. Fuzzy logic is not meant to replace classical control theory. Its purpose, as with *classical mechanics*, *quantum mechanics* and *relativity theory*, is to provided solutions in those areas where classical control theory breaks down.

This thesis applies *fuzzy logic* to the construction of a *Self-Learning, Predictive, Fuzzy-Logic Controller*. The controller developed in this thesis is based on a relational matrix formulation which has the advantages of being able to learn directly from experimental input/output data. The matrix format of this design clearly indicates how output depends on input and the numerical format permits quantitative analysis.

The *fuzzy controller* design is based on a *max-product* composition which has been shown to be superior to the *max-min* composition in many instances. The thesis analyzes the stability of this fuzzy composition and presents a method of stabilizing this composition in control situations.

A new on-line identification algorithm is proposed in this thesis. Since *max-product* composition has not been extensively used, there is little literature on effective identification algorithms which feature this composition. Thus this thesis compared several identification techniques using both *max-min* and *max-product* composition to determine the "*best*" from a *least squares* perceptive.

Many of the rule-based fuzzy controller designs in the literature are based on a *PI* structure, so there is little or no predictive ability. The controller algorithm presented in this thesis is model-based and the predictive capabilities of this algorithm make it more suitable for controlling systems with dead time.

The resulting controller design parallels the design of a conventional model based predictive controller but the application is in areas where conventional control fails.

11.2 Contributions

One objective of this thesis was to review and/or expand the *fuzzy logic* theory, starting with the basics, in order to provided a solid basis for the development of a *self-learning, predictive, fuzzy logic controller*. Chapters 2 through 8 of this thesis progress logically and comprehensively through the theory required for the final controller development. The chapter by chapter contributions of this thesis to the area *Fuzzy Logic Theory* are summarized in Section 11.2.1. Section 11.2.2 then summaries the main contributions in the area of *Fuzzy Process Identification and Control*.

11.2.1 Contributions in the Area of Fuzzy Theory

Chapter 2 presents a literature review of fuzzy rule-based systems theory as a starting point for the development of the relational-based control system presented in this thesis. The literature review consolidates the rule-based development theory from several sources into a format consistent with the development of relational-based theory, which is the focus of this thesis. The aim of the literature review of rule-based fuzzy theory and relational-based fuzzy theory, presented in Chapter 3, is to emphasize the fact that rule-based information can also be presented in a relational matrix format.

Chapter 4 references several sources of research confirming the superiority of the *max-product* composition, as well as presenting the reasons why this superiority exists. The literature review of Chapter 4 compares, using several examples, the physical interpretation and ability of the *max-min* and *max-product* compositions.

Chapter 5 begins with a literature review that consolidates solutions to the various inverse problems presented in the literature.

(1) "Given the fuzzy relations $\boldsymbol{R}$ and $\boldsymbol{b}$, find all fuzzy sets $\boldsymbol{a}$ such that $a \circ \boldsymbol{R} = \boldsymbol{b}$".

(2) "Given the fuzzy relations $\boldsymbol{R}$ and $\boldsymbol{T}$, find all fuzzy relations $\boldsymbol{Q}$ such that $\boldsymbol{Q} \circ \boldsymbol{R} = \boldsymbol{T}$".

(3) "Given the fuzzy sets $\boldsymbol{a}$ and $\boldsymbol{b}$, find the fuzzy relation $\boldsymbol{R}$ such that $\boldsymbol{a} \circ \boldsymbol{R} = \boldsymbol{b}$".

(4) "Given the fuzzy relations $\boldsymbol{Q}$ and $\boldsymbol{T}$, find the fuzzy relation $\boldsymbol{R}$ such that $\boldsymbol{Q} \circ \boldsymbol{R} = \boldsymbol{T}$".

A complete program, written in MATLAB®, to determine fuzzy inverses using either the *max-min* or the *max-product* composition for all the problem statements is available and a listing is provided in Appendix 1.

A table is provided in Chapter 5 which details the literature source for all the problem definitions for *max-min*, *max-product* and *max-t-norm*. Additionally, a complete solution guide for these problems for the *max-product* composition is provided in Appendix 2. The consolidated literature review for fuzzy inverse problems, available in Chapter 5, provides an important reference itself to the abundant material available on the subject.

Chapter 5 also presents the complete solution of the cartesian product of fuzzy sets for the *max-min* composition. This work was published in *Fuzzy Sets and Systems*, [Bourke *et al.* 1994]. This paper completes the work of two other papers, by different authors, and has potential for control systems from a on-line identification perspective. The complete solution theory of the cartesian product is then extended to the *max-product* composition for possible on-line identification applications.

Chapter 6 is important from a control perspective as it consolidates *fuzzy logic* design stability theory and demonstrates the agreement between *fuzzy control* and *conventional classical control*. Two important results provided in Chapter 6 are:

(1) Stability analysis and convergence properties for relational matrices combined with the *max-product* operator [Bourke *et al.*, 1995(b)]

(2) Eigen Fuzzy Sets analysis for *max-product* compositions [Bourke *et al.*, 1995(b)]

The stability analysis and convergence property results of this work are critical for relational matrices combined with the *max-product* composition because of the possibility that these relational matrices may converge to a [0] or null matrix. Therefore, before the development of a control policy with the *max-product* composition, the conditions for the existence of the unstable solution matrices must be determined and addressed.

The ability of a system to obtain and maintain a setpoint under a control policy is critical, and the knowledge of the conditions under which deterioration may result is crucial. Eigen fuzzy set analysis reveals the ability of a relational matrix, combined by successive *max-product* compositions, to maintain a setpoint under a control scenario. As well, a method to overcome poor or deteriorating response for those matrices that do not meet the criteria of stability with successive composition is provided for both *max-min* and *max-product* composition.

The literature review in Chapter 7 involves the evaluation, implementation and comparison of several identification algorithms [Pedrycz, 1984; DiNola *et al.*, 1984; Xu *et al.*, 1987; Shaw *et al.*, 1992; Chen *et al.*, 1994] using the same fuzzification and defuzzification methods, the same reference fuzzy set basis and the Box-Jenkins [1970] gas furnace data [Bourke *et al.*, 1995(c)]. This review is important in that it confirms the validity and ranks the relative ability of each of the algorithms tested.

The literature search for Chapter 8 provides a brief review of fuzzy analytical derivative theory which confirms the existence of a differential for a *fuzzy relational matrix* and validates this theory for application in the fuzzy domain.

Several optimization algorithms were reviewed in Chapter 8 [Pedrycz, 1993; Ikoma *et al.*, 1993; Valente de Oliveira, 1994]. The algorithm for neural learning [Pedrycz, 1991; Valente de Oliveira, 1993] was extended to the fuzzy cause problem. The probabilistic descent algorithm [Ikoma *et al.*, 1993] was extended to both the fuzzy cause problem and the *max-product* domain. The suitability of these optimization algorithms for identification were compared using the same fuzzification and defuzzification methods, the same reference fuzzy sets and the Box-Jenkins

[1970] gas furnace data, as the algorithms compared in Chapter 7 [Bourke *et al.*, 1995(c)]. The identification results showed that some non-optimized algorithms of Chapter 7 are capable of providing similar results with fewer calculations and less tuning.

11.2.2 Contributions in the Area of Fuzzy Process Identification and Control

Based on the identification techniques described by DiNola *et al.* [1984] and Shaw *et al.* [1992] a new identification algorithm was developed. The new algorithm uses the averaging technique, as described by Shaw *et al.* [1992], to determine the overall relation matrix $\boldsymbol{R}$ from the individual inverses calculated from a series of input-output data. The predictive results from this new algorithm are *better* than the other identification algorithms tested in Chapter 7.

The new identification algorithm developed in this thesis is based on a batch learning technique. However, this new technique can be applied on-line, as well as the algorithm by Shaw *et al.* [1992], by using a matrix resetting mechanism developed in this thesis. The resetting technique maintains model completeness while increasing the speed at which learning can be performed, both important features of fuzzy identification systems.

On-line identification requires an algorithm which can be completed within the sampling interval of the given process. Determination of relational matrices through an estimation procedure assumes that the results are comparable to an exact procedure. The estimation theory presented by Baboshin *et al.* [1990] is analyzed in detail and extended to the *non-normal* case. The final results of this work is the ordering of the variously identified relational matrices which are independent of the data used and are therefore representative of the capability of the estimation algorithms used.

The minimization of the control objective function or criterion is considered from an identification perspective and a causal perspective. For the work with identification it was necessary to extend the necessary and sufficient conditions from a solution to the single input-single output identification problem [Pedrycz, 1988 (*max-min*); Pedrycz, 1991 (*max-t-norm*)] to the single input-single output fuzzy cause problem for both *max-min* and *max-product*. These results, particularly for the *max-product* composition, demonstrated the need for numerical solution techniques.

A key result from a control system point of view is that the minimization of fuzzy criteria Q_1 or Q_2 does not imply a minimum J_1 or J_2, which are defined for the discrete domain. This knowledge is particularly important for those discrete control systems that are handled in the fuzzy domain due to lack of an adequate deterministic model.

Chapter 9 presents a new predictive fuzzy controller design based on relational matrices using a *max-product* composition. This work is important with respect to fuzzy control theory in that the controller is model-based and not an relational matrix application of *PI* control. Development of the model-based controller closely parallels the development of discrete model-based control theory.

Chapter 10 shows that the proposed controller design produces *good* control, from a fuzzy perspective, for a variety of control scenarios (i.e. underdamped process, overdamped process, inverse process, non-linear process, large system delay, large step disturbance). Additionally, the *max-product* composition provided consistently *better* control than the *max-min* composition, based on minimum distance criteria. The new controller algorithm also performed *better* than a discrete PI controller, with gain scheduling, for a highly non-linear process.

For all the simulations performed the absolute error, J_1, was less than 5% of the process output. Thus the fuzzy predictive control algorithm has demonstrated that it is capable of successfully controlling a variety of process situations. As well, the simulation with the Shell process data [Cott, 1995], demonstrated that the algorithm can handle large ($\tau \approx 6$) process dead time.

The final result of this work is a *practical fuzzy logic controller* suitable for industrial applications. Applications of *fuzzy control* to the Edmonton Water Treatment plant are currently planned as part of my NSERC PDF work.

11.3 Recommendations

Future work in the area of fuzzy model-based control could include:

(1) modification of the control algorithm presented in this thesis so that the dynamic matrix, $\boldsymbol{R}$, is larger (i.e. 7×7×7) however, the dynamic calculation is only implemented for *large* errors. *Near* the setpoint, *mean level control* or the 2-dimensional gain matrix, $\boldsymbol{G}$, would be employed.

(2) with regard to the fuzzy identification algorithm developed as part of this thesis, versus the averaging algorithm by Shaw *et al.* [1992], a study could be made to determine under what conditions each algorithm is ideally suited. Preliminary work in this area would suggest that the identification algorithm proposed in this thesis is more suited to systems with *large* noise/disturbances.

(3) the effect K and τ on a first order process system are well known. The entries of the relation matrix, $\boldsymbol{R}$, however, do not clearly indicate their effect on the dynamic performance of the relational model. As study could be undertaken to relate the shape or fuzzy measure of the relational matrix, $\boldsymbol{R}$, to τ.

(4) the controller algorithm presented in this thesis does not include the defuzzification algorithm in the overall optimization. Thus the defuzzification procedure chosen may results in some auxiliary error which in turn weakens the fuzzy model at the numeric level. Modification of the controller algorithm to include a joint optimization of the defuzzification and the relational model might then be considered in future work.

REFERENCES

[1] Baboshin, N.P., Naryshkin, D.G., "On Identification of Multidimensional Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 35, pp. 325-331, 1990.

[2] Bourke, M.M., Fisher, D.G., "The Complete Resolution of Cartesian Products of Fuzzy Sets, *Fuzzy Sets and Systems*, Vol. 63, pp. 111-115, 1994.

[3] Bourke, M.M., Fisher, D.G., "Calculation and Application of a Minimum Fuzzy Relational Matrix", *Fuzzy Sets and Systems*, Vol. 74, pp. 225-236, 1995(a).

[4] Bourke, M.M., Fisher, D.G., "Convergence, Eigen Fuzzy Sets and Stability Analysis of Relational Matrices", *Fuzzy Sets and Systems*, accepted June 1995(b).

[5] Bourke, M.M., Fisher, D.G., "A Comparative Analysis of Identification Algorithms for Fuzzy Relational Matrices", *Fuzzy Sets and Systems*, submitted 1995(c).

[6] Box, G.E.P., Jenkins, G.M., *Time Series Analysis, Forecasting and Control*, Holden Day, San Francisco, 1970.

[7] Chen, J.Q., Lu, J.H., Chen, L.J., "An On-Line Identification Algorithm for Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 64, pp. 63-72, 1994.

[8] Cott, B.J., "Summary of the Process Identification Workshop at the 1992 Canadian Chemical Engineering Conference", *Journal of Process Control*, Vol. 5, pp. 109-113, 1995.

[9] Cott, B.J., "Summary of the Process Identification Workshop at the 1992 Canadian Chemical Engineering Conference", Shell Manufacturing Centre, Corunna, Ontario, 1992.

[10] DiNola, A., Pedrycz, W., Sessa, S., "Some Theoretical Aspects of Fuzzy Relation Equations Describing Fuzzy Systems", *Information Sciences*, Vol. 34, pp. 241-264, 1984.

[11] Ikoma, N., Pedrycz, W., Hirota, K.,. "Estimation of Fuzzy Relational Matrix by using Probabilistic Descent Method", *Fuzzy Sets and Systems*, Vol. 57, pp. 335-349, 1993.

[12] Pedrycz, W., "Numerical and Applicational Aspects of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 11, pp. 1-18, 1983.

[13] Pedrycz, W., "An Identification Algorithm in Fuzzy Relational Systems", *Fuzzy Sets and Systems*, Vol. 13, pp. 153-167, 1984.

[14] Pedrycz, W., "Fuzzy Models and Relational Equations", *Mathematical Modeling*, Vol. 9., pp. 427-434, 1987.

[15] Pedrycz, W., "Approximate Solutions of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 28, pp. 183-202, 1988.

[16] Pedrycz, W., "Algorithms for Solving Fuzzy Relational Equations in a Probabilistic Setting", *Fuzzy Sets and Systems*, Vol. 38, pp. 313-327, 1990.

[17] Pedrycz, W., "Processing in Relational Structures: Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 40, pp. 77-106, 1991.

[18] Pedrycz, W., "Neurocomputations in Relational Systems", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, pp. 2889-297, 1991.

[19] Sanchez, E., "Resolution of Composite Fuzzy Relation Equations", *Information and Control*, Vol. 30, pp. 38-48, 1976.

[20] Sanchez, E., "Solutions in Composite Fuzzy Relation Equations: Application to Medical Diagnosis in Brouwerian Logic", In: *Fuzzy Automata and Decision Processes*, Gupta, M.M., Saridis, G.N., (Editors), Elsevier/North-Holland, Amsterdam, pp. 221-234, 1977.

[21] Shaw, I.S., Krüger, J.J., "New Fuzzy Learning Model with Recursive Estimation for Dynamic Systems", *Fuzzy Sets and Systems*, Vol. 48, pp. 217-229, 1992.

[22] Valente de Oliveira, J., "Neural Inspired Learning Rules for the Fuzzy Relational Structures", *Fuzzy Sets and Systems*, Vol. 57, pp. 41-53, 1993.

[23] Xu, C.-W., Lu, Y.-Z., "Fuzzy Model Identification and Self-Learning for Dynamic Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol., 17, pp. 683-689, 1987.

APPENDIX 1: INVERSE CALCULATION PROGRAM

A1.1 Contents

The contents of this appendix includes:

A1.2 Summary of Program

The inverse calculation program, written in MATLAB®, determines the range of analytical inverses for a fuzzy relational equation. The inverse can be determined either as a fuzzy cause (i.e. Given $\boldsymbol{R}$ and y, find x such that $x \circ \boldsymbol{R} = y$) or fuzzy ID (i.e. Given x and y, find $\boldsymbol{R}$ such that $x \circ \boldsymbol{R} = y$). The inverse is given as the unique maximum solution and the set of non-redundant minimum solutions.

A1.3 Getting Started

Function:

[Smax,Smin,USmin] = inverse(Q,R,T,oper,prob,tol)

The inverse function determines a unique maximum solution, S_{max}, the union of all the minimum solutions, $\cup S_{min}$, and all non-redundant minimum solutions, S_{min}.

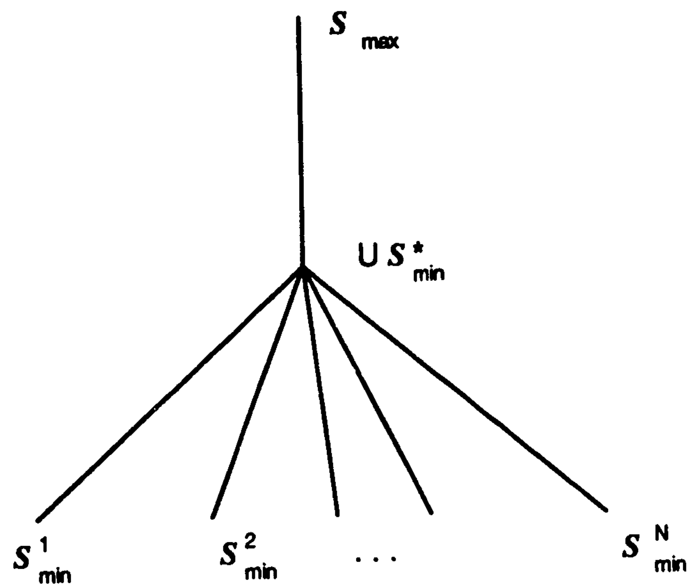


Figure A1.1: Illustration of Solution Structure

Inputs:

Test Tolerance: tol = 0.000001

The test tolerance for equality or approximate equality.
(i.e. If $|a-b| < tol$, then $a \approx b$)

Operation: oper = 0; for *max-prod*
oper = 1; for *max-min*

The fuzzy sets can be combined either with the *max-prod* composition or the *max-min* composition.

(1) Fuzzy Cause: prob = 1;

Problem 1 is that of fuzzy cause (i.e. Given ***R*** and ***y***, find ***x*** such that $x \circ R = y$)

For fuzzy cause problems, ***R*** and ***T*** are inputs and ***Q*** is set to a null matrix (i.e. ***Q*** = [ ]) as it is the quantity being determined.

[Smax,Smin,USmin] = inverse([ ],R,T,oper,prob,tol);

R is the relational matrix, (i.e. ***R*** = [.2 .5 .9 ; .3 .6 .8 ; .1 .5 .7]) and ***T*** is the result of the relational equation and is either a single vector (***b*** = [.3 .6 .8]) or several vectors that comprise a matrix (***T*** = [.3 .6 .8; .4 .7 .2])

(2) Fuzzy ID: prob = 2;

Problem 2 is that of fuzzy identification (i.e. Given ***x*** and ***y***, find ***R*** such that $x \circ R = y$)

For fuzzy identification problems, ***Q*** and ***T*** are inputs and ***R*** is set to a null matrix (i.e. ***R*** = [ ]) as it is the quantity being determined.

[Smax,Smin,USmin] = inverse(Q,[ ],T,oper,prob,tol);

Q is the fuzzy input to the relational equation and is either a single vector, (***a*** = [.5 1.0 .7]) or several vectors that comprise a matrix (i.e. ***Q*** = [.2 .5 .9 ; .3 .6 8])and ***T*** is the result of the relational equation and is either a single vector (***b*** = [.3 .6 .8]) or several vectors that comprise a matrix (***T*** = [.3 .6 .8; .4 .7 .2]). Note that the size of the resultant ***T*** must correspond to the size of the input ***Q***.

Outputs:

As shown in Figure A1.1, the inverse function determines:

Smax	- the unique maximum solution
USmin	- the union of all the minimum solutions
Smin	- all non-redundant minimum solutions

A1.4 Program Structure

```
INVERSE →→ ALPHA
        ↓
        →→ MAXOP
        ↓
        →→ BETA
        ↓
        →→ DELTA    →→ BETA
        ↓
        →→ PHISETS  →→ SSTAR
        ↓           ↓
        ↓           →→ BINARY
        ↓
        →→ PHITEST  →→ MAXOP
```

A1.5 Program Description

ALPHA.m

function [Mmax] = alpha(Q,R,T,oper,prob)

Determines the maximum inverse of fuzzy relation equations for *max-min* and *max-product* compositions

INPUT:

$\boldsymbol{Q}$ is a $i \times j$ relational matrix (for row vector $i=1;\boldsymbol{a}$)
$\boldsymbol{R}$ is an $j \times k$ relational matrix
$\boldsymbol{T}$ is a $i \times k$ relational matrix (for row vector $i=1;\boldsymbol{b}$)

oper - determines the composition: 0 = maxprod; 1 = maxmin
prob - determines problem type: 1 = cause; 2 = id

COMPOSITION: $\boldsymbol{Q}(i,j)*\boldsymbol{R}(j,k)=\boldsymbol{T}(i,k)$
OR $\boldsymbol{a}(j)*\boldsymbol{R}(j,k)=\boldsymbol{b}(k)$

OUTPUT:

Mmax - maximum solution

BETA.m

function [Mbeta] = beta(Q,R,T,oper,prob)

Determines the initial estimate of minimum solutions of fuzzy relation equations for *max-min* and *max-product* compositions

INPUT:

$\boldsymbol{Q}$ is a $i \times j$ relational matrix (for row vector $i=1;\boldsymbol{a}$)
$\boldsymbol{R}$ is an $j \times k$ relational matrix
$\boldsymbol{T}$ is a $i \times k$ relational matrix (for row vector $i=1;\boldsymbol{b}$)

oper - determines the composition: 0 = maxprod; 1 = maxmin
prob - determines problem type: 1 = cause; 2 = id

COMPOSITION: $\boldsymbol{Q}(i,j)*\boldsymbol{R}(j,k)=\boldsymbol{T}(i,k)$
OR $\boldsymbol{a}(j)*\boldsymbol{R}(j,k)=\boldsymbol{b}(k)$

OUTPUT:

Mbeta - union of minimum solutions

BINARY.m

function [bin] = binary(n)

Converts a number to its binary equivalent in vector format

DELTA.m

function [Mdelta] = delta(Mmax,Mbeta,oper,prob,tol)

Determines the union of all minimum solution including redundant solution of fuzzy relation equations for *max-min* and *max-product* compositions

INPUT:

Mmax - output from ALPHA.m
Mbeta - output from BETA.m

oper - determines the composition: 0 = maxprod; 1 = maxmin
prob - determines problem type: 1 = cause; 2 = id
tol - is the calculation tolerance

OUTPUT:

Mdelta - maximum solution

INVERSE.m

function [Smax,Smin,USmin] = inverse(Q,R,T,oper,prob,tol)

Determines the analytical inverse of fuzzy relation equations for *max-min* and *max-prod* compositions

INPUT:

Q is a $i{\times}j$ relational matrix (for row vector i=1;***a***)
R is an $j{\times}k$ relational matrix
T is a $i{\times}k$ relational matrix (for row vector i=1;***b***)

oper - determines the composition: 0 = *max-prod*; 1 = *max-min*
prob - determines problem type: 1 = cause; 2 = id
tol - is the calculation tolerance

Fuzzy Cause Example: [Smax,Smin,USmin] = inverse([ ],R,T,oper,prob,tol);
Fuzzy ID Example: [Smax,Smin,USmin] = inverse(Q,[ ],T,oper,prob,tol);

INVERSE.m (Cont'd)

COMPOSITION: $Q(i, j)*R(j, k)=T(i, k)$
OR $a(j)*R(j, k)=b(k)$

OUTPUT:

Smax - maximum solution
Smin - all non-redundant minimum solutions
USmin - union of all non-redundant minimum solutions

MAXOP.m

function T = maxop(Q,R,oper)

Determines the analytical solution of fuzzy relation equations for *max-min* and *max-product* compositions

INPUT:

Q is a $i \times j$ relational matrix (for row vector $i=1;a$)
R is an $j \times k$ relational matrix

oper - determines the composition: 0 = maxprod; 1 = maxmin

COMPOSITION: $Q(i, j)*R(j, k)=T(i, k)$
OR $a(j)*R(j, k)=b(k)$

OUTPUT:

T is a $i \times k$ relational matrix (for row vector $i=1;b$)

PHISETS.m

function [Phimax] = phisets(Mdelta)

Determines all non-redundant minimum solutions of Mdelta for the fuzzy cause case (prob = 1) for *max-min* and *max-product* operations

INPUT:

Mdelta - output from DELTA.m

OUTPUT:

Phimax - non-redundant solution matrix

PHITEST.m

function [Phisub] = phitest(Phimax,R,B,oper,tol)

Brute force check for additional redundant minimum solutions for the fuzzy cause case (prob = 1) for *max-min* and *max-product* compositions

INPUT:

Phimax	is the $1 \times j$ row vector from PHISETS.m
R	is the $j \times k$ relational matrix
T	is the $1 \times k$ output row vector

oper - determines the composition: 0 = maxprod; 1 = maxmin
tol - is the calculation tolerance

COMPOSITION: Phimax(j)**$R(j, k)=b(k)$*

OUTPUT:

Phisub - complete non-redundant minimum solution

SSTAR.m

function [Sast] = Sstar(Mdelta)

Delete the vectors of Mdelta that results in redundant solution vectors

A1.6 Program Listings

MATLAB™ M-File Listing: ALPHA.M

```
function [Mmax] = alpha(Q,R,T,oper,prob)
%
% alpha composition for maxmin & maxproa
%
% Q is a  ixj relational matrix (for row vector i=1;A)
% R is an jxk relational matrix
% T is a  ixk relational matrix (for row vector i=1;B)
%
% Composition:  Q(i,j)*R(j,k)=T(i,k) or A(j)*R(j,k)=B(k)
%
% oper - determines the composition: 0 = maxprod; 1 = maxmin
% prob - determines problem type: 1 = cause; 2 = id
%

% Fuzzy Cause Calculation:

if prob==1,

  [Rr Rc] = size(R);
  [Tr Tc] = size(T);

  Qmax = [];
  for j = 1:Rr,

    Qtmp = [];
    for i = 1:Tr,
      for k = 1:Rc,
        if R(j,k) <= T(i,k),
          Qtmp(i,k) = 1;
        else
          if oper == 1,
            Qtmp(i,k) = T(i,k);
          else
            Qtmp(i,k) = T(i,k)./R(j,k);
          end
        end
      end
    end
    Qtmp = min(Qtmp')
    Qmax = [Qmax ; Qt
  end
  Mmax = Qmax';
end
```

MATLAB™ M-File Listing: ALPHA.M (Cont'd)

```
% Fuzzy ID Calculation:

if prob==2,

  [Qr Qc] = size(Q);
  [Tr Tc] = size(T);

  if Tr > 1,
    Q = Q';
  end
  T = T';

  [Qr Qc] = size(Q);
  [Tr Tc] = size(T);

  Rmax = [];
  for i = 1:Qr,

     Rtmp = [];
     for j = 1:Qc,
        if Tc == 1,
          l = i;
        else
          l = j;
        end
        for k = 1:Tr,
           if Q(i,j) <= T(k,l),
              Rtmp(k,j) = 1;
           else
              if oper == 1,
                 Rtmp(k,j) = T(k,l);
              else
                 Rtmp(k,j) = T(k,l)./Q(i,j);
              end
           end
        end
     end
     if Tc > 1
        Rtmp = min(Rtmp');
     end
```

```
    Rmax = [Rmax ; Rtmp];
  end
  Rmax = Rmax';
  if Tc > 1,
    Q = Q';
    Rmax = Rmax';
  end
  T = T';
  Mmax = Rmax;
end
```

MATLAB™ M-File Listing: BETA.M

```
function [Mbeta] = beta(Q,R,T,oper,prob)
%
% beta composition for maxmin & maxprod
%
% Q is a  1xj relational matrix (for row vector i=1;A)
% R is an jxk relational matrix
% T is a  1xk relational matrix (for row vector i=1;B)
%
% oper - determines the composition: 0 = maxprod; 1 = maxmin
%

% Fuzzy Cause Calculation:

if prob==1,

  [Rr Rc] = size(R);
  [Tr Tc] = size(T);

  Qbeta = [];
  for j = 1:Rr,
     for i = 1:Tr,
        for k = 1:Rc,
           if R(j,k) < T(i,k),
              Qbeta(j,k) = 0;
           else
              if oper == 1,
                 Qbeta(j,k) = T(i,k);
              else
                 if R(j,k) == 0 & T(i,k) == 0,
                    Qbeta(j,k) = 0;
                 else
                    Qbeta(j,k) = T(i,k)./R(j,k);
                 end
              end
           end
        end
     end
  end

  Mbeta = Qbeta;
end
```

```
% Fuzzy ID Calculation:

if prob==2,

  T = T';

  [Qr Qc] = size(Q);
  [Tr Tc] = size(T);

  Rbeta = [];
  for i = 1:Qr,
     for j = 1:Qc,
        for k = 1:Tr,
           if Q(i,j) < T(k,i),
             Rbeta(k,j) = 0;
           else
             if oper == 1,
               Rbeta(k,j) = T(k,i);
             else
               if Q(i,j) == 0 & T(k,i) == 0,
                 Rbeta(k,j) = 0;
               else
                 Rbeta(k,j) = T(k,i)./Q(i,j);
               end
             end
           end
        end
     end
  end
  Mbeta = Rbeta';
end
•
```

MATLAB™ M-File Listing: BINARY.M

```
function [bin] = binary(n)

% Converts a number to its binary equivalent in vector format

i = 0;

while n > 0,
    i = i+1;
    if (n-2^(i-1)) < 2^(i-1),
      M(i) = 1;
      n = n-2^(i-1);
      i = 0;
    end
end

bin = fliplr(M);
```

•

MATLAB™ M-File Listing: DELTA.M

```
function [Sdelta] = delta(Smax,Sbeta,oper,prob,tol)
%
% delta composition for maxmin & maxprod
%
% Q is a  1xj relational matrix (for row vector i=1;A)
% R is an jxk relational matrix
% T is a  1xk relational matrix (for row vector i=1;B)
%
% oper - determines the composition: 0 = maxprod; 1 = maxmin
% prob - determines problem type: 1 = cause; 2 = id
%

[Br,Bc] = size(Sbeta);
Sdelta = [];

%Fuzzy Cause

if prob == 1,
  Qdelta = [];
  for i = 1:Br,
     for j = 1:Bc,
        if oper == 1,
          Qdelta(i,j) = beta([],Smax(i),Sbeta(i,j),oper,prob);
        else
          if abs(Sbeta(i,j) - Smax(i)) <= tol,
            Qdelta(i,j) = Smax(i);
          else
            Qdelta(i,j) = 0;
          end
        end
     end
  end
  Sdelta = Qdelta;
end
```

MATLAB™ M-File Listing: DELTA.M (Cont'd)

```
%Fuzzy Identification

if prob == 2,
  Rdelta = [];
  for i = 1:Br,
     for j = 1:Bc,
        if oper == 1,
          % Note: This is the beta operation
          if Sbeta(i,j) <= Smax(i,j),
            Rdelta(i,j) = Sbeta(i,j);
          else
            Rdelta(i,j) = 0;
          end
        else
          if abs(Sbeta(i,j) - Smax(i,j)) <= tol,
            Rdelta(i,j) = Smax(i,j);
          else
            Rdelta(i,j) = 0;
          end
        end
     end
  end
  Sdelta = Rdelta;
end
```

MATLAB™ M-File Listing: INVERSE.M

```
function [Smax,Smin,USmin] = inverse(Q,R,T,oper,prob,tol)
%
% Determines the inverse of the relation equation A.R = B
% for maxmin & maxprod compositions
%
% Q is a  ixj relational matrix (for row vector i=1;A)
% R is an jxk relational matrix
% T is a  ixk relational matrix (for row vector i=1;B)
%
% Composition:  Q(i,j)*R(j,k)=T(i,k) or A(j)*R(j,k)=B(k)
%
% oper - determines the composition: 0 = maxprod; 1 = maxmin
% prob - determines problem type: 1 = cause; 2 = id
%
%

 Smax = [];
 Smin = [];

 % Fuzzy Cause:

 if prob == 1,

   [Tr Tc] = size(T);
   [Rr Rc] = size(R);
   Zvector = zeros(1,Rr);

   fprintf('For the Relational Matrix:')
   R

   fprintf('For the Solution Matrix:')
   T

   Qmax = [];
   Qmin = [];
```

```
% Determine Qmax

Smax = alpha([],R,T,oper,prob);
Ttmp = maxop(Smax,R,oper);

if abs(sum(sum(Ttmp-T))) > tol,
  fprintf('No inverse solution exists! \n')
  return
end

% Determine Qmin

USmin = zeros(Smax);
for l = 1:Tr,

   Tvector = T(l,:);

   Qbeta = beta([],R,Tvector,oper,prob);
   Qdelta = delta(Smax(l,:),Qbeta,oper,prob,tol);

   % Remove Redundant Solutions

   phimax = phisets(Qdelta);
   phisub = phitest(phimax,R,Tvector,oper,tol);

   [pr,pc] = size(phisub);
   if pr > 1,
     phivect = max(phisub);
   else
     phivect = phisub;
   end
   USmin = max(USmin,phivect);
   Smin = [Smin ;phisub];
   if l < Tr,
     Smin = [Smin ;Zvector];
   end
 end
```

```
  % Print Solutions

  fprintf('The maximal inverse solution is:')
  Qmax = Smax

  fprintf('The minimum inverse solution(s) are:')
  Qmin = Smin

  fprintf('The union of the minimum inverse solution(s) are:')
  UQmin = USmin

end

% Fuzzy ID

if prob == 2,

  [Tr Tc] = size(T);
  [Qr Qc] = size(Q);
  Zvector = zeros(1,Tc);

  fprintf('For the Input Matrix:')
  Q

  fprintf('For the Solution Matrix:')
  T

  Rmax = [];
  Rmin = [];

  % Determine Rmax

  Smax = alpha(Q,[],T,oper,prob);
  Ttmp = maxop(Q,Smax,oper);

  if abs(sum(sum(Ttmp-T))) > tol,
    fprintf('No inverse solution exists! \n')
    return
  end

  % Print Max Solution

  fprintf('The maximal inverse solution is:')
  Rmax = Smax
```

MATLAB™ M-File Listing: INVERSE.M (Cont'd)

```
% Determine Rmin

% Print Min Solutions Banner

fprintf('The minimum inverse solution(s) are:')

USmin = zeros(Smax);
for l = 1:Qr,

   Tvector = T(l,:);
   Qvector = Q(l,:);

   Rbeta = beta(Qvector,Smax,Tvector,oper,prob);
   Rdelta = delta(Smax,Rbeta,oper,prob,tol);

   USmin = max(USmin,Rdelta);
   Smin = [Smin ;Rdelta];

   if l < Qr,
     Smin = [Smin ;Zvector];
   end

   % Print Min Solutions

   Rmin = Rdelta
   fprintf('\n')

 end

fprintf('The union of the minimal inverse solution is:')
URmin = USmin

end
•
```

MATLAB™ M-File Listing: MAXOP.M

```
function T = maxop(Q,R,oper)
%
% maxop composition for maxmin & maxprod
%
% Q is a  ixj relational matrix (for vector i=1;A)
% R is an jxk relational matrix
% T is a  ixk relational matrix (for vector i=1;B)
%
% oper - determines the composition: 0 = maxprod; 1 = maxmin
%

[Rr Rc] = size(R);
[Qr Qc] = size(Q);

T = [];
for k = 1:Qr,
   tmp = [];
   for i = 1:Rc,
      for j = 1:Rr,
         if oper == 1,
           tmp(j,i) = min(Q(k,j),R(j,i));
         else
           tmp(j,i) = Q(k,j).*R(j,i);
         end
      end
   end
   T = [T ; max(tmp)];
end
•
```

MATLAB™ M-File Listing: PHISETS.M

```
function [Phimax] = phisets(Qdelta)
%
% phiset operation for maxmin & maxprod
%
% A is a  pxn relational matrix (for vector p=1)
% R is an nxm relational matrix
% B is a  pxm relational matrix (for vector p=1)
%
% oper - determines the composition: 0 = maxprod; 1 = maxmin
%

Sast = sstar(Qdelta);

[Sr Sc] = size(Sast);
onemat = ceil(Sast);
sumc = sum(onemat);
comb = prod(sumc);
numcol = sum(sumc);

% Expand matrix to column vectors with a single entry
%
AA = [];
for i = 1:Sc,
   for j = 1:Sr,
      onev = zeros(Sr,1);
      onev(j) = 1;
      tmp = [];
      tmp = Sast(j,i)*onev;
      if sum(tmp) ~= 0,
         AA = [AA tmp];
      end
   end
end

% Determine the number of repetitions for each column
%
reps = zeros(1,Sc);
last = comb;
for i = 1:Sc,
   if sumc(i) ~= 1,
      reps(i) = last./sumc(i);
      last = reps(i);
   end
end
```

```
% Determine the remainder for each column
%
rem = zeros(1,Sc);
for i = 1:Sc,
   if reps(i) ~= 0,
     rem(i) = comb./(reps(i)*sumc(i));
   end
end

% Calculate the matrix of possible combinations
%
mcomb = ones(comb,Sc);
for p = 1:Sc,
   if sumc(p) ~= 1,
     index = 1;
     for i = 1:rem(p),
        for k = 1:sumc(p),
           for j = 1:reps(p),
              mcomb(index,p) = k;
              index = index + 1;
           end
        end
     end
   end
end

% Calculate the maximum for each Phi matrix
%
Phimax = [];
for i = 1:comb,
   mvect = [];
   for k = 1:Sc,
      if sumc(k) == 1,
        mvect = [mvect 1];
      else
        xn = mcomb(i,k);
        tmp = binary(2^(xn-1));
        q = sumc(k) - length(tmp);
        for r = 1:q,
           tmp = [0 tmp];
        end
        mvect = [mvect tmp];
      end
   end
```

```
    Phi = [];
    for p = 1:numcol,
      if mvect(p) == 1,
        Phi = [Phi AA(:,p)];
      end
    end

    [Pr,Pc] = size(Phi);
    if Pc == 1,
      Phimax = [Phima: ; Phi'];
    else
      Phimax = [Phimax ; max(Phi')];
    end
  end
  •
```

MATLAB™ M-File Listing: PHITEST.M

```
function [Phisub] = phitest(Phimax,R,B,oper,tol)
%
% subset of Phi which satisfies A.R = B
%
% A is a  pxn relational matrix (for vector p=1)
% R is an nxm relational matrix
% B is a  pxm relational matrix (for vector p=1)
%
% oper - determines the composition: 0 = maxprod; 1 = maxmin
%

% Phimax is a matrix of row vectors

[r, c] = size(Phimax);

% Test for equal solutions:
%
for i = 1:r,
   for j = 1:r,
      if i ~=j,
         if Phimax(i,:) == Phimax(j,:),
            for k = 1:c,
               Phimax(j,k) = 0;
            end
         end
      end
   end
end

Phis1 = [];
for i = 1:r,
   Btmp = [];
   Btmp = maxop(Phimax(i,:),R,oper);
   if abs(sum(Btmp - B)) < tol,
      Phis1 = [Phis1 ; Phimax(i,:)];
   end
end
```

```
% Test for greatest lower bound solution
%

[r, c] = size(Phis1);

tmp = zeros(r,r);
for i = 1:r,
  for j = 1:r,
    if i ~=j,
      if Phis1(i,:) >= Phis1(j,:),
        tmp(i,j) = 1;
      else
        tmp(i,j) = 0;
      end
    end
  end
end

% Zero greatest lower bound solution vectors
%
for i = 1:r,
  if max(tmp(i,:)) == 1,
    for j = 1:c,
      Phis1(i,j) = 0;
    end
  end
end

% Delete zeroed vectors
%
Phisub = [];
for i = 1:r
  if max(Phis1(i,:)) ~= 0,
    Phisub = [Phisub ; Phis1(i,:)];
  end
end
•
```

MATLAB™ M-File Listing: SSTAR.M

```
function [Sast] = Sstar(Rdb)

% Improved inverse solution

[r c] = size(Rdb);

% Delete the vectors of Rdb that results in redundant solution
% vectors
%
for i = 1:c,
  for j = 1:c,
    tmp = [];
    for k = 1:r,
      if (i ~= j & sum(Rdb(:,i)) ~=0)
        tmp(1,k)=(Rdb(k,i)~=0 & Rdb(k,j)~=0 & Rdb(k,i)>=Rdb(k,j));
        tmp(2,k)=(Rdb(k,i)==0);
      end
    end

    tmp(3,:)=sum(tmp);

    if min(tmp(3,:)) >= 1,
      for p = 1:r,
        Rdb(p,j) = 0;
      end
    end
  end
end

tmp = [];
tmp = sum(Rdb);
Sast = [];
for i = 1:c,
  if tmp(i) ~= 0,
    Sast = [Sast Rdb(:,i)];
  end
end
•
```

REFERENCES

[1] Di Nola, A., Pedrycz, W., Sessa, S., "On Solutions of Fuzzy Relational Equations and Their Characterization", *BUSEFAL*, Vol. 12, pp. 60-71, 1982.

[2] Di Nola, A., Sessa, S., "On the Fuzziness of Solutions of @-Fuzzy Relation Equations on Finite Spaces", *Fuzzy Sets and Systems*, Vol. 11, pp. 65-77, 1983.

[3] Di Nola, A., Pedrycz, W., Sessa, S., Pei-Zhuang, W., "Fuzzy Relation Equations Under a Class of Triangular Norms: A survey and New Results", *Stochastica*, Vol. 8, pp. 99-145, 1984.

[4] Di Nola, A., "Relational Equations in Totally Ordered Lattices and Their Complete Resolution", *Journal of Mathematical Analysis and Applications*, Vol. 107, pp. 148-155, 1985.

[5] Pappis, C.P., Sugeno, M., "Fuzzy Relational Equations and the Inverse Problem", *Fuzzy Sets and Systems*, Vol. 15, pp. 79-90, 1985.

[6] Pappis, C.P., "Multi-Input Multi-Output Fuzzy Systems and the Inverse Problem", *European Journal of Operational Research*, Vol. 28, pp. 228-230, 1987.

[7] Pappis, C.P., "Resolution of Cartesian Products of Fuzzy Sets", *Fuzzy Sets and Systems*, Vol. 26, pp. 387-391, 1988.

[8] Pappis, C.P., Adamopoulos, G.I., "A Computer Algorithm for the Solution of the Inverse Problem of Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 39, pp. 279-290, 1991.

[9] Pappis, C.P., Adamopoulos, G.I., "A Software Routine to Solve the Generalized Inverse Problem of Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 47, pp. 319-322, 1992.

[10] Pedrycz, W., "Fuzzy Relational Equations with Triangular Norms and their Resolution", *BUSEFAL*, Vol. 11, pp. 24-32, 1982.

[11] Pedrycz, W., "Fuzzy Relational Equations with Generalized Connectives and Their Applications", *Fuzzy Sets and Systems*, Vol. 10, pp. 185-201, 1983(a).

[12] Pedrycz, W., "Numerical and Aplicational Aspects of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 11, pp. 1-18, 1983(b).

[13] Pedrycz, W., "Some Applicational Aspects of Fuzzy Relational Equations in Systems Analysis", *International Journal of General Systems*, Vol. 9, pp. 125-132, 1983(c).

[14] Pedrycz. W., "An Identification Algorithm in Fuzzy Relation Systems", *Fuzzy Sets and Systems*, Vol. 13, pp. 153-167, 1984(a).

[15] Pedrycz, W., "Identification in Fuzzy Systems", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 14, pp. 361-366, 1984(b).

[16] Pedrycz, W., "On Generalized Fuzzy Relational Equations and Their Applications", *Journal of Mathematical Analysis and Applications*, Vol. 107, pp. 520-536, 1985.

[17] Pedrycz, W., "Approximate Solutions of Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 28, pp. 183-202, 1988.

[18] Pedrycz, W., *Fuzzy Control and Fuzzy Systems*, John Wiley & Sons, Inc., New York, N.Y., 1989.

[19] Pedrycz, W., "Inverse Problem in Fuzzy Relational Equations", *Fuzzy Sets and Systems"*, Vol. 36, pp. 277-291, 1990.

[20] Pedrycz, W., "Processing in Relational Structures: Fuzzy Relational Equations", *Fuzzy Sets and Systems*, Vol. 40, pp. 77-106, 1991.

[21] Pei-Zhuang, W., Sessa, S., Di Nola, A., Pedrycz, W., "How Many Lower Solutions Does a Fuzzy Relation Equation Have? , *BUSEFAL*, Vol. 18, pp. 67-74, 1984.

[22] Sanchez, E., "Resolution of Composite Fuzzy Relation Equations", *Information and Control*, Vol. 30, pp. 38-49, 1976.

[23] Sanchez, E., "Solutions in Composite Fuzzy Relation Equations: Applications to Medical Diagnosis in Brouwerian Logic", In: *Fuzzy Automata and Decision Processes*, Gupta, M.M., Saridis, G.N., Ragade, R.K., North-Holland, Amsterdam, pp. 221-234, 1977.

[24] Sessa, S., "Some Results in the Setting of Fuzzy Relation Equations Theory", *Fuzzy Sets and Systems*, Vol. 14, pp. 281-297, 1984.

[25] Sessa, S., "Finite Fuzzy Relation Equations with Unique Solution in Complete Brouwerian Lattices", *Fuzzy Sets and Systems*, Vol. 29, pp. 103-113, 1989.

APPENDIX 2: SOLUTION GUIDE FOR INVERSE PROBLEMS FOR THE *MAX-PRODUCT* COMPOSITION

A2.1 Contents

The contents of this appendix includes:

A2.2 Introduction

The original work by Sanchez [1976] considered the resolution of the following fuzzy relational equation with *max-min* composition :

$$T(u_i, w_k) = Q(u_i, v_j) \circ R(v_j, w_k) = \bigvee_j [Q(u_i, v_j) \wedge R(v_j, w_k)] \tag{A2.1}$$

A short time later, Sanchez [1977] then considered the simplified version of equation (A2.1), where $\mathcal{U}$ s defined for a single point:

$$b(w_k) = a(v_j) \circ R(v_j, w_k) = \bigvee_j [a(v_j) \wedge R(v_j, w_k)] \tag{A2.2}$$

In equations (A2.1) and (A2.2) the symbols $\vee$ and $\wedge$ denote the fuzzy set operators or connectors *max* and *min*, respectively. And the symbol $\circ$ denotes the *max-min* operator.

Di Nola *et.al.* [1984] showed that if the *min* operator $\wedge$ in equations (A2.1) and (A2.2) was replaced with a *t-norm* operator then the following fuzzy relational equations should be considered:

$$T(u_i, w_k) = Q(u_i, v_j) \odot R(v_j, w_k) = \bigvee_j [Q(u_i, v_j) \text{ t } R(v_j, w_k)] \qquad \text{(A2.3)}$$

$$b(w_k) = a(v_j) \odot R(v_j, w_k) = \bigvee_j [a(v_j) \text{ t } R(v_j, w_k)] \qquad \text{(A2.4)}$$

In equations (A2.3) and (A2.4) the symbol $\vee$ still denotes the fuzzy set connector *max* while the symbol **t** denotes any *t-norm* operator. The symbol $\odot$ represent the *max-t-norm* composition operator.

There are two basic inverse problems to be investigated and resolved, that of fuzzy identification and fuzzy cause. When these two problems are applied to each of equations (A2.3) and (A2.4) the result is the four problem statements listed below:

(1) "Given the fuzzy relations R and b, find all fuzzy sets a such that $a \odot R = b$".

(2) "Given the fuzzy relations R and T, find all fuzzy relations Q such that $Q \odot R = T$".

(3) "Given the fuzzy sets a and b, find the fuzzy relation R such that $a \odot R = b$".

(4) "Given the fuzzy relations Q and T, find the fuzzy relation R such that $Q \odot R = T$".

Problem statements (1) and (2) represent the search for a fuzzy cause, while statements (3) and (4) represent fuzzy identification. All four problem statements are valid when $\odot$ represents either *max-min* or *max-product* composition.

The complete inverse solution algorithms, with examples, for when $\odot \equiv$ *max-product* composition are provided in this appendix. Each partition in the appendix corresponds to the appropriate Problem Statement.

In order to consider the problem of resolving the inverse of the fuzzy relational equation the following definitions are required:

Definition 1: For a and $b \in [0,1]$, the ψ-composition is defined as:

$$a \,\psi\, b = \begin{cases} 1 & \text{if } \; a \leq b \\ b/a & \text{if } \; a > b \end{cases}$$

The composition $a \,\psi\, b$ is call the *relative pseudo-complement* of a in b.

Definition 2: For a and $b \in [0,1]$, the θ-composition is defined as:

$$a \,\theta\, b = \begin{cases} 0 & \text{if } \; a < b \;\; \text{or} \;\; a = b = 0 \\ b/a & \text{if } \; a \geq b \end{cases}$$

Definition 3: For a and $b \in [0,1]$, the γ-composition is defined as:

$$a \,\gamma\, b = \begin{cases} a & \text{if } \; a = b \\ 0 & \text{if } \; a \neq b \end{cases}$$

Definition 4: For the *max-product* composition the β-composition can be specifically defined for a and $b \in [0,1]$ as:

$$a \,\beta\, b = (a \,\theta\, b) \,\gamma\, (a \,\psi\, b)$$

Other definitions required for this appendix can be found in Chapter 5.

A2.3 1. Given R and b, find a

Greatest Solution

Given the fuzzy relation $R \in \mathcal{V} \times \mathcal{W}$ and the fuzzy set $b \in \mathcal{W}$, the *greatest* solution, $\hat{a} \in \mathcal{A}$, such that $\hat{a}(v) \odot R(v, w) = b(w)$, is:

$$\hat{a}(v) = [R(v, w) \circledpsi b(w)^{-1}]^{-1}$$
$$= \bigwedge_{w} [R(v, w) \psi b(w)^{-1}]^{-1}$$

Minimal Solutions

The union of the *minimal* solutions, $\tilde{a}_z \in \mathcal{A}$, such that $\tilde{a}_z(v) \odot R(v, w) = b(w)$, is:

$$\tilde{a}_z(v) = \bigvee_{w} [\Phi(R(v, w)\ \beta\ b(w)^{-1})^{*}]^{-1}$$

$$= \bigvee_{w} [\Phi([R(v, w)\ \theta\ b(w)^{-1}] \gamma [R(v, w) \circledpsi b(w)^{-1}])^{*}]^{-1}$$

$$= \bigvee_{w} [\Phi([R(v, w)\ \theta\ b(w)^{-1}]\ \gamma\ [\tilde{a}(v)]^{-1})^{*}]^{-1}$$

where z is the cardinality of $\Phi(R(v, w)\ \beta\ b(w)^{-1})^{*}$.

Solution Example

Let $R(v, w) = \begin{pmatrix} 0.6 & 0.2 & 0.6 \\ 0.4 & 0.2 & 0.7 \\ 0.5 & 0.4 & 0.3 \end{pmatrix}$ and $b(w) = \begin{bmatrix} 0.5 & 0.4 & 0.42 \end{bmatrix}$ be given.

Find $\hat{a}$ and $\tilde{a}_z$.

Solution Maximum:

$$\hat{a}(v) = \bigwedge_{w}\left[\begin{pmatrix}0.6 & 0.2 & 0.6\\0.4 & 0.2 & 0.7\\0.5 & 0.4 & 0.3\end{pmatrix}\Psi\begin{pmatrix}0.5\\0.4\\0.42\end{pmatrix}\right]^{-1}$$

$$= \bigwedge_{w}\left[\begin{pmatrix}0.83 & 1.0 & 0.7\\1.0 & 1.0 & 0.6\\1.0 & 1.0 & 1.0\end{pmatrix}\right]^{-1}$$

$$= \begin{bmatrix}0.7 & 0.6 & 1.0\end{bmatrix}$$

Solution Minimum:

$$\tilde{a}_z(v) = \bigvee_{w}\left[\Phi\left[\begin{pmatrix}0.6 & 0.2 & 0.6\\0.4 & 0.2 & 0.7\\0.5 & 0.4 & 0.3\end{pmatrix}\beta\begin{pmatrix}0.5\\0.4\\0.42\end{pmatrix}\right]\right]^{-1}$$

$$= \bigvee_{w}\left[\Phi\begin{pmatrix}0.83 & 0.0 & 0.7\\0.0 & 0.0 & 0.6\\1.0 & 1.0 & 0.0\end{pmatrix}\gamma\begin{pmatrix}0.7\\0.6\\1.0\end{pmatrix}\right]^{-1}$$

$$= \bigvee_{w}\left[\Phi\begin{pmatrix}0.0 & 0.0 & 0.7\\0.0 & 0.0 & 0.6\\1.0 & 1.0 & 0.0\end{pmatrix}\right]^{-1}$$

$$= \bigvee_{w}\left[\Phi\begin{pmatrix}0.0 & 0.7\\0.0 & 0.6\\1.0 & 0.0\end{pmatrix}^{*}\right]^{-1} \qquad \text{(redundant columns removed)}$$

$$z = 1\times 2 = 2$$

$$\therefore \quad \tilde{a}_1(v) = \begin{bmatrix}0.0 & 0.6 & 1.0\end{bmatrix}$$

$$\tilde{a}_2(v) = \begin{bmatrix}0.7 & 0.0 & 1.0\end{bmatrix}$$

The complete solution is: $[0.0 \;\; 0.6 \;\; 1.0] \subseteq a \subseteq [0.7 \;\; 0.6 \;\; 1.0]$

$[0.7 \;\; 0.0 \;\; 1.0] \subseteq a \subseteq [0.7 \;\; 0.6 \;\; 1.0]$

A2.4 2. Given R and T, find Q

Greatest Solution

Given the two fuzzy relations $R \in \mathcal{V} \times \mathcal{W}$ and $T \in \mathcal{U} \times \mathcal{W}$ the *greatest* solution, $\hat{Q} \in \mathcal{U} \times \mathcal{V} \in \mathcal{Q}$, such that $\hat{Q}(u, v) \odot R(v, w) = T(u, w)$, is:

$$\hat{Q}(u, v) = [R(v, w) \circledast T(u, w)^{-1}]^{-1}$$

$$= \bigwedge_w [R(v, w) \,\psi\, T(w, u)]^{-1}$$

Minimal Solutions

The union of the *minimal* solutions, $\tilde{Q}_z \in \mathcal{U} \times \mathcal{V} \in \mathcal{Q}$, such that $\tilde{Q}_z(u, v) \odot R(v, w) = T(u, w)$, is:

$$\tilde{Q}_z(u_i, v) = \bigvee_w [\Phi(R(v, w) \,\beta\, T(u_i, w)^{-1})^{*}]^{-1}$$

$$= \bigvee_w [\Phi([R(v, w) \,\theta\, T(w, u_i)] \,\gamma\, [R(v, w) \circledast T(w, u_i)])^{*}]^{-1}$$

$$= \bigvee_w [\Phi([R(v, w) \,\theta\, T(w, u_i)^{-1}] \,\gamma\, [\hat{Q}(v, u_i)])^{*}]^{-1}$$

where z is the cardinality of $\Phi(R(v, w) \beta T(u_i, w)^{-1})^{*}$.

Solution Example

Let $R(v, w) = \begin{pmatrix} 0.6 & 0.2 & 0.6 \\ 0.4 & 0.2 & 0.7 \\ 0.5 & 0.4 & 0.3 \end{pmatrix}$ and $T(u, w) = \begin{pmatrix} 0.5 & 0.3 & 0.42 \\ 0.45 & 0.36 & 0.42 \end{pmatrix}$ be given.

Find $\hat{Q}$ and $\tilde{Q}_z$.

Solution Maximum:

$$\hat{Q}(u, v) = \bigwedge_w \left[\begin{pmatrix} 0.6 & 0.2 & 0.6 \\ 0.4 & 0.2 & 0.7 \\ 0.5 & 0.4 & 0.3 \end{pmatrix} \Psi \begin{pmatrix} 0.5 & 0.45 \\ 0.4 & 0.36 \\ 0.42 & 0.42 \end{pmatrix} \right]^{-1}$$

$$\hat{Q}(u_1, v) = \bigwedge_w \left[\begin{pmatrix} 0.8333 & 1.0 & 0.7 \\ 1.0 & 1.0 & 0.6 \\ 1.0 & 1.0 & 1.0 \end{pmatrix} \right]^{-1} = \begin{bmatrix} 0.7 & 0.6 & 1.0 \end{bmatrix}$$

$$\hat{Q}(u_2, v) = \bigwedge_w \left[\begin{pmatrix} 0.75 & 1.0 & 0.7 \\ 1.0 & 1.0 & 0.6 \\ 0.9 & 0.9 & 1.0 \end{pmatrix} \right]^{-1} = \begin{bmatrix} 0.7 & 0.6 & 0.9 \end{bmatrix}$$

$$\therefore \quad \hat{Q}(u, v) = \begin{pmatrix} 0.7 & 0.6 & 1.0 \\ 0.7 & 0.6 & 0.9 \end{pmatrix}$$

Solution Minimum:

$$\tilde{Q}_z(u, v) = \bigvee_w \left[\Phi \left[\begin{pmatrix} 0.6 & 0.2 & 0.6 \\ 0.4 & 0.2 & 0.7 \\ 0.5 & 0.4 & 0.3 \end{pmatrix} \beta \begin{pmatrix} 0.5 & 0.45 \\ 0.4 & 0.36 \\ 0.42 & 0.42 \end{pmatrix} \right] \right]^{-1}$$

$$\tilde{Q}_z(u_1, v) = \bigvee_w \left[\Phi \begin{pmatrix} 0.8333 & 0.0 & 0.7 \\ 0.0 & 0.0 & 0.6 \\ 1.0 & 1.0 & 0.0 \end{pmatrix} \gamma \begin{pmatrix} 0.7 \\ 0.6 \\ 1.0 \end{pmatrix} \right]^{-1} = \bigvee_w \left[\Phi \begin{pmatrix} 0.0 & 0.0 & 0.7 \\ 0.0 & 0.0 & 0.6 \\ 1.0 & 1.0 & 0.0 \end{pmatrix} \right]^{-1}$$

$$= \bigvee_w \left[\Phi \begin{pmatrix} 0.0 & 0.7 \\ 0.0 & 0.6 \\ 1.0 & 0.0 \end{pmatrix}^{*} \right]^{-1}$$

(redundant columns removed)

$$z = 1 \times 2 = 2$$

$$\therefore \quad \tilde{Q}_1(u_1, v) = [0.0 \quad 0.6 \quad 1.0]$$

$$\tilde{Q}_2(u_1, v) = [0.7 \quad 0.0 \quad 1.0]$$

$$\tilde{Q}_z(u_2, v) = \bigvee_w \left[\Phi \begin{pmatrix} 0.75 & 0.0 & 0.7 \\ 0.0 & 0.0 & 0.6 \\ 0.9 & 0.9 & 0.0 \end{pmatrix} \gamma \begin{pmatrix} 0.7 \\ 0.6 \\ 0.9 \end{pmatrix} \right]^{-1} = \bigvee_w \left[\Phi \begin{pmatrix} 0.0 & 0.0 & 0.7 \\ 0.0 & 0.0 & 0.6 \\ 0.9 & 0.9 & 0.0 \end{pmatrix} \right]^{-1}$$

$$= \bigvee_w \left[\Phi \begin{pmatrix} 0.0 & 0.7 \\ 0.0 & 0.6 \\ 0.9 & 0.0 \end{pmatrix}^{*} \right]^{-1} \quad \text{(redundant columns removed)}$$

$$z = 1 \times 2 = 2$$

$$\therefore \quad \tilde{Q}_1(u_2, v) = [0.0 \quad 0.6 \quad 0.9]$$

$$\tilde{Q}_2(u_2, v) = [0.7 \quad 0.0 \quad 0.9]$$

For the complete solution of Q there a 4 possibilities: 2 ways to choose $\tilde{Q}_z(u_1,v)$ and 2 ways to choose from $\tilde{Q}_z(u_1,v)$. Thus:

$$\tilde{Q}_1(u, v) = [\tilde{Q}_1(u_1, v) \ \tilde{Q}_1(u_2, v)]^{-1} = \begin{pmatrix} 0.0 & 0.6 & 1.0 \\ 0.0 & 0.6 & 0.9 \end{pmatrix}$$

$$\tilde{Q}_2(u, v) = [\tilde{Q}_1(u_1, v) \ \tilde{Q}_2(u_2, v)]^{-1} = \begin{pmatrix} 0.0 & 0.6 & 1.0 \\ 0.7 & 0.0 & 0.9 \end{pmatrix}$$

$$\tilde{Q}_3(u, v) = [\tilde{Q}_2(u_1, v) \ \tilde{Q}_1(u_2, v)]^{-1} = \begin{pmatrix} 0.7 & 0.0 & 1.0 \\ 0.0 & 0.6 & 0.9 \end{pmatrix}$$

$$\tilde{Q}_4(u, v) = [\tilde{Q}_2(u_1, v) \ \tilde{Q}_2(u_2, v)]^{-1} = \begin{pmatrix} 0.7 & 0.0 & 1.0 \\ 0.7 & 0.0 & 0.9 \end{pmatrix}$$

The complete solution is: $\begin{pmatrix} 0.0 & 0.6 & 1.0 \\ 0.0 & 0.6 & 0.9 \end{pmatrix} \subseteq Q \subseteq \begin{pmatrix} 0.7 & 0.6 & 1.0 \\ 0.7 & 0.6 & 0.9 \end{pmatrix}$

$$\begin{pmatrix} 0.0 & 0.6 & 1.0 \\ 0.7 & 0.0 & 0.9 \end{pmatrix} \subseteq Q \subseteq \begin{pmatrix} 0.7 & 0.6 & 1.0 \\ 0.7 & 0.6 & 0.9 \end{pmatrix}$$

$$\begin{pmatrix} 0.7 & 0.0 & 1.0 \\ 0.0 & 0.6 & 0.9 \end{pmatrix} \subseteq Q \subseteq \begin{pmatrix} 0.7 & 0.6 & 1.0 \\ 0.7 & 0.6 & 0.9 \end{pmatrix}$$

$$\begin{pmatrix} 0.7 & 0.0 & 1.0 \\ 0.7 & 0.0 & 0.9 \end{pmatrix} \subseteq Q \subseteq \begin{pmatrix} 0.7 & 0.6 & 1.0 \\ 0.7 & 0.6 & 0.9 \end{pmatrix}$$

A2.5 3. Given *a* and *b*, find *R*

Greatest Solution

Given the fuzzy sets $a \in \mathcal{V}$ and $b \in \mathcal{W}$ the *greatest* solution, $\hat{R} \in \mathcal{V} \in \mathcal{R}$, such that $a(v) ⊙ \hat{R}(v, w) = b(w)$, is:

$$\hat{R}(v, w) = a(v)^{-1} \circledast b(w)$$
$$= a(v)^{-1} \psi\, b(w)$$

Minimal Solutions

The union of the *minimal* solutions, $\tilde{R}_z \in \mathcal{V} \in \mathcal{R}$, such that $a(v) ⊙ \tilde{R}_z(v, w) = b(w)$, is:

$$\tilde{R}_z(v, w) = \Phi(a(v)^{-1}\, \beta\, b(w))$$
$$= \Phi([A(v)^{-1}\, \theta\, b(w)]\gamma[a(v)^{-1}\, \psi\, b(w)])$$
$$= \Phi([a(v)^{-1}\, \theta\, b(w)]\, \gamma\, [\hat{R}(v, w)])$$

where z is the cardinality of $\Phi(a(v)^{-1}\, \beta\, b(w))$.

Solution Example

Let $a(v) = [0.5 \quad 0.2 \quad 0.9]$ and $b(w) = [0.63 \quad 0.54 \quad 0.2]$ be given.
Find $\hat{R}$ and $\tilde{R}_z$.

Solution Maximum:

$$\hat{\boldsymbol{R}}(v, w) = \left[\begin{pmatrix} 0.5 \\ 0.2 \\ 0.9 \end{pmatrix} \Psi \begin{pmatrix} 0.63 & 0.54 & 0.2 \end{pmatrix}\right]$$

$$= \begin{pmatrix} 1.0 & 1.0 & 0.4 \\ 1.0 & 1.0 & 1.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

Solution Minimum:

$$\tilde{\boldsymbol{R}}_z(v, w) = \Phi\left[\begin{pmatrix} 0.5 \\ 0.2 \\ 0.9 \end{pmatrix} \beta \begin{pmatrix} 0.63 & 0.54 & 0.2 \end{pmatrix}\right]$$

$$= \Phi\left[\begin{pmatrix} 0.0 & 0.0 & 0.4 \\ 0.0 & 0.0 & 1.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix} \gamma \begin{pmatrix} 1.0 & 1.0 & 0.4 \\ 1.0 & 1.0 & 1.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}\right]$$

$$= \Phi\begin{pmatrix} 0.0 & 0.0 & 0.4 \\ 0.0 & 0.0 & 1.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

$$z = 1 \times 1 \times 3 = 3$$

$$\therefore \quad \tilde{\boldsymbol{R}}_1(v, w) = \begin{pmatrix} 0.0 & 0.0 & 0.4 \\ 0.0 & 0.0 & 0.0 \\ 0.7 & 0.6 & 0.0 \end{pmatrix}$$

$$\tilde{\boldsymbol{R}}_2(v, w) = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \\ 0.7 & 0.6 & 0.0 \end{pmatrix}$$

$$\tilde{R}_3(v,w) = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

The complete solution is:

$$\begin{pmatrix} 0.0 & 0.0 & 0.4 \\ 0.0 & 0.0 & 0.0 \\ 0.7 & 0.6 & 0.0 \end{pmatrix} \subseteq \boldsymbol{R} \subseteq \begin{pmatrix} 1.0 & 1.0 & 0.4 \\ 1.0 & 1.0 & 1.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

$$\begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \\ 0.7 & 0.6 & 0.0 \end{pmatrix} \subseteq \boldsymbol{R} \subseteq \begin{pmatrix} 1.0 & 1.0 & 0.4 \\ 1.0 & 1.0 & 1.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

$$\begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix} \subseteq \boldsymbol{R} \subseteq \begin{pmatrix} 1.0 & 1.0 & 0.4 \\ 1.0 & 1.0 & 1.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

A2.6 4. Given Q and T, find R

Greatest Solution

Given the fuzzy relations $\boldsymbol{Q} \in \mathcal{U}\times\mathcal{V}$ and $\boldsymbol{T} \in \mathcal{U}\times\mathcal{W}$, the *greatest* solution, $\hat{\boldsymbol{R}} \in \mathcal{V}\times\mathcal{W} \in \mathcal{R}$, such that $\boldsymbol{Q}(u,v) \odot \hat{\boldsymbol{R}}(v,w) = \boldsymbol{T}(u,w)$, is:

$$\begin{aligned} \hat{\boldsymbol{R}}(v,w) &= [\boldsymbol{Q}(u,v)^{-1} \circledpsi \boldsymbol{T}(u,w)] \\ &= \bigwedge_u [\boldsymbol{Q}(u,v)^{-1}\, \psi\, \boldsymbol{T}(u,w)] \end{aligned}$$

Minimal Solutions

The union of the *minimal* solutions, $\tilde{R}_z \in \mathcal{V} \times \mathcal{W} \in \mathcal{R}$, such that $Q(u, v) \odot \tilde{R}_z(v, w) = T(u, w)$, is:

$$\begin{aligned}
\tilde{R}_z(v, w) &= \bigvee_i [\Phi(Q(u_i, v)^{-1} \beta T(u_i, w))] \\
&= \Phi([Q(u, v)^{-1} \theta T(u, w)] \gamma [Q(u, v)^{-1} \circledast T(u, w)]) \\
&= \Phi([Q(u, v)^{-1} \theta T(u, w)] \gamma [\hat{R}(v, w)])
\end{aligned}$$

where z is the cardinality of $\Phi(Q(u_i, v)^{-1} \beta T(u_i, w))$.

Solution Example

Let $Q(u, v) = \begin{pmatrix} 0.5 & 0.2 & 0.9 \\ 0.6 & 0.5 & 0.8 \end{pmatrix}$ and $T(u, w) = \begin{pmatrix} 0.63 & 0.54 & 0.2 \\ 0.56 & 0.6 & 0.25 \end{pmatrix}$ be given.

Find $\hat{R}$ and $\tilde{R}_z$.

Solution Maximum:

$$\hat{R}(v, w) = \bigwedge_u \left[\begin{pmatrix} 0.5 & 0.6 \\ 0.2 & 0.5 \\ 0.9 & 0.8 \end{pmatrix} \Psi \begin{pmatrix} 0.63 & 0.54 & 0.2 \\ 0.56 & 0.6 & 0.25 \end{pmatrix} \right]$$

$$\hat{R}(u_1, v, w) = \begin{pmatrix} 1.0 & 1.0 & 0.4 \\ 1.0 & 1.0 & 1.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

$$\hat{R}(u_2, v, w) = \begin{pmatrix} 0.9333 & 1.0 & 0.4167 \\ 1.0 & 1.0 & 0.5 \\ 0.7 & 0.75 & 0.3125 \end{pmatrix}$$

$$\therefore \quad \hat{R}(v, w) = \bigwedge_u [\hat{R}(u_1, v, w), \hat{R}(u_2, v, w)]$$

$$= \begin{pmatrix} 0.9333 & 1.0 & 0.4 \\ 1.0 & 1.0 & 0.5 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

Solution Minimum:

$$\tilde{R}_z(v, w) = \Phi\left[\begin{pmatrix} 0.5 & 0.6 \\ 0.2 & 0.5 \\ 0.9 & 0.8 \end{pmatrix} \beta \begin{pmatrix} 0.63 & 0.54 & 0.2 \\ 0.56 & 0.6 & 0.25 \end{pmatrix}\right]$$

$$\tilde{R}_z(u_1, v, w) = \Phi\left[\begin{pmatrix} 0.5 \\ 0.2 \\ 0.9 \end{pmatrix} \beta \begin{pmatrix} 0.63 & 0.54 & 0.2 \end{pmatrix}\right]$$

$$= \Phi\left[\begin{pmatrix} 0.0 & 0.0 & 0.4 \\ 0.0 & 0.0 & 1.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix} \gamma \begin{pmatrix} 0.9333 & 1.0 & 0.4 \\ 1.0 & 1.0 & 0.5 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}\right]$$

$$= \Phi\begin{pmatrix} 0.0 & 0.0 & 0.4 \\ 0.0 & 0.0 & 0.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

$$z = 1 \times 1 \times 2 = 2$$

$$\therefore \quad \tilde{R}_1(u_1, v, w) = \begin{pmatrix} 0.0 & 0.0 & 0.4 \\ 0.0 & 0.0 & 0.0 \\ 0.7 & 0.6 & 0.0 \end{pmatrix}$$

$$\tilde{R}_2(u_1, v, w) = \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

$$\tilde{R}_z(u_2, v, w) = \Phi\left[\begin{pmatrix}0.6\\0.5\\0.8\end{pmatrix}\beta\begin{pmatrix}0.56 & 0.6 & 0.25\end{pmatrix}\right]$$

$$= \Phi\left[\begin{pmatrix}0.9333 & 1.0 & 0.4167\\0.0 & 0.0 & 0.5\\0.7 & 0.75 & 0.3125\end{pmatrix}\gamma\begin{pmatrix}0.9333 & 1.0 & 0.4\\1.0 & 1.0 & 0.5\\0.7 & 0.6 & 0.2222\end{pmatrix}\right]$$

$$= \Phi\begin{pmatrix}0.9333 & 1.0 & 0.0\\0.0 & 0.0 & 0.5\\0.7 & 0.0 & 0.0\end{pmatrix}$$

$$z = 2\times1\times1 = 2$$

$$\therefore \quad \tilde{R}_1(u_2, v, w) = \begin{pmatrix}0.9333 & 1.0 & 0.0\\0.0 & 0.0 & 0.5\\0.0 & 0.0 & 0.0\end{pmatrix}$$

$$\tilde{R}_2(u_2, v, w) = \begin{pmatrix}0.0 & 1.0 & 0.0\\0.0 & 0.0 & 0.5\\0.7 & 0.0 & 0.0\end{pmatrix}$$

For the complete solution of $\tilde{R}$ there a 4 possibilities: 2 ways to choose from $\tilde{R}_z(u_1, v, w)$ and 2 ways to choose $\tilde{R}_z(u_2, v, w)$. Thus:

$$\therefore \quad \tilde{R}_1(v, w) = \bigwedge_u \{\tilde{R}_1(u_1, v, w), \tilde{R}_1(u_2, v, w)\}$$

$$= \begin{pmatrix}0.9333 & 1.0 & 0.4\\0.0 & 0.0 & 0.5\\0.7 & 0.6 & 0.0\end{pmatrix}$$

$$\tilde{R}_2(v, w) = \bigwedge_u \{\tilde{R}_1(u_1, v, w), \tilde{R}_2(u_2, v, w)\}$$

$$= \begin{pmatrix} 0.0 & 1.0 & 0.4 \\ 0.0 & 0.0 & 0.5 \\ 0.7 & 0.6 & 0.0 \end{pmatrix}$$

$$\tilde{R}_3(v, w) = \bigwedge_u \{\tilde{R}_2(u_1, v, w), \tilde{R}_1(u_2, v, w)\}$$

$$= \begin{pmatrix} 0.9333 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.5 \\ 0.7 & 0.6 & 0.222 \end{pmatrix}$$

$$\tilde{R}_4(v, w) = \bigwedge_u \{\tilde{R}_2(u_1, v, w), \tilde{R}_2(u_2, v, w)\}$$

$$= \begin{pmatrix} 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.5 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

Clearly, it can be seen from these results that $\tilde{R}_2(v, w) \subseteq \tilde{R}_1(v, w)$ and $\tilde{R}_4(v, w) \subseteq \tilde{R}_3(v, w)$, therefore there are only 2 *minimums*, $\tilde{R}_2$ and $\tilde{R}_4$.

The complete solution is:

$$\begin{pmatrix} 0.0 & 1.0 & 0.4 \\ 0.0 & 0.0 & 0.5 \\ 0.7 & 0.6 & 0.0 \end{pmatrix} \subseteq \boldsymbol{R} \subseteq \begin{pmatrix} 0.9333 & 1.0 & 0.4 \\ 1.0 & 1.0 & 0.5 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

$$\begin{pmatrix} 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.5 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix} \subseteq \boldsymbol{R} \subseteq \begin{pmatrix} 0.9333 & 1.0 & 0.4 \\ 1.0 & 1.0 & 0.5 \\ 0.7 & 0.6 & 0.2222 \end{pmatrix}$$

REFERENCES

[1] Pappis, C.P., Sugeno, M., "Fuzzy Relational Equations and the Inverse Problem", *Fuzzy Sets and Systems*, Vol. 15, pp. 79-90, 1985.

[2] Pappis, C.P., "Multi-Input Multi-Output Fuzzy Systems and the Inverse Problem", *European Journal of Operational Research*, Vol. 28, pp. 228-230, 1987.

[3] Pappis, C.P., "Resolution of Cartesian Products of Fuzzy Sets", *Fuzzy Sets and Systems*, Vol. 26, pp. 387-391, 1988.

[4] Pappis, C.P., Adamopoulos, G.I., "A Computer Algorithm for the Solution of the Inverse Problem of Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 39, pp. 279-290, 1991.

[5] Pappis, C.P., Adamopoulos, G.I., "A Software Routine to Solve the Generalized Inverse Problem of Fuzzy Systems", *Fuzzy Sets and Systems*, Vol. 47, pp. 319-322, 1992.

[6] Di Nola, A., Pedrycz, W., Sessa, S., Pei-Zhuang, W., "Fuzzy Relation Equations Under a Class of Triangular Norms: A Survey and New Results", *Stochastica*, Vol. 8, pp. 99-145, 1984.

[7] Sanchez, E., "Resolution of Composite Fuzzy Relation Equations", *Information and Control*, Vol. 30, pp. 38-49, 1976.

[8] Sanchez, E., "Solutions in Composite Fuzzy Relation Equations: Applications to Medical Diagnosis in Brouwerian Logic", In: *Fuzzy Automata and Decision Processes*, Gupta, M.M., Saridis, G.N., Ragade, R.K., North-Holland, Amsterdam, pp. 221-234, 1977.

APPENDIX 3: RESULTS TABLES FOR CHAPTER 7

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
DiNola *et.al.* [1984]	*Prod*	*Max-prod*	0.7321	0.9668	0.6176	0.2647
		Max-min	1.0950	1.9050	0.5967	0.2944
	Min	*Max-prod*	0.7507	1.0020	0.7921	0.4201
		Max-min	0.8522	1.2220	0.8766	0.5369
Pedrycz [1984(a)]	*Prod*	*Max-prod*	0.5956	0.5450	0.5733	0.1558
		Max-min	0.5342	0.4993	0.6675	0.1603
	Min	*Max-prod*	0.6293	0.5947	0.7742	0.2135
		Max-min	0.5899	0.6378	1.1060	0.3978
Chen *et.al.* [1994]	*Prod*	*Max-prod*	0.6065	0.5658	0.6818	0.2236
		Max-min	0.5422	0.5089	0.7991	0.1705
	Min	*Max-prod*	0.6522	0.6298	0.8873	0.2834
		Max-min	0.6030	0.6635	1.1150	0.3886
Xu *et.al.* [1987]	*Prod*	*Max-prod*	0.5337	0.5076	0.5164	0.1503
	q=0.225	*Max-min*	0.5023	0.4717	0.5429	0.1396
	Min	*Max-prod*	0.6345	0.5749	0.7091	0.2065
	q=0.3	*Max-min*	0.6425	0.6308	0.9669	0.3737
Shaw *et.al.* [1992]	*Prod*	*Max-prod*	0.4959	0.3923	0.5549	0.1780
		Max-min	0.4390	0.3640	0.4988	0.1021
	Min	*Max-prod*	0.4678	0.3589	0.5000	0.1189
		Max-min	0.4844	0.4650	0.6522	0.1641
New Proposal	*Prod*	*Max-prod*	0.4732	0.3826	0.5314	0.1243
		Max-min	0.4433	0.3640	0.4878	0.1005
	Min	*Max-prod*	0.4301	0.3359	0.4557	0.0993
		Max-min	0.4535	0.4127	0.4911	0.1017

Table A3.1: Comparison of Batch Identification; (*R* unadjusted)

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
DiNola *et.al.* [1984]	*Prod*	*Max-prod*	0.8484	1.3290	0.5226	0.1916
	Min	*Max-min*	0.8435	1.3990	0.4702	0.1705
Pedrycz [1984(a)]	*Prod*	*Max-prod*	0.4813	0.3787	0.5395	0.1248
		Max-min	0.5155	0.4484	0.7514	0.2031
	Min	*Max-prod*	0.5616	0.4796	0.7878	0.2186
		Max-min	0.6096	0.6837	1.1890	0.4531
Chen *et.al.* [1994]	*Prod*	*Max-prod*	0.5162	0.4190	0.1513	0.6098
		Max-min	0.5515	0.4896	0.8679	0.2726
	Min	*Max-prod*	0.6321	0.5937	0.9709	0.3383
		Max-min	0.6577	0.7882	1.4050	0.6552
Xu *et.al.* [1987]	*Prod* q=0.225	*Max-prod*	0.4884	0.4339	0.4635	0.1156
		Max-min	0.5184	0.4957	0.5790	0.1639
	Min q=0.3	*Max-prod*	0.5605	0.4629	0.7108	0.2084
		Max-min	0.6594	0.6745	1.0240	0.4180
Shaw *et.al.* [1992]	*Prod*	*Max-prod*	0.4473	0.3545	0.4602	0.1098
		Max-min	0.4308	0.3683	0.5747	0.1388
	Min	*Max-prod*	0.4584	0.3547	0.5076	0.1202
		Max-min	0.4911	0.4768	0.6820	0.1797
New Proposal	*Prod*	*Max-prod*	0.4724	0.3835	0.5356	0.1263
		Max-min	0.4766	0.4128	0.5399	0.1272
	Min	*Max-prod*	0.4461	0.3681	0.4757	0.1087
		Max-min	0.5086	0.4972	0.5531	0.1313

Table A3.2: Comparison of Batch Identification; (*R* normalized)

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
Pedrycz [1984(a)]	*Prod*	*Max-prod*	0.6184	0.6141	0.5696	0.1689
		Max-min	0.5545	0.5647	0.6340	0.1548
	Min	*Max-prod*	0.6320	0.6426	0.7373	0.2081
		Max-min	0.6070	0.6636	1.013	0.3465
Chen *et.al.* [1994]	*Prod*	*Max-prod*	0.6416	0.6662	0.6613	0.2327
		Max-min	0.5730	0.5814	0.6735	0.1702
	Min	*Max-prod*	0.6447	0.6661	0.8233	0.2685
		Max-min	0.6093	0.6668	1.0120	0.3363
Xu *et.al.* [1987]	*Prod* q=0.225	*Max-prod*	0.6247	0.6301	0.5438	0.1786
	h=0.05; ε=1.0	*Max-min*	0.5189	0.4696	0.5112	0.1305
	Min q=0.3	*Max-prod*	0.6240	0.5641	0.7401	0.2242
	h=0.05; ε=1.0	*Max-min*	0.6249	0.6021	0.9766	0.3746
Shaw *et.al.* [1992]	*Prod*	*Max-prod*	0.5784	0.5495	0.5663	0.1871
		Max-min	0.5067	0.4748	0.5300	0.1193
	Min	*Max-prod*	0.5596	0.4970	0.5205	0.1309
		Max-min	0.5392	0.5246	0.6741	0.1803
New Proposal	*Prod*	*Max-prod*	0.5373	0.4674	0.5518	0.1370
		Max-min	0.4649	0.4114	0.5132	0.1160
	Min	*Max-prod*	0.5493	0.5142	0.5173	0.1263
		Max-min	0.5147	0.5105	0.5233	0.1189

Table A3.3: Comparison of On-Line Identification; (*R* unadjusted)

Author	ID Method	Prediction	J_1	J_2	Q_1	Q_2
Pedrycz [1984(a)]	*Prod*	*Max-prod*	0.5182	0.4590	0.5320	0.1314
		Max-min	0.5042	0.4499	0.7050	0.1889
	Min	*Max-prod*	0.5839	0.5602	0.7574	0.2126
		Max-min	0.5950	0.6704	1.1090	0.4077
Chen *et.al.* [1994]	*Prod*	*Max-prod*	0.5702	0.5409	0.5837	0.1496
		Max-min	0.5743	0.5455	0.8204	0.2548
	Min	*Max-prod*	0.6220	0.6318	0.8779	0.2802
		Max-min	0.6363	0.7433	1.2780	0.5465
Xu *et.al.* [1987]	*Prod* q=0.225 h=0.05; ε=1.0	*Max-prod*	0.5513	0.5015	0.4628	0.1220
		Max-min	0.5978	0.6174	0.5674	0.1757
	Min q=0.3 h=0.05; ε=1.0	*Max-prod*	0.5779	0.4957	0.7604	0.2368
		Max-min	0.6460	0.6532	1.1090	0.4900
Shaw *et.al.* [1992]	*Prod*	*Max-prod*	0.5435	0.5126	0.4873	0.1256
		Max-min	0.5135	0.4827	0.6019	0.1568
	Min	*Max-prod*	0.5467	0.4942	0.5327	0.1349
		Max-min	0.5477	0.5361	0.7026	0.1959
New Proposal	*Prod*	*Max-prod*	0.5373	0.4684	0.5544	0.1381
		Max-min	0.4927	0.4528	0.5595	0.1406
	Min	*Max-prod*	0.552	0.5219	0.5211	0.1273
		Max-min	0.5586	0.5888	0.5780	0.1482

Table A3.4: Comparison of On-Line Identification; (*R* normalized)

APPENDIX 4: *FUZZY* PREDICTIVE CONTROLLER ALGORITHM

The *fuzzy long range predictive controller* algorithm is composed of 5 parts:

(1) Initial calculations

(2) Identification

(3) Gain Input Calculations

(4) Dynamic Input Calculations

(5) Weighted Average Input Calculation

For this discussion of the controller design, define $u \in \mathcal{U}$ and $y \in \mathcal{Y}$ as the discrete input and output, respectively. And, let $u=\{u_i \mid i = \{1,2, ..., m\}\} \in \mathcal{U}$ and $y=\{y_l \mid l = \{1,2, ..., n\}\} \in \mathcal{Y}$ and be the *fuzzy* spaces of input and output, respectively, all defined on the finite fuzzy universes of discourses indicated.

So that the controller is not application specific, the error calculation made by the controller is determined and tested using *normalize* variables. Therefore:

$$\text{norm} = |y_{max} - y_{min}|$$

For this algorithm presentation the superscript, $\hat{}$, signifies an estimated variable. The overview of the controller algorithm is provided next:

(1) Initial Calculations: for $k > \tau$

(*i*) $\boldsymbol{K}_{k\text{-}1}(y(k+\tau), y(k+\tau-1)) = \boldsymbol{R}_{k\text{-}1} \circ \boldsymbol{u}(k-1)$

(*ii*) $\boldsymbol{K}_{k\text{-}2}(y(k+\tau-1), y(k+\tau-2)) = \boldsymbol{R}_{k\text{-}2} \circ \boldsymbol{u}(k-2)$

⋮

(τ) $\boldsymbol{K}_{k\text{-}\tau}(y(k+1), y(k)) = \boldsymbol{R}_{k\text{-}\tau} \circ \boldsymbol{u}(k-\tau)$

(2) Identification:

(*i*) $\boldsymbol{y}(k) = \text{fuzz}(\mathbf{y}(k))$

(*ii*) $\boldsymbol{R}_{k\text{-}\tau\text{-}1} = f_i(\boldsymbol{y}(k), \boldsymbol{y}(k-1), \boldsymbol{u}(k-\tau-1))$

(*iii*) If $|\mathbf{y}_{sp}(k) - \mathbf{y}(k)|/\text{norm} \leq \varepsilon$ where $0 < \varepsilon$ is a indication of controller tolerance

then $\boldsymbol{G}_{k\text{-}\tau\text{-}1} = f_i(\boldsymbol{y}(k), \boldsymbol{u}(k-\tau-1))$

where f_i is a predetermined identification algorithm

(3) Gain Input Calculations:

(*i*) $\tilde{\boldsymbol{y}}(k) = \boldsymbol{R}_{k\text{-}\tau\text{-}1} \circ \boldsymbol{y}(k\text{-}1) \circ \boldsymbol{u}(k\text{-}\tau\text{-}1)$

(*ii*) $\tilde{\mathbf{y}}(k) = \text{defuzz}(\tilde{\boldsymbol{y}}(k))$

(*iii*) $\tilde{\boldsymbol{y}}(k+\tau+1) = \boldsymbol{R}_{k\text{-}\tau\text{-}1} \circ \boldsymbol{y}(k) \circ \boldsymbol{u}(k\text{-}1)$

(*iv*) $\tilde{\mathbf{y}}(k+\tau+1) = \text{defuzz}(\tilde{\boldsymbol{y}}(k+\tau+1))$

(*v*) $\mathbf{lerr}(k) = \mathbf{y}(k) - \tilde{\mathbf{y}}(k)$

(*vi*) $\mathbf{perr}(k) = \mathbf{y}_{sp}(k) - \tilde{\mathbf{y}}(k)$

(*vii*) $\mathbf{ferr}(k) = \mathbf{y}_{sp}(k+\tau+1) - \tilde{\mathbf{y}}(k+\tau+1)$

(*viii*) $\mathbf{e}_s(k) = [\eta \cdot \mathbf{lerr}(k) + (1\text{-}\eta) \cdot \mathbf{perr}(k) - \mathbf{ferr}(k)]/\text{norm}$

where $0 \leq \eta \leq 1$ is a tuning parameter

(*ix*) $\mathbf{e}_g(k) = \sum_{i=k-\omega}^{k} \mathbf{e}_s(i)/(\omega+1)$

where $\omega \in \{0, 1, 2, \ldots\}$ is the window of error

(*x*) $\tilde{\mathbf{y}}_{sp}(k+\tau-1) = \mathbf{y}_{sp}(k+\tau-1) - \mathbf{e}_g(k)$

(*xi*) $\tilde{\boldsymbol{y}}_{sp}(k+\tau-1) = \text{fuzz}(\tilde{\mathbf{y}}_{sp}(k+\tau-1))$

(*xii*) $\boldsymbol{u}_{gain}(k) = \boldsymbol{G}_{k\text{-}\tau\text{-}1} \circ \tilde{\boldsymbol{y}}_{sp}(k+\tau-1)$

(*xiii*) $\mathbf{u}_{gain}(k) = \text{defuzz}(\boldsymbol{u}_{gain}(k))$

(4) Dynamic Input Calculations:

(*i*) $P_k(y(k+\tau), y(k)) = K_{k-1} \circ K_{k-2} \circ \ldots \circ K_{k-\tau}$

(*ii*) $\tilde{y}(k+\tau) = P_k \circ y(k)$

(*iii*) $Q_k(y(k+\tau+1), u(k)) = K_{k-\tau-1} \circ \tilde{y}(k+\tau)$

(*iv*) $u_r(1) = u_r(k-1)$

(*v*) For $i = 1$ to r where r is an arbitrary iteration value

(*a*) $\tilde{y}(k+\tau-1) = Q_k \circ u_r(i)$

(*b*) $\bar{y}(k+\tau-1) = \text{defuzz}(\tilde{y}(k+\tau-1))$

(*c*) $e_d(i) = y_{sp}(k+\tau-1) - \bar{y}(k+\tau-1)$

(*d*) If $e_d(i) \leq \varepsilon$ continue at (*ix*)

(*e*) $u_r(i) = f_s(u_r(i), e_d(i), \gamma)$ where $\gamma \geq 1$ is a tuning parameter

(*f*) $i = i + 1$, Repeat from (*v*)

(*vi*) $\tilde{y}(k+\tau-1) = Q \circ u_r(i)$

(*vii*) $\bar{y}(k+\tau-1) = \text{defuzz}(\tilde{y}(k+\tau-1))$

(*viii*) $e_d(i) = y_{sp}(k+\tau-1) - \bar{y}(k+\tau-1)$

(*ix*) $u_d(k) = u_r(i)$

(*x*) $\mathrm{u}_d(k) = \text{defuzz}(u_d(k))$

(*xi*) $\Delta \mathrm{u}_d(k) = \mathrm{u}_d(k) - \mathrm{u}(k-1)$

(*xii*) $a(k) = 0/1 + \beta|e_d(i)|$ where $0 \leq \beta \leq 1$ is a tuning parameter

(*xiii*) $\mathrm{u}_{dync}(k) = \mathrm{u}(k-1) + a(k) \cdot \Delta \mathrm{u}_d(k)$

where f_s is the shifting function defined separately below.

(5) Weighted Average Input Calculation:

(*i*) $\mathbf{u}(k) = \alpha \cdot \mathbf{u}_{dync}(k) + (1-\alpha) \cdot \mathbf{u}_{gain}(k)$ where $0 \leq \alpha \leq 1$ is a tuning parameter

(*ii*) $u(k) = \text{fuzz}(\mathbf{u}(k))$

(*iii*) $K_k(y(k+\tau+1), y(k+\tau)) = R_k \circ u(k)$

(*iv*) $k = k+1$

(*v*) repeat from (2)

The Adjusting Function, f_s is defined:

$u_r(i) = f_s(u_r(i), e_d(i), \gamma)$ $\gamma \geq 1$ is a tuning parameter for the convergence rate.

(*i*) adjfactor = $\gamma \cdot |e_d(i)|$

(*ii*) $uold = u_r(i)$ where $uold = \{uold_j \mid j = \{1,2, \ldots, m\}\} \in \mathcal{U}$

(*iii*) If R represents and inverse relationship process

$e_d(i) = - e_d(i)$

(*iv*) If $e_d(i) > 0$

(a) adj = adjfactor

(b) for $1 \leq j \leq m$

If $uold_j > 0$

$unew_j = uold_j -$ adj

If $unew_j < 0$

$unew_j = 0$

adj = $-(uold_j -$adj)

Else adj = 0

(c) adj = adjfactor

$uold = unew$

(d) for $m \geq j \geq 1$

If $uold_j > 0$

$unew_j = uold_j +$ adj

If $unew_j > 1$

$unew_j = 1$

adj = $(uold_j -$adj$)-1$

$unew_{j-1} = uold_{j-1} +$ adj

(e) $u_r(i) = unew$

(*v*) If $e_d(i) < 0$

execute an algorithm in the opposite sense to (*iv*)

APPENDIX : *FUZZY* PREDICTIVE CONTROLLER EXAMPLE

This appendix provides all the additional information required to reproduce the Shell Process Model [Cott, 1995] controller simulation from Section 10.5.4. Included in this appendix are:

(1) The dynamic relational matrix, ***R***

(2) The gain relational matrix, ***G***

(3) The input and output universes for the ***R*** matrix

(4) The input and output universes for the ***G*** matrix

(5) The fuzzification algorithm

(6) The defuzzification algorithm

The identification algorithm used for on-line identification in the controller algorithm is the Shaw *et al.* [1992] algorithm. The dynamic relational matrix, ***R***, and the gain relational matrix, ***G***, are learned prior to the simulation until the relational matrices are *complete* (~50,000 data points).

(1) The dynamic relational matrix, *R*

$$\boldsymbol{R}(y_1) = \begin{bmatrix} 0.0786 & 0.0036 & 0.0 & 0.0 & 0.0 \\ 0.1775 & 0.007 & 0.0 & 0.0 & 0.0 \\ 0.2045 & 0.0014 & 0.0 & 0.0 & 0.0 \\ 0.1934 & 0.0462 & 0.0 & 0.0 & 0.0 \\ 0.3941 & 0.2925 & 0.0001 & 0.0 & 0.0 \end{bmatrix}$$

$$\boldsymbol{R}(y_2) = \begin{bmatrix} 0.8768 & 0.3893 & 0.0365 & 0.0 & 0.0 \\ 0.8175 & 0.3890 & 0.0424 & 0.0 & 0.0 \\ 0.7946 & 0.5436 & 0.2918 & 0.0 & 0.0 \\ 0.8066 & 0.8164 & 0.5416 & 0.0001 & 0.0 \\ 0.6059 & 0.6864 & 0.4946 & 0.0007 & 0.0 \end{bmatrix}$$

$$
\boldsymbol{R}(y_3) = \begin{bmatrix} 0.0446 & 0.6013 & 0.5908 & 0.0301 & 0.0 \\ 0.0050 & 0.6033 & 0.7158 & 0.1910 & 0.0 \\ 0.0009 & 0.4550 & 0.6677 & 0.4933 & 0.0003 \\ 0.0 & 0.1375 & 0.4464 & 0.2932 & 0.0005 \\ 0.0 & 0.021\text{[illegible]} & 0.4802 & 0.3668 & 0.0017 \end{bmatrix}
$$

$$
\boldsymbol{R}(y_4) = \begin{bmatrix} 0.0 & \text{[illegible]}57 & 0.3725 & 0.6854 & 0.1213 \\ 0.0 & 0.0002 & 0.2418 & 0.7596 & 0.6226 \\ 0.0 & 0.0 & 0.0405 & 0.4676 & 0.4625 \\ 0.0 & 0.0 & 0.0121 & 0.6213 & 0.4701 \\ 0.0 & 0.0 & 0.0252 & 0.5802 & 0.5635 \end{bmatrix}
$$

$$
\boldsymbol{R}(y_5) = \begin{bmatrix} 0.0 & 0.0 & 0.0002 & 0.2845 & 0.8787 \\ 0.0 & 0.0 & 0.0 & 0.0493 & 0.3773 \\ 0.0 & 0.0 & 0.0 & 0.0391 & 0.5372 \\ 0.0 & 0.0 & 0.0 & 0.0854 & 0.5293 \\ 0.0 & 0.0 & 0.0 & 0.0523 & 0.4347 \end{bmatrix}
$$

(2) The gain relational matrix, *G*

$$G = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & .1134 & .2348 & .4116 & .5251 \\ 0 & 0 & 0 & 0 & 0 & 0 & .1995 & .3894 & .5114 & .3240 & .1739 \\ 0 & 0 & 0 & 0 & 0 & .2696 & .4933 & .3593 & .1708 & 0 & 0 \\ 0 & 0 & 0 & 0 & .2778 & .5364 & .2891 & .1159 & 0 & 0 & 0 \\ 0 & 0 & 0 & .2288 & .5179 & .2806 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .1262 & .4346 & .3268 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & .2957 & .4744 & .1351 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .1118 & .4953 & .2631 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .2433 & .5029 & .1147 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .4178 & .3366 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & .5159 & .2099 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(3) The input and output universes for the *R* matrix

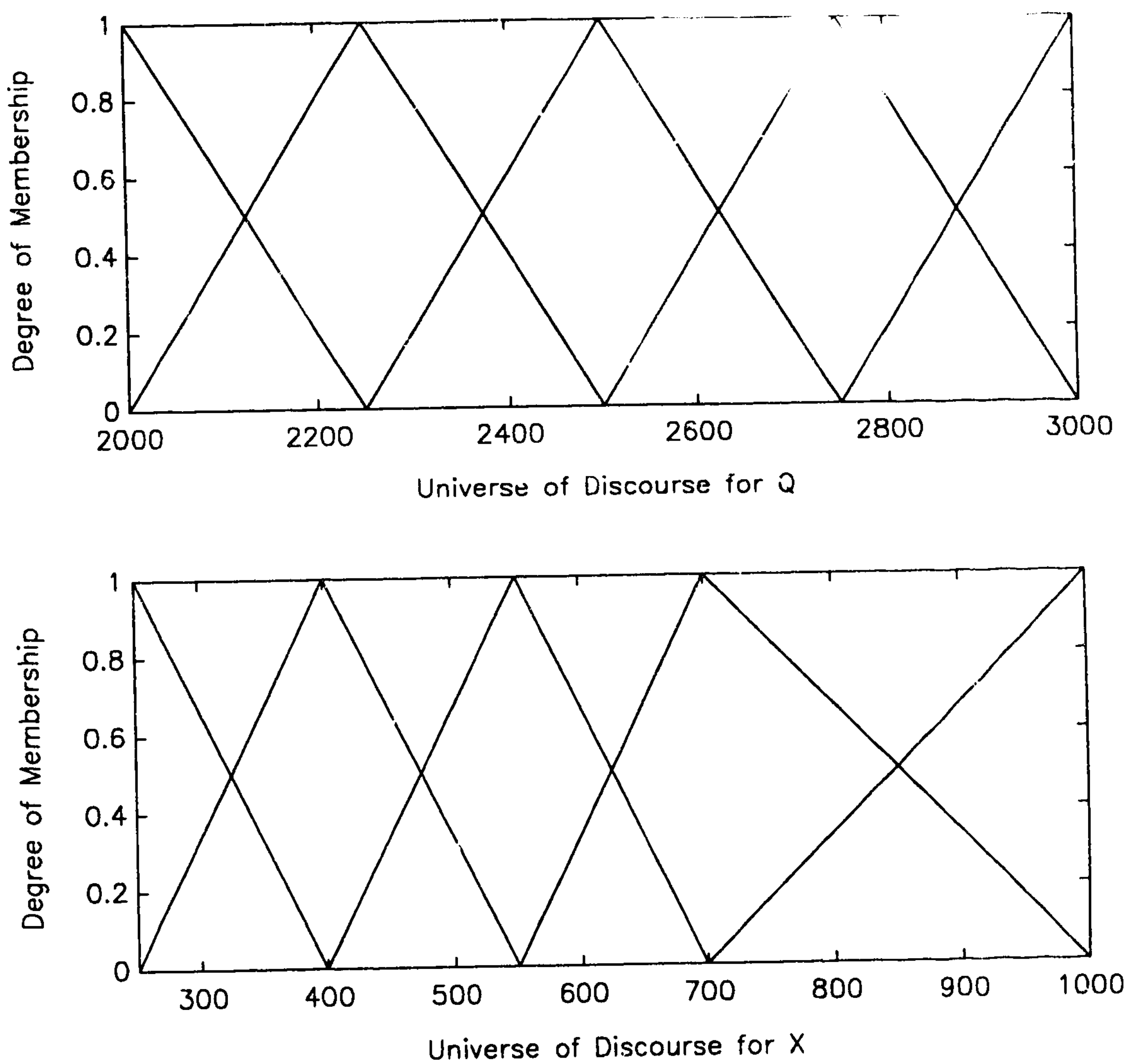
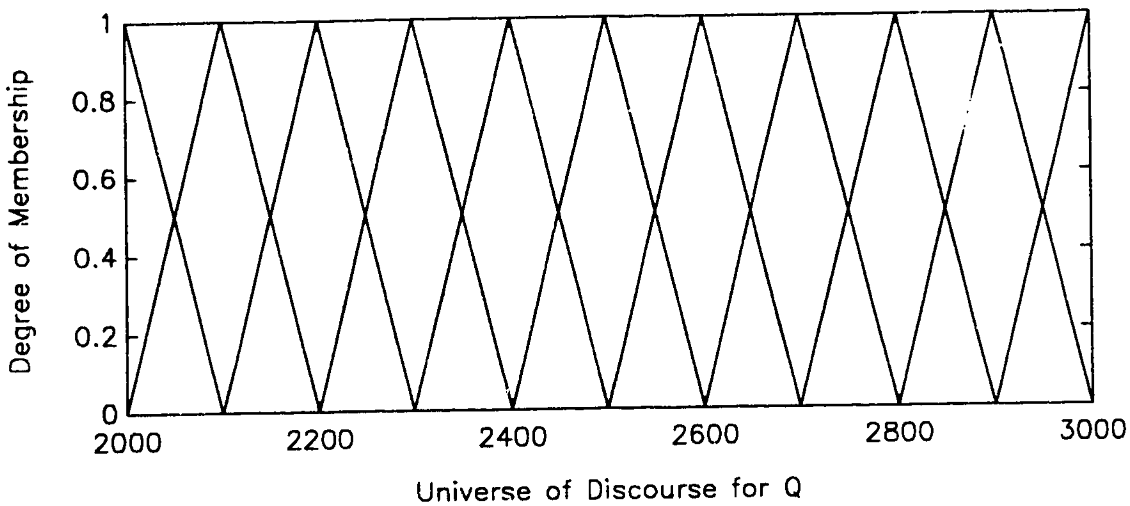


Figure A5.1: Universe of Discourse for *R*

(4) The input and output universes for the *G* matrix



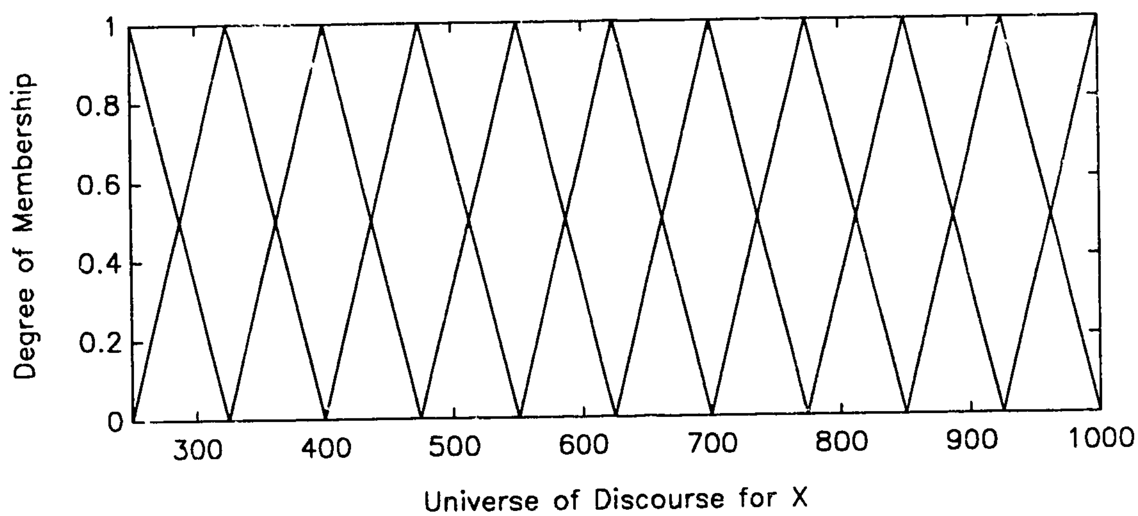


Figure A5.2: Universe of Discourse for *G*

(5) The fuzzification algorithm

The following fuzzification algorithms are written in MATLAB™ and assume that the fuzzy membership functions are either triangular or trapezoidal.

MATLAB™ M-File Listing: FUZZI.m

This program fuzzifies a discrete value. It calls the fuzzification algorithm p ram FUZZ.m

```
function muy = fuzzi(y,univy)

% fuzzifies a discrete value y
%
% Definitions:
%
%       muy - is the fuzzified value
%       y - is the discrete value
%       univy - is the corresponding universe of discou  e


n = length(y);

for i = 1:n,
   muy(i,:) = fuzz(y(i),univy);
end

.
```

MATLAB™ M-File Listing: FUZZ.m

This program is the fuzzification algorithm. The fuzzification algorithm assumes that the fuzzy membership functions are either triangular or trapezoidal.

```
function [muy] = fuzz(y,univy)

% calculates the fuzzy value muy from the universe univy
%
% Definitions:
%
%       muy - is the fuzzified value
%       y - is the discrete value
%       univy - is the corresponding universe of discourse
```

MATLAB™ M-File Listing: FUZZ.m (Cont'd)

```
[r,c] = size(univy);
U = univy;

if c == 4,                      %trapezoidal calculation
  for i = 1:r,
    if y < U(i,1),
      muy(i) = 0;
    elseif (y < U(i,2) & y >= U(i,1)),
      muy(i) = (y-U(i,1))./(U(i,2)-U(i,1));
    elseif (y < U(i,3) & y >= U(i,2)),
      muy(i) = 1;
    elseif (y <= U(i,4) & y >= U(i,3)),
      muy(i) = (U(i,4)-y)./(U(i,4)-U(i,3));
    elseif y > U(i,4),
      muy(i) = 0;
    end
  end
end

if c == 3,                      %triangular calculation
  for i = 1:r,
    if y < U(i,1),
      muy(i) = 0;
    elseif (y < U(i,2) & y >= U(i,1)),
      muy(i) = (y-U(i,1))./(U(i,2)-U(i,1));
    elseif (y <= U(i,3) & y >= U(i,2)),
      muy(i) = (U(i,3)-y)./(U(i,3)-U(i,2));
    elseif y > U(i,3),
      muy(i) = 0;
    end
  end
end
•
```

(6) The defuzzification algorithm

The following defuzzification algorithms are written in MATLAB™ and assume that the fuzzy membership functions are either triangular or trapezoidal.

MATLAB™ M-File Listing: DEFUZZI.m

This program defuzzifies the fuzzy value. It calls the defuzzification algorithm program DEFUZZ.m

```
function y = defuzzi(muy,univy)

% fuzzifies the discrete value y
%
% Definitions:
%
%        muy - is the fuzzified value
%        y - is the discrete value
%        univy - is the corresponding universe of discourse


[r,c] = size(muy);

for i = 1:r,
  y(i) = defuzz(muy(i,:),univy);
end

.
```

MATLAB™ M-File Listing: DEFUZZ.m

This program is the defuzzification algorithm. The defuzzification algorithm assumes that the fuzzy membership functions are either triangular or trapezoidal.

```
function [y] = defuzz(muy,univy)

% calculate the discrete value y from the universe univy
%
% Definitions:
%
%        muy - is the fuzzified value
%        y - is the discrete value
%        univy - is the corresponding universe of discourse
```

MATLAB™ M-File Listing: DEFUZZ.m (Cont'd)

```
[r,c] = size(univy);
U = univy;

if c == 4,                          %trapedoidal calculation
  for i = 1:r,
    if muy(i) > 0
      wcentroid(i) = ((1-muy(i))*(U(i,1)+U(i,4))+muy(i)*(U(i,2)+U(i,3))) ...
             ./2*muy(i);
    else
      wcentroid(i) = 0;
    end
  end
end

if c == 3,                          %triangular calculation
  for i = 1:r,
    if muy(i) > 0
      wcentroid(i) = ((1-muy(i))*(U(i,1)+U(i,3))+muy(i)*2*U(i,2)) ...
             ./2*muy(i);
    else
      wcentroid(i) = 0;
    end
  end
end

if sum(muy) > 0
  y = sum(wcentroid)./sum(muy);
else
  y = 0;
end
•
```

REFERENCES

[1] Cott, B.J., "Summary of the Process Identification Workshop at the 1992 Canadian Chemical Engineering Conference", *Journal of Process Control*, Vol. 5, pp. 109-113, 1995(b).

[2] Shaw, I.S., Krüger, J.J., "New Fuzzy Learning Model with Recursive Estimation for Dynamic Systems", *Fuzzy Sets and Systems*, Vol. 48, pp. 217-229, 1992.