Design and Analysis of Intelligent Fuzzy Tension Controllers for Rolling Mills

by

Jingrong Liu

A thesis

presented to the University of Waterloo

in fulfilment of the

thesis requirement for the degree of

Master of Applied Science

in

Electrical Engineering

Waterloo, Ontario, Canada, 2002

I hereby declare that I am the sole author of this thesis.

I authorize the University of Waterloo to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Jingrong Liu

I further authorize the University of Waterloo to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Jingrong Liu

The University of Waterloo requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

Abstract

This thesis presents a fuzzy logic controller aimed at maintaining constant tension between two adjacent stands in tandem rolling mills. The fuzzy tension controller monitors tension variation by resorting to electric current comparison of different operation modes and sets the reference for speed controller of the upstream stand. Based on modeling the rolling stand as a single input single output linear discrete system, which works in the normal mode and is subject to internal and external noise, the element settings and parameter selections in the design of the fuzzy controller are discussed.

To improve the performance of the fuzzy controller, a dynamic fuzzy controller is proposed. By switching the fuzzy controller elements in relation to the step response, both transient and stationary performances are enhanced.

To endow the fuzzy controller with intelligence of generalization, flexibility and adaptivity, self-learning techniques are introduced to obtain fuzzy controller parameters. With the inclusion of supervision and concern for conventional control criteria, the parameters of the fuzzy inference system are tuned by a backward propagation algorithm or their optimal values are located by means of a genetic algorithm. In simulations, the neuro-fuzzy tension controller exhibits the real-time applicability, while the genetic fuzzy tension controller reveals an outstanding global optimization ability.

Acknowledgements

My deepest gratitude goes to my supervisor, Professor Farrokh Janabi-Sharifi, for his outstanding guidance and invaluable advice throughout my graduate studies. Also deserving my heartfelt gratitude is the insightful comments and suggestions from my co-supervisor Professor William J. Wilson.

I have benefited immensely in my studying at the University of Waterloo. Inspiring working conditions at UW also greatly facilitated the writing of this thesis. I am especially indebted to Ms. Wendy Boles, Elaine Garner and Audrey Sloboda for helping me with administrative issues.

I have enjoyed the very pleasant collaboration with resourceful colleagues at QUAD. Dr. Jalal Biglou's patience, Dr. Jian Fan's academic inspiration, Robert Majka's constructive suggestions and their friendship are highly appreciated. A very special thanks to Leon Winitsky for his invaluable comments.

Without financial support from National Sciences and Engineering Research Council of Canada and Materials and Manufacturing Ontario, this program would not have been possible. I would like to thank QUAD Engineering Inc. for giving me this opportunity.

Finally, I dedicate this undertaking to my parents and my brother. Thanks for their encouragement during the past two years.

Contents

List of Figures

List of Tables

Notations

B.1 Signal Part

X_i — Linguistic variable X of ith input signal

Y_j — Linguistic variable Y of jth output signal

x_i — Measurement of ith input signal

y_j — Value of jth output signal

i_t — Target current given as reference for tension control system

i_d — Desired current generated from tutor

i_a — Actual current output from rolling mill stand

B.2 Fuzzy Logic Part

FLC — Fuzzy Logic Controller

i_e — Current error

Δi_e — Current error variance

Δv_f — Speed reference variance from FLC

MF — Membership Function

U — Universe of discourse for fuzzy variable

SF — Scaling Factor

WO — whole overlap of fuzzy variables

$\mu_F(f)$ — Degree of membership for value f in fuzzy variable F

NB — Fuzzy variable negative big and parameter for fuzzy singleton

NM	Fuzzy variable negative medium and parameter for fuzzy singleton
NS	Fuzzy variable negative small and parameter for fuzzy singleton
N	Fuzzy variable negative and parameter for fuzzy singleton
Z	Fuzzy variable zero and parameter for fuzzy singleton
P	Fuzzy variable positive and parameter for fuzzy singleton
PS	Fuzzy variable positive small and parameter for fuzzy singleton
PM	Fuzzy variable positive medium and parameter for fuzzy singleton
PB	Fuzzy variable positive big and parameter for fuzzy singleton
a_F	Parameter a of the fuzzy variable F with triangle/trapezoid MF
b_F	Parameter b of the fuzzy variable F with triangle/trapezoid MF
c_F	Parameter c of the fuzzy variable F with triangle/trapezoid MF
d_F	Parameter d of the fuzzy variable F with trapezoid MF
X^r	Fuzzy variable X of input linguistic variable in rule r
Y^r	Fuzzy variable Y of output linguistic variable in rule r
X'	Fuzzy variable X of input linguistic variable in fact part
Y'	Fuzzy variable Y of output linguistic variable in conclusion part
$\rightarrow$	Implication operation
$\wedge$	Min operation
$\vee$	Max operation
$\bullet$	Product operation
DFC	Dynamic Fuzzy Controller

B.3 Backward Propagation Part

NFC	Neuro-Fuzzy Controller
$P(x_1, x_2)$	Approximate performance index with input x_1 and x_2
e	Error
x	Neural network input
y	Neural network output

y^t	Target System output
y^a	Actual System output
y_i^l	Output of node i in layer l
x_i^l	Input of node transfer function i in layer l
$^-$	Average operation
$\times$	Linear operation
s	Scaling factor as weight
m	Parameter of membership function as weight
r	Rule weight
J_i^l	Sensitivity of node i in layer l
J	Sensitivity of Model

B.4 Genetic Algorithm Part

GFC	Genetic Fuzzy Controller
$F(x_1, x_2)$	Fitness with input x_1 and x_2
L_{sf}	Gene of scaling factor for linguistic variable L
F_p	Gene of membership function parameter p for fuzzy variable F
Ri_w	Gene of ith rule weight

Chapter 1

Introduction

1.1 Motivation

Tandem rolling mills usually consist of a number of mill stands arranged in alignment (Figure 1.1.1). Long metal products with different cross-sections, such as strips or bars, are produced based on the principle of multistage shaping as they proceed through mill stands sequentially. The cross-sections of workpieces, such as blooms, billets or slabs, are reduced in each stand under high pressure [1].
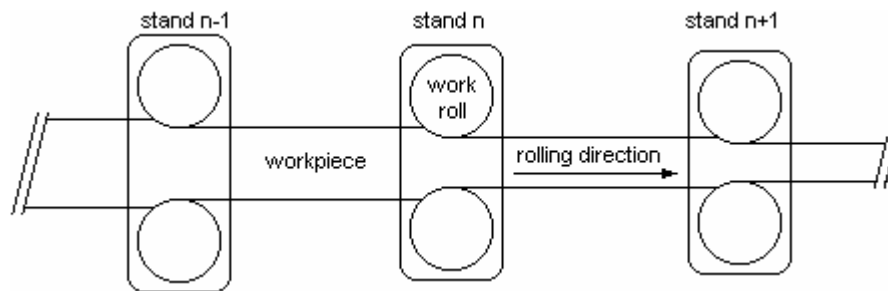
Figure 1.1.1   Tandem Rolling Mills

To meet the dimension requirements, such as thickness, width, flatness and shape, automatic gage controllers (AGCs) are employed to control the roll gap and pressure. Automatic speed regulators (ASRs) are used to control the mass flow passing the rolling mills. A single stand in tandem rolling mills with

dimension and yield regulation systems is schematically illustrated in Figure 1.1.2.
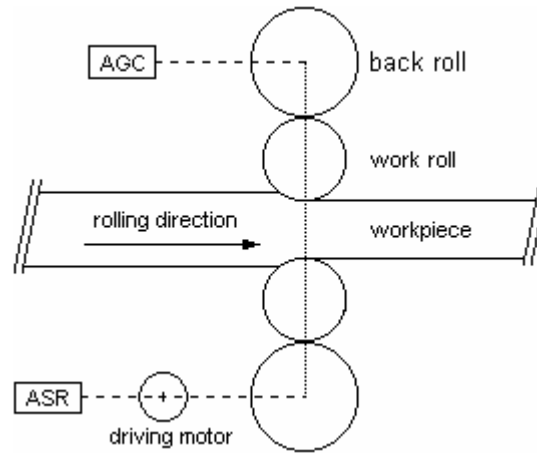
Figure 1.1.2   Rolling Mill Stand

According to the rolling processes of a workpiece, the operation modes of a stand fall into four categories [2]. Initially, no workpiece passes and the stand n works in an idle mode as shown in Figure 1.1.3. When a workpiece passing the stand n is engaged on the upstream stand n-1, but has not entered the downstream stand n+1 yet, stand n is in a run-in mode as shown in Figure 1.1.4. Following that is the normal mode in which the workpiece passing stand n is engaged on both the upstream stand n-1 and the downstream stand n+1 (Figure 1.1.5). The run-out mode refers to the state when the workpiece passing stand n is engaged on the downstream stand n+1 and leaves the upstream stand n-1 (Figure 1.1.6).
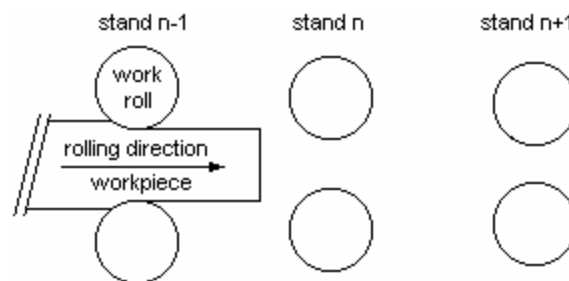
Figure 1.1.3   Idle Mode
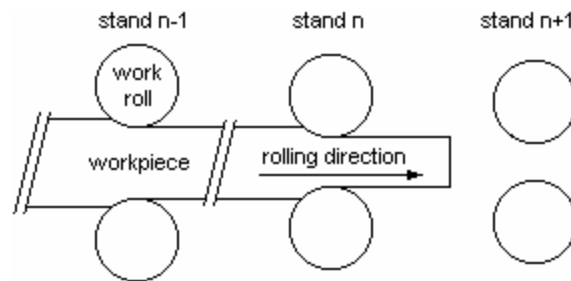
Figure 1.1.4   Run-in Mode
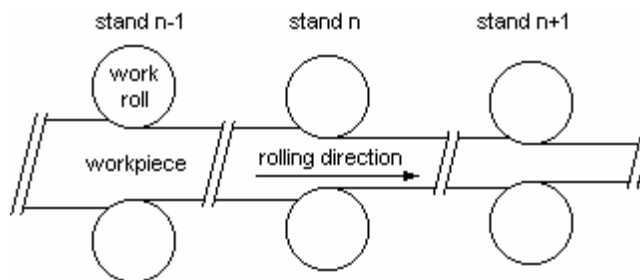
Figure 1.1.5   Normal Mode
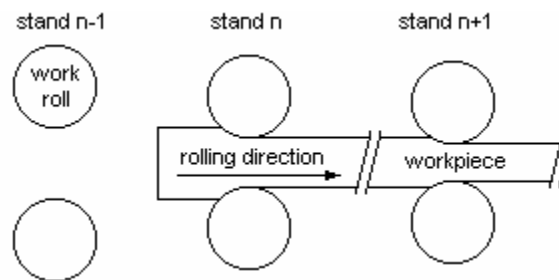
Figure 1.1.6   Run-out Mode

A specific problem associated with tendem rolling mills is the presence of tension, a longitudinal force inside the workpiece resulting from the inequality of mass flow of the rolled material between two adjacent mill stands. Tension can cause undesired product deformation such as cross-sectional reduction or cobbling. On the other hand, to optimize the performance of AGC and ASR, it is desirable to keep tension constant by means of addtitional control action. However, interaction effects, *i.e.*, activities of AGC and ASR, such as adjusting

the roll gap and stand speed, will incur tension variation and in turn tension maintenance activity, such as adjusting stand speed, will worsen the gage and speed control, and will complicate the situation.

1.2 Literature Survey

Some conventional tension control systems, such as the full stand loopless PID controller, optimum looper controller [4] and minimum tension controller [5], have been used extensively. However, these controllers did not take the interaction effects into consideration and the design and adjustment are difficult despite their high performance.

In view of the interaction that exists between the tension, dimension and massflow control systems, endeavors have been made to deal with this coupling problem by feed forward compensation or developing multivariable cross controllers [6, 7 and 8].

1.2.1 General Control Schemes

Tension control methods heavily depend on the way of obtaining tension data. Tension can be measured directly using a tension meter or can be inferred indirectly from the measurement or estimation of the torque, material contact length, roll radius and rolling bite angle. In general, Non-Interfering Tension Control (NIC) can make feedback and interaction compensation easy, and further, the response quick; however, its usage relies on the precision of instruments and rolling conditions.

To overcome the drawbacks associated with a multivariable controller, (*i.e.*, choosing a weight matrix and solving the Riccati equation), Y. Kadoya [9] applied an inverse linear quadratic (ILQ) control design theory. In their

experiments in cold tandem rolling, the tension variation was eliminated, while the accuracy of the product dimension was improved.

The repetitive fashion of the rolling process allows the application of iterative learning controller which can track reference values asymptotically by examining the periodic cycle. In a mill acceleration simulation [10], the learning controller demonstrated its efficiency in tension/gage control with the roll-bite friction as a disturbance.

Torque disturbance estimation observer [11] also facilitated the interaction compensation and loosened the requirement of a precise model.

1.2.2 Looper Tension Control

A workpiece looper driven by a deflector, such as a electric motor, pneumatic cylinder or hydraulic cylinder, as shown in Figure 1.2.1, is a mechanism that elevates the strip between two adjacent stands above pass line. Tension data can be indicated by the looper height, inferred from the load of the deflector or measured by load cells and accelerometers.
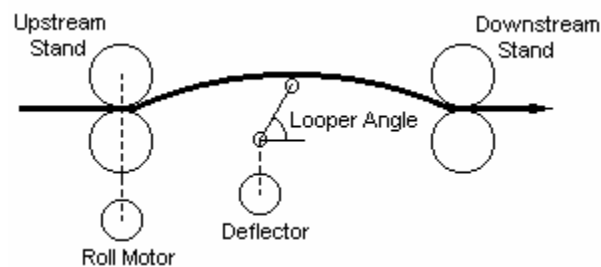
Figure 1.2.1 Looper

Some control schemes for keeping tension constant by means of loopers are enumerated as follows:

(1)   Applying optimal regulator theory.

Y. Seki *et al* [12] designed an optimum regulator for tension control using the speed and current of both the roll and looper driving motor as well as the looper angle as feedback variables. Based on the optimum regulator, AGC and looper control can also be integrated to achieve multivariable control.

H. Miura *et al* [13] also presented the application of this theory to gauge and tension control in hot strip finishing rolling mills.

(2)   Applying H-infinity theory.

Imanari *et al* [14] reported the embodiment of an H-infinity controller in the looper tension control. They also, with practical control data, made a valuable comparison between the conventional PI control, NIC, and H-infinity control. By resorting to adjusting the looper angle co-operatively instead of keeping it constant tightly, their control system effectively reduced strip tension fluctuation in finishing mills. The systems showed quick response and robust stability against skid marks, which are the sources of the main disturbance in hot strip rolling process.

In the simulation of bar rolling [15], robust stability and sensitivity reduction against such disturbances as skid marks and variations in the speed coefficient resulting from varying rolling conditions were achieved.

Another application of H-infinity theory is the disturbance compensator for suppressing the disturbance due to the interaction effects, which was developed by M. Shioya *et al* [16].

(3)   Applying back-stepping-algorithm.

T. Hesketh *et al* [17] applied a back-stepping algorithm experimentally based on recursive non-linear techniques to hot strip finishing mills. They

also argued the advantage of output feedback over observer feedback in the PLC implementation.

The application of the looper tension control is restricted to flexible workpieces at the intermediate and finishing stands with enough inter-stand distance such as a hot strip finishing rolling.

1.2.3   Load Comparison

In the electric current comparison method, the measurement of the motor armature current, which is proportional to the rolling torque, serves as the tension indicator. The set point of the current is captured in the run-in mode and compared with the one in the normal mode; based on the divergence, the controller adjusts the motor speed to maintain the tension. In the run-out mode, the controller locks on a rigid speed control to avoid reacting to the tension disappearing on the upstream side. This method is an effective start-up tool instead of control tool in the normal mode since the speed ratios determined in the run-in mode are maintained for the whole run.

N. Hur [18] proposed a load-sharing control scheme in which one stand and its downstream stand were treated as a master and a slave respectively. By modelling the tension as the output of a first–order filter driven by the speed difference between two adjacent stands, their scheme exhibited superior decoupling effect in simulations.

In another decoupling scheme proposed in [19], the rolling speed controller uses the torque sum, while the tension controller uses the torque difference between two adjacent stands as references respectively.

In the dynamic multivariable free tension controller developed by H. Ogai *et al* [20], tension was captured as the load variation between two adjacent workpiece

rolling cycles; the speed reference is derived by a set-up system based on fuzzy inference.

The electric current comparison method is susceptible to disturbances such as temperature and material deformation resistance. A complex supervisory network is necessary to set the motor speed-ratio in the normal mode, taking many parameters into consideration, e.g., mill delivery speed, roll barrel diameter, pass form factor, roll-gap, pass reduction and mill gear ratio.

1.2.4 Quotient Control Method

In this method, the tension is represented as the quotient of the rolling torque to the rolling force. The quotient acquired in the run-in mode indicates a tension-free state and is stored as the set point. In the normal mode, according to the quotient variation, the minimum inter-stand tension is maintained by adjusting the motor speed based on the mathematical model known as Sims equations [3].

Apart from the requirement of the stand-dependent measurement of the rolling force, this method is complicated and its application is rolling conditions related.

1.2.5 Forward Slip Method

Forward slip results when the workpiece exits at a higher speed than the roll peripheral speed. It can be expressed as:

$$f = \frac{V_W - V_R}{V_R} \times 100\%$$

(1-2-1)

Where:

V_W :   is the workpiece exit speed.

V_R :   is the roll peripheral speed.

The region where forward slip occurs is depicted in Figure 1.2.2.
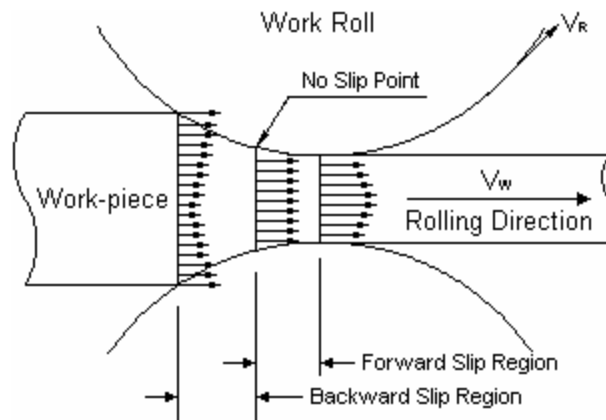
Figure 1.2.2   Forward Slip

Since forward slip is mainly affected by inter-stand tension, it can be used as a tension indicator. The control mechanism based on forward slip is similar to the quotient method as discussed in the previous subsection.

The deficiency of the forward slip method is that this scheme heavily relies on the accurate measurement of the material speed.

1.3 Contribution of the Thesis

The aforementioned tension control schemes need measurements of tension directly by delicate instruments or inferring the measurement indirectly from sophisticated formulae. The success of the tension control relies on the availability of an exact rolling mathematical depiction and is susceptible to noise. On the other hand, most of the applications are limited to parts of the rolling section such as roughing, intermediate or finishing rolling mills.

In practice, human operators can manage the rolling for the whole mill line under uncertain conditions. By monitoring a few state variables, such as the looper

height or the current of the stand driving motor, tension can be maintained at an acceptable level without much knowledge of the stands' physical models. This thesis aims at developing a tension controller that works with incomplete rolling mathematical models, inexact measurements and can maintain the tension under most rolling conditions. This research is the continuation of ongoing research ([2] and [22]) for the development of robust tension controllers in rolling mills led by Professor Janabi-Sharifi at Ryerson University and University of Waterloo and incorporated with national steel industries. Based on previous work on looper control, the group currently examines the applications of neuro-fuzzy intelligent control techniques to achieve tension-free rolling. The contributions of this thesis include the development and examination of the tension controller based on fuzzy logic, and self-tuning (or self-organization) techniques for on-line and off-line tuning of the tension control system.

1.3.1 Tension Controller Design

For the purpose of eliminating the tension variation in the normal mode, a fuzzy logic controller (FLC), which performs human reasoning, is superimposed on each rolling stand's ASR to adjust its reference speed during the normal mode. In this scheme, the armature current of the stand driving motor is used as a rough indicator of tension between the controlled and its downstream stands. From this point on, the terms tension and current will be used interchangeably. The nominal value of the target tension is sampled when the rolling process of a controlled stand enters the run-in mode. During its normal mode, this target tension is set as the reference for the FLC and compared with the feedback of actual tension to derive the tension variation. The input signals of the FLC are input1 $x_1 = i_e$, *i.e.*, the current error between the target and actual one, and input2 $x_2 = \Delta i_e$, *i.e.*, current error variance. The output signal of the FLC is output $y = \Delta v_f$, *i.e.*, the speed reference variance. By resorting to adjusting the reference for the motor speed controller ASR according to the difference between the

actual and target current, the FLC is expected to remove tension fluctuation during normal mode (Figure 1.3.1).
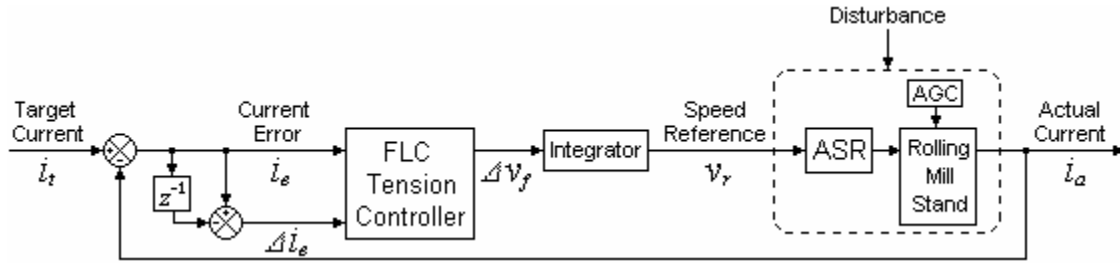
Figure 1.3.1   Fuzzy Tension Control System

In this study, the simulations of the system and FLC are conducted in Matlab$^{\circledR}$ 5.3. To delve into the various facets and variations of the FLC before an actual experimental verification, a discrete linear dynamic system is used as a black box model of the rolling mill stand working in the normal mode, which is identified by means of a generalized least-squares method. The sampling of target current i_t during the run-in mode is simulated by a signal generator block.

The influence of the control parameters of the FLC, including (de)fuzzification methods and aggregation/implication operations, will be investigated based on the simulation results.

Based on the advantages of control elements of the FLC, a straightforward variation of the FLC known as Dynamic Fuzzy Controller (DFC) is proposed. According to the performance of the FLC for a unit step response of current and the definition of trigger conditions, the control elements, such as the scaling factors for input/output, the aggregation operation of antecedents, the implication operation and the defuzzification method, are changed during control activity to improve transient and steady performances.

1.3.2 Self-organization Technique application

Following the analysis and design of the FLC, two self-organization techniques are applied to deal with the sophisticated parameters. In this study, the system training and/or evolving is facilitated using first/second order transfer functions in the frequency domain as a tutor or natural environment. This will generate the desired tension behavior concurrently accompanying the actual system output.

One self-learning scheme, known as tuning strategy, is the backward-propagation-algorithm-based fuzzy controller, *i.e.*, neuro-fuzzy controller (NFC). In the NFC, the fuzzy inference system is viewed as a neural network with the control parameters as adjustable weights and the fuzzy reasoning is realized by a forward propagation algorithm driven by the input signals. Guided by supervision from the tutor, the fuzzy controller imitates the human brain nerve net to learn the best FLC parameters around nominal values by means of a backward propagation algorithm driven by the error between the desired and actual tension. In this work, the detailed procedure of the forward propagation for the control action and the backward propagation for the error are provided. The tuning ability of this algorithm is tested in both offline applications with a unit step target signal and online applications with a trapezoidal target signal.

The other self-learning scheme, known as searching strategy, is the genetic-algorithm-based fuzzy controller (GFC). In the GFC, the fuzzy inference system is viewed as organism characterized by, as genes, the control parameters. Via emulation, the fuzzy controller imitates species evolution to pick the optimal FLC parameters from the whole control space. In this study, the gene coding and circumstantial evolution process are supplied. The searching capability of this algorithm is examined for a unit step response.

In both tuning and searching strategies, the principles and methods to preserve logical parameters of the FLC during the self-learning are discussed. The

measures taken here for parameter protection and restriction can prevent the emergence of parameters that either violate intuition or obstruct the execution of the fuzzy inference.

1.4 Structure of the Thesis

This thesis is composed of seven chapters; the overall organization is as follows:

First, the derivation and embodiment of the model for a rolling mill stand on which the simulations will base in this study is given at the beginning of chapter 2. This chapter deals with the detailed elements selection issues in the design of the fuzzy logic controller. Continuing from previous discussion, the dynamic fuzzy tension controller is presented in Chapter 3. Fundamental frame for the application of self-organization techniques is provided in Chapter 4. The paradigms used for self-learning are introduced as well. Chapter 5 is devoted to the offline and online applications and simulations of the backward propagation algorithm. Chapter 6 is concerned with the application and simulation of the genetic algorithm. The conclusions and future work are discussed in Chapter 7.

Chapter   2

Fuzzy Tension Controller

Conventional tension control schemes for rolling mills need exact mathematical models and complete knowledge of rolling processes. However, it is difficult to identify rolling processes from the measurement of tension data because of the complicated characteristics of rolling mills, noisy environments and lack of delicate instruments. Moreover, the interaction between the control actions for tension, workpiece gage and mass flow speed deteriorates the performance of the tension controller. Multivariable controllers based on advanced control theories, despite taking interaction effects into consideration, are difficult to implement and configure.

On the other hand, human reasoning in the form of linguistic language can manipulate the rolling stands and manage the inter-stand tension satisfactorily. By imitating the human intelligence, fuzzy logic can deal with ambiguities in the rolling processes [21]. In a dynamic system, a fuzzy-logic-based controller can work with incomplete plant knowledge and inexact information by formalizing expert knowledge as fuzzy IF-THEN rules. Motivated by the advantages of the fuzzy logic control, F. Janabi-Sharifi [22] has recently proposed the use of the FLC in tension control for rolling mills. However, the initial work has been limited to looper control schemes. In this work, inspired by these merits and the competence of human operators, an attempt is made to apply the fuzzy logic to the tension control for the complex nonlinear rolling processes.

2.1 Model of Rolling Mill Stand

2.1.1 Motivation

Although the fuzzy inference system is a model-free numerical estimator, for the purpose of simulating and evaluating the FLC before putting it into practice, the dynamical model of the rolling mill stand working in the normal mode is needed. To put emphasis on prediction and control, a black-box model applicable to generic estimation techniques and with simple structure and standard form is preferred. The resultant model is evaluated based on such criteria as fitness of the predicted output, agreement of actual and predicted characteristics.

2.1.2 Assumption

Based on the purposes and observations, it is assumed that the dynamics of the stand n (Figure 1.1.2) is:
- Causal
- Lumped
- Linear
- Time invariant in each mode
- Finite-order
- Single input single output (SISO)

  Input signal u(t) is the driving motor speed RPM of the stand.

  Output signal y(t) is the driving motor current AMP of the stand.
- Rolling mill dynamics can be sectioned by each single stand after the introduction of tension between two adjacent stands.

Then, one general discrete-time, parametric, and stochastic model possessing the aforementioned attributes can be expressed as a linear difference equation [23] in time domain in the form of:

$$A(q)\,y(t) = \frac{B(q)}{F(q)}u(t - nk) + \frac{C(q)}{D(q)}e(t) \qquad (2\text{-}1\text{-}1)$$

where:

$u(t)$:   input signal RPM, $t = 1, 2, \dots$.

$y(t)$:   output signal AMP, $t = 1, 2, \dots$.

$e(t)$:   measurement noise.

q: delay operator.

$A(q) = 1 + a_1 q^{-1} + \dots + a_{na} q^{-na}$.

$B(q) = b_0 + b_1 q^{-1} + \dots + b_{nb} q^{-nb+1}$.

$C(q) = 1 + c_1 q^{-1} + \dots + c_{nc} q^{-nc}$.

$D(q) = 1 + d_1 q^{-1} + \dots + d_{nd} q^{-nd}$.

$F(q) = 1 + f_1 q^{-1} + \dots + f_{nf} q^{-nf}$.

2.1.3   Observation

Generally, rolling mills work in harsh noisy environments. Some noise sources can be listed in Table 2.1.1 [3].

From mills	From control systems	From workpiece
▪ Mill chatter ▪ Roll eccentricity, ovality, crown, wear, flattening, expansion or contraction ▪ Roll thermal expansion/ contraction ▪ Roll bearing oil film thickness variation ▪ Roll bite lubricant film thickness variation ▪ Roll gap screw-down extension, hydraulic cylinder extension ▪ Motor speed error	▪ Roll speed controller ▪ Roll gap controller ▪ Roll force, bending, balance controller ▪ Roll cool and lubricant control ▪ Gauge monitor control ▪ Instrument error ▪ Controller Parameter variation	▪ Temperature ▪ Hardness ▪ Gauge, width, profile and flatness ▪ Skid marks ▪ Strip lateral spread ▪ Coefficient of friction

Table 2.1.1   Sources of Noise

For identification, the Sidbec bar mills (Contrecoeur, Québec) are used to collect tension data. To maintain the tension between stand 7 and 8 when a billet passes through both stands and stand 7 works in the normal mode, the current AMP, as the output signal , and the speed RPM, as the input signal, of the driving motor

of stand 7 are recorded at frequency 20Hz, *i.e.*, the sampling interval is 0.05 second.

2.1.4 Identification and Implementation

The data sequence for identification are collected when a billet passes through stand 7, which working in the normal mode. The AMP and RPM are recorded and profiled in Figure 2.1.1. To evaluate the resulting model, another group of AMP/RPM data are collected (illustrated in Figure 2.1.2) as another billet passes through stand 7 during the normal mode.
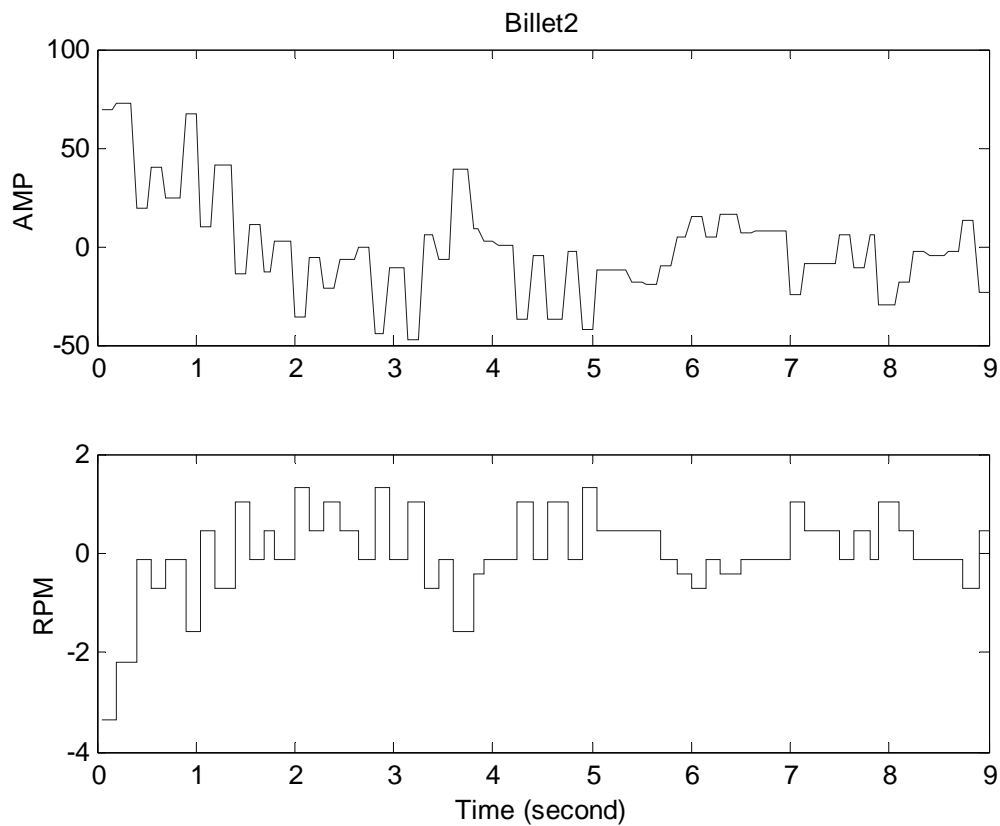
Figure 2.1.1   Data for Identification
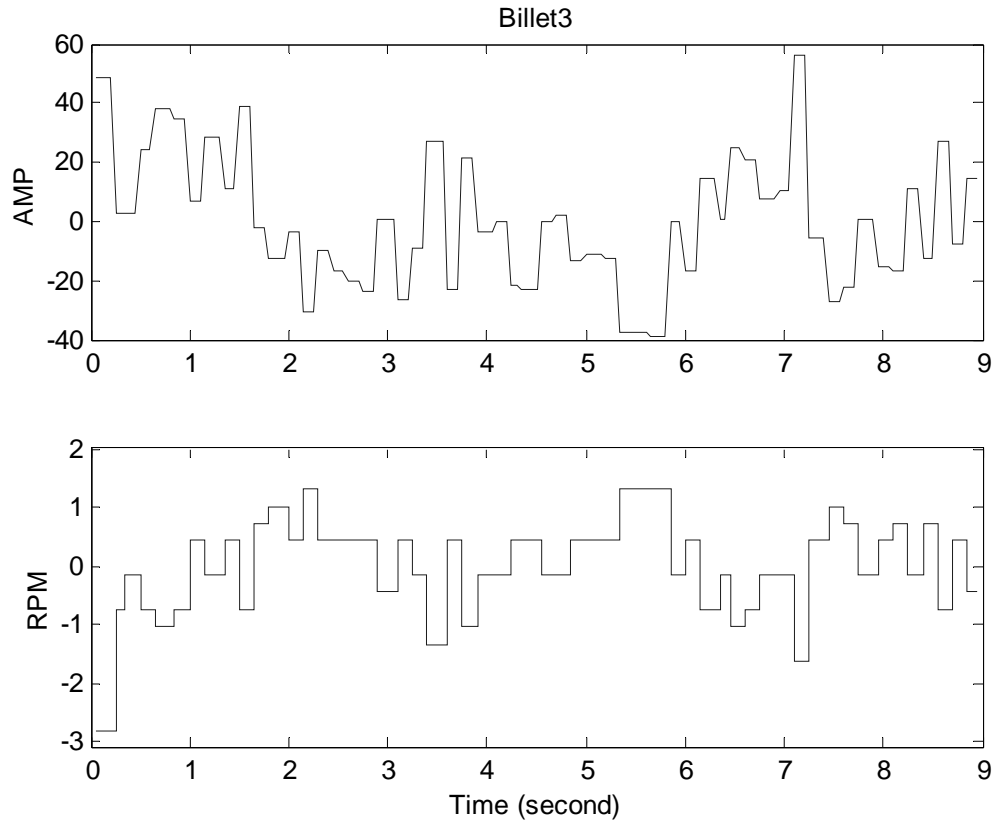
Figure 2.1.2   Data for Verification

After comparing with (modified) least-squares methods and instrument variable method, the generalized least-squares method, based on model structure Equation 2-1-2 or 2-1-3 ([24] and [25]), gives a better description of the rolling mill stand.

$$y(t) = \frac{B(q)}{A(q)} u(t - nk) + \frac{G(q)}{H(q)} e(t) \qquad (2\text{-}1\text{-}2)$$

or

$$A(q) y(t) = B(q) u(t - nk) + \frac{1}{F(q)} e(t) \qquad (2\text{-}1\text{-}3)$$

where:

$$\frac{1}{F(q)} = \frac{A(q) G(q)}{H(q)}.$$

Via trial-and-error, a third-order dynamic discrete transfer function, as shown in Equation 2-1-4, predicts system behavior reasonably. The prediction performance , as shown in Figure 2.1.3, justifies the use of this model in the following simulations.

$$y(z) = \frac{-28.6666z^{-1} + 23.9194z^{-2}}{1 - 0.87224z^{-1} + 0.0027044z^{-2} - 0.039773z^{-3}} u(z) \qquad (2\text{-}1\text{-}4)$$

The mathematical model of the rolling mill stand working in the normal mode is constructed according to the discrete transfer function $\dfrac{b_0 z^2 + b_1 z^1}{z^3 + a_1 z^2 + a_2 z + a_3}$ where

a1 = -0.8722, a2 = 0.0027, a3 = -0.0398, b0 = -28.6667, b1 = 23.9194.
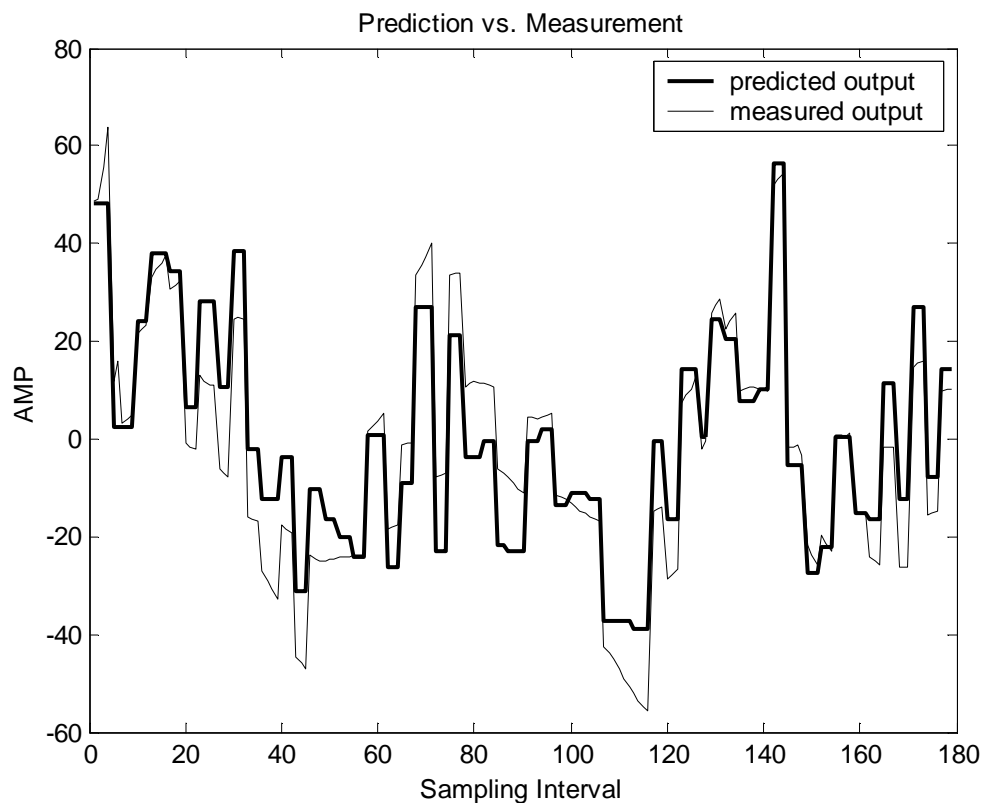
Figure 2.1.3   Identification and Verification

2.2 Design Factors of FLC

The design of a fuzzy logic controller (Figure 2.2.1) needs the selections of such control elements and parameters as scaling factors for input/output signals, a set of rule base, fuzzification and defuzzification methods and operations for the fuzzy reasoning, which include a implication operation, a compositional operation and aggregation operations of antecedents and consequents [26]. The performance of the FLC heavily relies on the configuration of these factors.
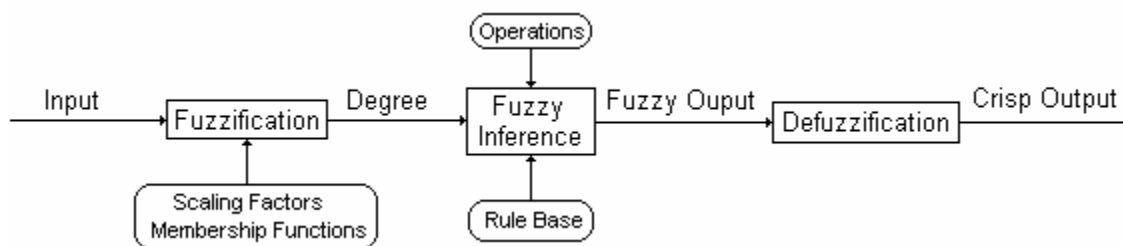
Figure 2.2.1   Fuzzy Logic Controller

2.2.1 Fuzzification Method

In a fuzzy control system, the measurement of the input signal is interpreted as a fuzzy singleton.

According to the type of fuzzy reasoning, the linguistic variables can be fuzzified in two ways. In the Mamdani-type fuzzy inference system, both the input and output linguistic variables will take fuzzy variables as values. In the Sugeno-type, generally zero order, fuzzy inference system, the input linguistic variables are fuzzified as fuzzy variables, while the output linguistic variables take fuzzy singletons as values.

Fuzzy variables are defined by membership functions (MF) and characterized by shapes, positions and width or whole overlap. The number of the fuzzy variables

that a linguistic variable can take is known as fuzzy partition and determines the control granularity obtainable from a FLC. A trial-and-error based on operators' experience and engineering knowledge is extensively employed in the choice of MF's. Despite *ad hoc* approaches for MF's selection, some guidelines exist. In general, for the sake of computational efficiency, efficient use of the computer memory and performance analysis, the MF's are required to have uniform shapes, parameters and function definitions [28]. Likewise, the number of the MF's determined by fuzzy partitions are a trade-off between adequate approximation and available memory space. J. V. D. Oliveira [28] suggested that the number of fuzzy variables that a linguistic variable can take should be between 5 and 9. The fuzzification should cover the entire universe of discourse, and there exists a fuzzy number to represent the fuzzy variable "around zero". As for a certain shape of a membership function, narrower membership functions, despite superiority in faster response and lower steady-state error, may incur larger oscillation, and thus the system will be unstable especially in noisy environment [27 and 28].

In [22], the piece-wise continuous membership functions, trapezoidal and triangular shapes, are successfully applied in the looper control for rolling mills. Since these two MF shapes match human's intuition and possess computation simplicity, they will be kept on in this work. Within the universe of discourse U = [-1, 1], they are defined in Table 2.2.1.

Shape	Formula	Graph
Trapezoid	$\mu(a,b,c,d) = \begin{cases} 0 & x < a \\ \dfrac{x-a}{b-a} & a \le x < b \\ 1 & b \le x < c \\ \dfrac{x-d}{c-d} & c \le x < d \\ 0 & d \le x \end{cases}$ $a \le b \le c \le d$	Trapezoid Membership Function
Triangle	$\mu(a,b,c) = \begin{cases} 0 & x < a \\ \dfrac{x-a}{b-a} & a \le x < b \\ \dfrac{x-c}{b-c} & b \le x < c \\ 0 & c \le x \end{cases}$ $a \le b \le c$	Triangle Membership Function

Table 2.2.1   Membership Function Candidates

2.2.2 Scaling Factor

The scaling factors (SF) determined by the plant and actuator are used to normalized different range of input/output signals into the universe of discourse [-1, 1] and thus can generalize and facilitate the design and configuration of the fuzzy controller. However, unsuitable SF's for the input/output signals can worsen transient and steady-state responses, while the scopes of input/output signals, as the scaling factors, are safe but conservative..

2.2.3 Fuzzy Rule Base

The fuzzy rule base assemblies plant information and applies the human control expertise to the given problem. For a two-input-two-output fuzzy controller, the form of fuzzy rules can be:

$$\vdots$$

Rule r: If   (input X_1 is X_1^r)   and   (input X_2 is X_2^r)
                 then   (output Y_1 is Y_1^r)   and   (output Y_2 is Y_2^r)

$$\vdots$$

2.2.4 Operations

From input i to output j in rule r, two common kinds of implication operations, min (Equation 2-2-1) and product (Equation 2-2-2) will be studied.

$$\mu_{X_i^r \to Y_j^r}(x_i, y_j) = \mu_{X_i^r}(x_i) \wedge \mu_{Y_j^r}(y_j) \tag{2-2-1}$$

$$\mu_{X_i^r \to Y_j^r}(x_i, y_j) = \mu_{X_i^r}(x_i) \cdot \mu_{Y_j^r}(y_j) \tag{2-2-2}$$

Because of the boundary conditions, the max-t-norms are equalivalent provided that measurements of the input signals are fuzzified as fuzzy singletons. For computation simplicity, max-min compositional operation is used, *i.e.*, for input i and output j in rule r:

$$\mu_{Y_{i,j}^r}(y_j) = \bigvee_U (\mu_{X_i}(x_i) \wedge \mu_{X_i^r \to Y_j^r}(x_i, y_j)) \tag{2-2-3}$$

Stemming from computation simplicity, the aggregation operations of antecedents min and product are employed and will be compared. For rule r in a two-input fuzzy inference system, these two operations can be defined by Equation 2-2-4 and Equation 2-2-5 respectively.

$$\mu_{Y_j^r}(y_j) = \mu_{Y_{1,j}^r}(y_j) \wedge \mu_{Y_{2,j}^r}(y_j) \qquad (2\text{-}2\text{-}4)$$

$$\mu_{Y_j^r}(y_j) = \mu_{Y_{1,j}^r}(y_j) \cdot \mu_{Y_{2,j}^r}(y_j) \qquad (2\text{-}2\text{-}5)$$

When the implication operation is selected as aforementioned, it is customary to use union

$$\mu_{Y_j^{'}}(y_j) = \bigvee_r \mu_{Y_j^r}(y_j) \qquad (2\text{-}2\text{-}6)$$

as the aggregation operation of consequents to draw the fuzzy conclusion.

2.2.5 Defuzzification Method

To derive the crisp control action from the fuzzy conclusion, two defuzzification methods can be chosen:

(1)   Centroid of Area (COA) in the Mamdani-type inference system:

$$y_j = \frac{\displaystyle\sum_q^U y_{j,q} \cdot \mu_{Y_j^{'}}(y_{j,q})}{\displaystyle\sum_q^U \mu_{Y_j^{'}}(y_{j,q})} \qquad (2\text{-}2\text{-}7)$$

where:

$y_{j,q}$ the value of the jth output fuzzy variable at the quantization level q.

or

Weighted Average (WA) in the Sugeno-type inference system:

$$y_j = \frac{\sum\limits_{r}^{R} y_{j,r} \cdot \mu_{Y_j^{'}}(y_{j,r})}{\sum\limits_{r}^{R} \mu_{Y_j^{'}}(y_{j,r})} \tag{2-2-8}$$

where:

$y_{j,r}$ the fuzzy singleton value of the jth consequent in the rth fuzzy rule.

R   rule number in the fuzzy rule base.

(2)   Mean of Maximum (MOM):

$$y_j = \sum\limits_{m=1}^{n} \frac{y_{j,m}}{n} \tag{2-2-9}$$

where:

$y_{j,m}$ the value of the jth output fuzzy variable at which the degree of membership function $\mu_{Y_j^{'}}(y_{j,m})$ reaches the maximum value.

n   the number of such values.

In general, COA demonstrates better steady-state performance [21], while MOM yields superior transient performance. According to R. Jager *et al* [30], steady state error associated with MOM due to the discontinuity property can be overcome by increasing the number of membership functions for outputs, which, however, will result in oscillatory control signals. This problem can be solved, in the price of more memory and calculation time, by increasing the number of membership functions for inputs.

2.2.6 Inference Engine

Based on engineering experience and stand operation knowledge presented in [22], the forthcoming simulation will be conducted based on a set of intuitively chosen fuzzy variables. Five fuzzy variables are set for input1 i_e, *i.e.*, NB, NS,

Z, PS and PB, three for input2 Δi_e, *i.e.*, N, Z and P, and seven for output Δv_f, *i.e.*, NB, NM, NS, Z, PS, PM and PB. Based on the fuzzy variables selected, the observation of behavior of human operators and trial-and-error, the fuzzy rule base that can ensure the stability and steady state precision are summarized as follows:

1. If (input1 is NB) and (input2 is N) then (output is PB)
2. If (input1 is NB) and (input2 is Z) then (output is PB)
3. If (input1 is NB) and (input2 is P) then (output is PB)
4. If (input1 is NS) and (input2 is N) then (output is PM)
5. If (input1 is NS) and (input2 is Z) then (output is PM)
6. If (input1 is NS) and (input2 is P) then (output is PS)
7. If (input1 is Z) and (input2 is N) then (output is PS)
8. If (input1 is Z) and (input2 is Z) then (output is Z)
9. If (input1 is Z) and (input2 is P) then (output is NS)
10. If (input1 is PS) and (input2 is N) then (output is NS)
11. If (input1 is PS) and (input2 is Z) then (output is NM)
12. If (input1 is PS) and (input2 is P) then (output is NM)
13. If (input1 is PB) and (input2 is N) then (output is NB)
14. If (input1 is PB) and (input2 is Z) then (output is NB)
15. If (input1 is PB) and (input2 is P) then (output is NB)

2.3 Simulation of Fuzzy Logic Controller

Given preceding configurations, the following simulations focus on analyzing the influences of elements, such as the fuzzification/defuzzification methods, implication operations, aggregation operations of antecedents and scaling factors, on the performance of the FLC.

The fuzzy reasoning can be depicted as follows:

```
//Fuzzy inference Engine
while (simulation running)
```

        Take the measurement of input1 and input2 weighted by the SF's for inputs

        Compute the fact by the fuzzification of the measurement

        for r = 1 to 15

            for i = 1 to 2

                Compute the antecedent by the compositional operation

            end i

            Compute the firing strength by the aggregation operation of antecedents

            Weight the firing strength by the rule weight

            Compute the consequent by the implication operation

        end r

        Draw the fuzzy conclusion by the aggregation operation of consequents

        Compute the crisp action by the defuzzification method

        Output the actuation signal weighted by the SF for output

end while

In this point, the simplest behavior of human operators is adopted: unit step signal is used as tension reference, *i.e.*, target current i_t.

2.3.1 Influence of Fuzzification Method

The parameter whole overlap is an appropriate criteria for the evaluation of membership functions since both the width and the relevant position of the MF's are taken into account. In this scenario, the piece-wise continuous MF's and the Mamdani-type fuzzy inference system will be employed; min is used for both the aggregation operation of antecedents and the implication operation; and the defuzzification method is COA. The investigation will concentrate on the influence of the whole overlap (WO), which is defined as [27]

$$wo = \frac{\int_U \min(\mu_{F_1}(f), \mu_{F_2}(f))}{\int_U \max(\mu_{F_1}(f), \mu_{F_2}(f))} \times 100\% \qquad (2\text{-}3\text{-}1)$$

and expressed as the ratio between the overlap and the overall area of the fuzzy variables Z and PS for input1 i_e, *i.e.*,

$$wo = \frac{\displaystyle\int_{[-1,1]} \min(\mu_Z(x_1), \mu_{PS}(x_1))}{\displaystyle\int_{[-1,1]} \max(\mu_Z(x_1), \mu_{PS}(x_1))} \times 100\% \tag{2-3-2}$$

The influence of the whole overlap is discussed in three cases: WO = 2.58%, 14.49% and 23.36%. The MF's parameters for the three cases are listed in Table 2.3.1 ~ Table 2.3.3 respectively.

The results of simulations with these three WO values are contrasted in Figure 2.3.1. It can be seen the whole overlap *14.29%* give fast response, small overshoot and accurate steady state. Small whole overlap can accelerate the response at the expense of higher peak value and bigger steady-state error.

Plot	Parameters
Membership Functions input1: i_e NB NS Z PS PB input2: Δi_e N Z P output: Δv_r NB NM NS Z PS PM PB Control Surface output: Δv_r input1: i_e input2: Δi_e	**Case 2:** WO = 2.58% **Input1 i_e :** Scaling factor: 1 NB: Trapezoidal μ(-1, -1, -0.6, −0.4) NS: Triangular μ(-0.5, -0.3, −0.1) Z: Triangular μ(-0.2, 0, 0.2) PS: Triangular μ(0.1, 0.3, 0.5) PB: Trapezoidal μ(0.4, 0.6, 1, 1) **Input2 Δi_e :** Scaling factor: 3 N: Trapezoidal μ(-1, -1, -0.5, -0.2) Z: Triangular μ(-0.3, 0, 0.3) P: Trapezoidal μ(0.2, 0.5, 1, 1) **Output Δv_f :** Scaling factor: 0.1 NB: Trapezoidal μ(-1, -1, -0.9, -0.7) NM: Triangular μ(-0.8, -0.6, -0.4) NS: Triangular μ(-0.5, -0.3, -0.1) Z: Triangular μ(-0.2, 0, 0.2) PS: Triangular μ(0.1, 0.3, 0.5) PM: Triangular μ(0.4, 0.6, 0.8) PB: Trapezoidal μ(0.7, 0.9, 1, 1)

Table 2.3.1   Parameters of MF's (WO = 2.58%)

Plot	Parameters
Membership Functions *(input1: i_e, input2: Δi_e, output: Δv_r)* Control Surface	**Case 3:** WO = 14.29% **Input1 i_e :** Scaling factor: 1 NB: Trapezoidal μ(-1, -1, -0.6, –0.3) NS: Triangular μ(-0.6, -0.3, 0) Z: Triangular μ(-0.3, 0, 0.3) PS: Triangular μ(0, 0.3, 0.6) PB: Trapezoidal μ(0.3, 0.6, 1, 1) **Input2 Δi_e :** Scaling factor: 3 N: Trapezoidal μ(-1, -1, -0.5, 0) Z: Triangular μ(-0.5, 0, 0.5) P: Trapezoidal μ(0, 0.5, 1, 1) **Output Δv_f :** Scaling factor: 0.1 NB: Trapezoidal μ(-1, -1, -0.9, -0.6) NM: Triangular μ(-0.9, -0.6, -0.3) NS: Triangular μ(-0.6, -0.3, 0) Z: Triangular μ(-0.3, 0, 0.3) PS: Triangular μ(0, 0.3, 0.6) PM: Triangular μ(0.3, 0.6, 0.9) PB: Trapezoidal μ(0.6, 0.9, 1, 1)

Table 2.3.2   Parameters of MF's (WO = 14.29%)
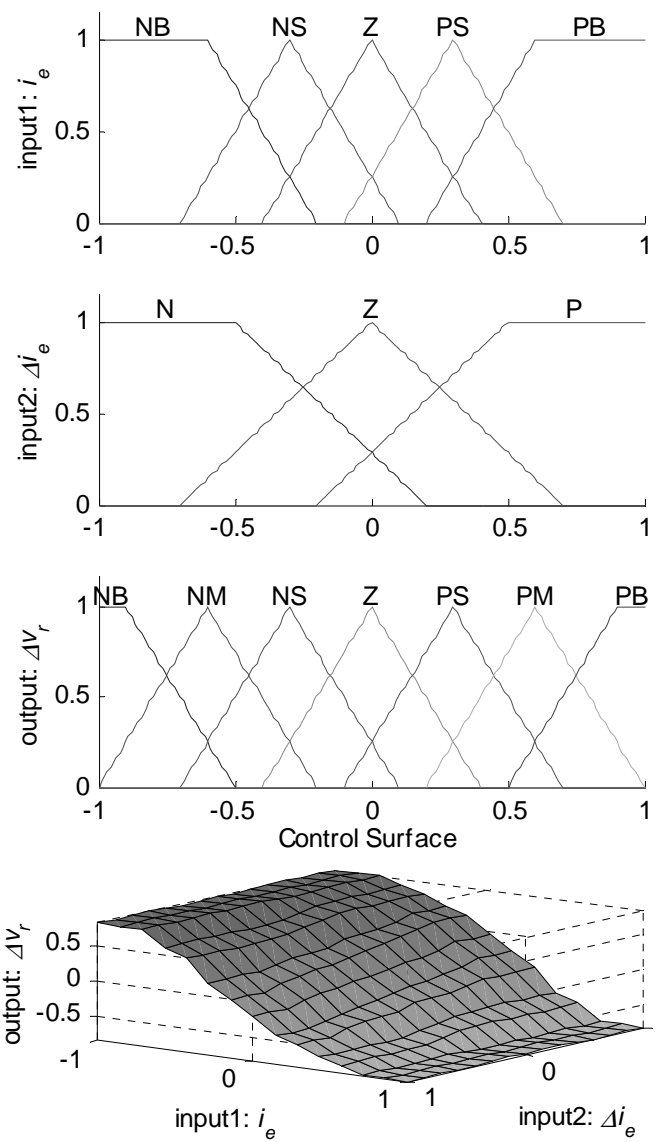
Plot	Parameters
Membership Functions 	**Case 3:** WO = 23.36% **Input1** i_e **:** Scaling factor: 1 NB: Trapezoidal μ(-1, -1, -0.6, –0.2) NS: Triangular μ(-0.7, -0.3, 0.1) Z: Triangular μ(-0.4, 0, 0.4) PS: Triangular μ(-0.1, 0.3, 0.7) PB: Trapezoidal μ(0.2, 0.6, 1, 1) **Input2** Δi_e **:** Scaling factor: 3 N: Trapezoidal μ(-1, -1, -0.5, 0.2) Z: Triangular μ(-0.7, 0, 0.7) P: Trapezoidal μ(-0.2, 0.5, 1, 1) **Output** Δv_f **:** Scaling factor: 0.1 NB: Trapezoidal μ(-1, -1, -0.9, -0.5) NM: Triangular μ(-1, -0.6, -0.2) NS: Triangular μ(-0.7, -0.3, 0.1) Z: Triangular μ(-0.4, 0, 0.4) PS: Triangular μ(-0.1, 0.3, 0.7) PM: Triangular μ(0.2, 0.6, 1) PB: Trapezoidal μ(0.5, 0.9, 1, 1)

Table 2.3.3   Parameters of MF's (WO = 23.36%)
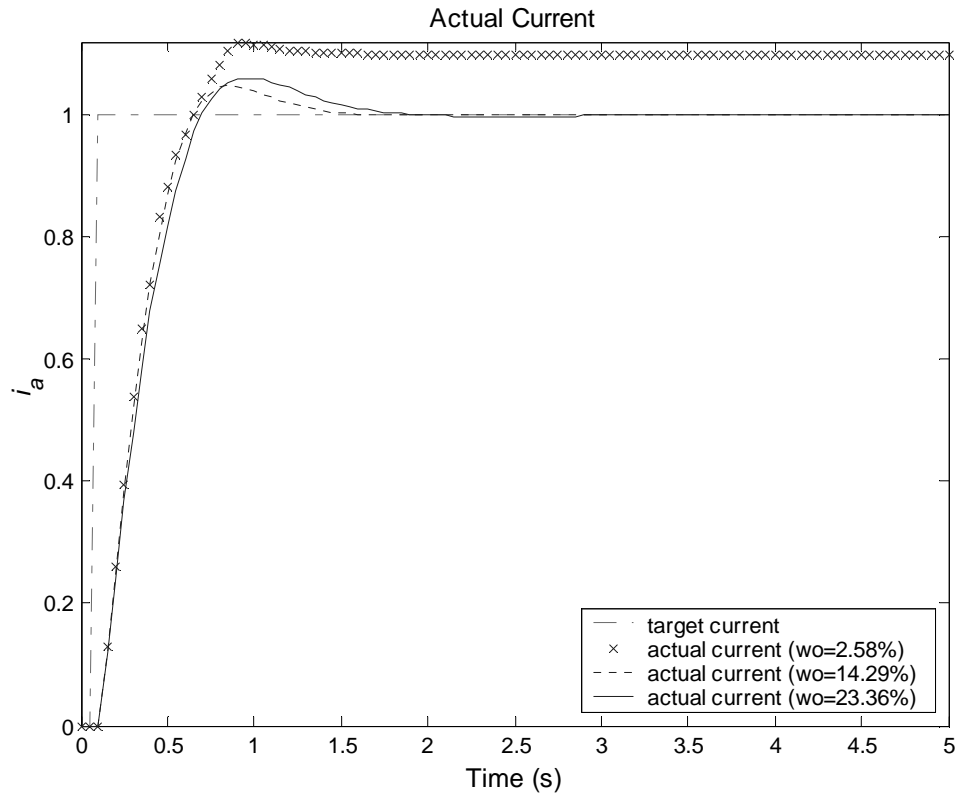
Figure 2.3.1   Influence of Whole Overlap

2.3.2 Influence of Scaling Factors

In this situation, the piece-wise continuous MF's with WO = 14.29% (as configured in Table 2.3.3) and the Mamdani-type fuzzy inference system will be employed; min is used for both the aggregation operation of antecedents and the implication operation; and the defuzzification method is COA. The investigation will concentrate on the influences of the scaling factors for the input/output signals. The four groups of the scaling factor options are listed in Table 2.3.4.

Case	input1 i_e	input2 Δi_e	Output Δv_f
1	0.5	1	0.1
2	1	1	0.1
3	1	3	0.1
4	1	3	0.25

Table 2.3.4 Configurations of SF's

The simulation results of these four groups of SF's are shown in Figure 2.2.3. The selection of the scaling factors is a trade-off between response speed and precision. Inappropriate SF's for the output can cause oscillation (case 4). The best response result is obtained in case 3.
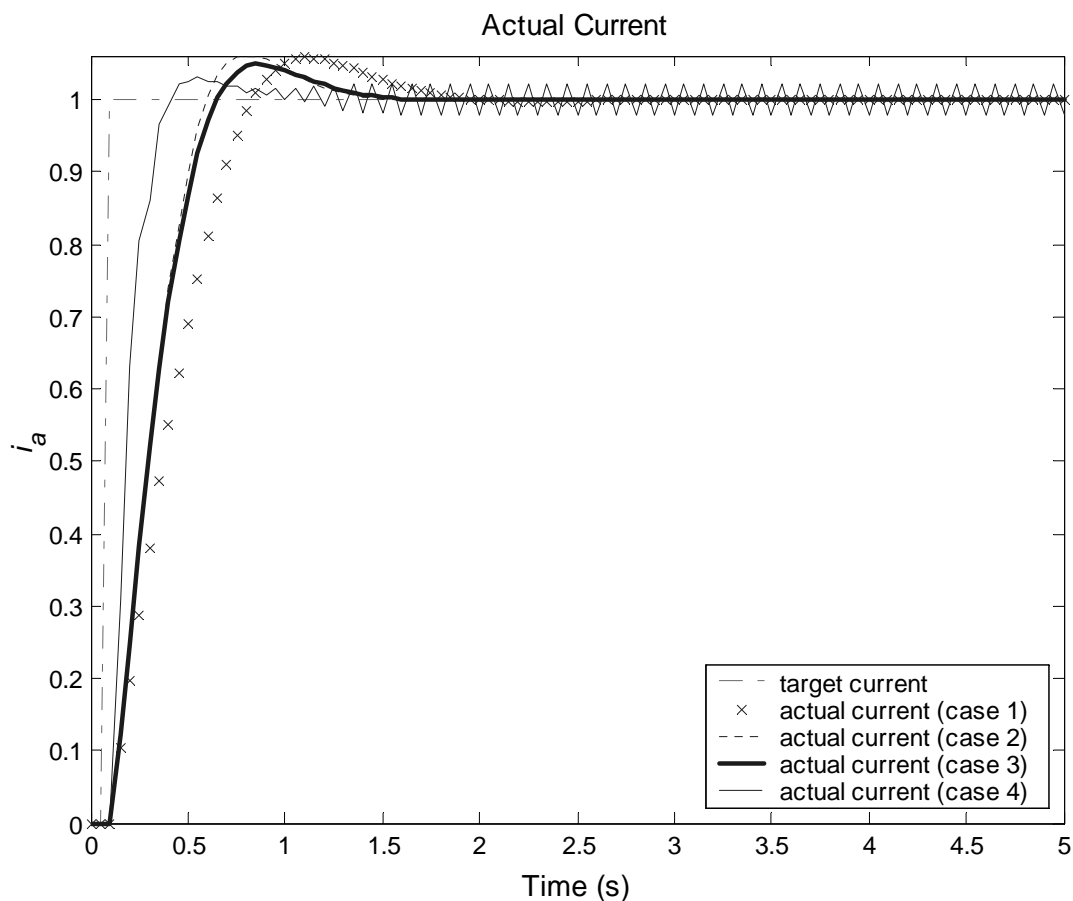
Figure 2.3.2   Influence of Scaling Factors

2.3.3 Influence of Fuzzy Reasoning Type

In this scenario, the piece-wise continuous MF's are configured as in Table 2.3.3; and min is used for both the aggregation operation of antecedents and the implication operation. The investigation will concentrate on the influence of the fuzzy reasoning type, *i.e.*, the Mamdani-type versus the Sugeno-type. The corresponding defuzzification methods are COA and WA respectively. The only modification in the Sugeno-type inference system is the fuzzy singletons for the output as listed in Table 2.3.5. The distribution of the fuzzy singletons over the universe of discourse is set as close to the center of the counterparts in the Mamdani-type as possible.

From the simulation results (Figure 2.3.3), it is safe to say that the Sugeno-type fuzzy inference system has compatible performance to the Mamdani-type. This assertion provides basis for the general application of the Sugeno-type fuzzy inference system because of its simple structure and future applicability of the backward propagation algorithm to it.

Plot	Parameters
Membership Functions	**Input1** i_e **:**
	Scaling factor: 1
	NB: Trapezoidal μ(-1, -1, -0.6, −0.3)
	NS: Triangular μ(-0.6, -0.3, 0)
	Z: Triangular μ(-0.3, 0, 0.3)
	PS: Triangular μ(0, 0.3, 0.6)
	PB: Trapezoidal μ(0.3, 0.6, 1, 1)
	Input2 Δi_e **:**
	Scaling factor: 3
	N: Trapezoidal μ(-1, -1, -0.5, 0)
	Z: Triangular μ(-0.5, 0, 0.5)
	P: Trapezoidal μ(0, 0.5, 1, 1)
	Output Δv_f **:**
	Scaling factor: 0.1
	NB: -0.75
	NM: -0.5
	NS: -0.25
	Z: 0
	PS: 0.25
	PM: 0.5
	PB: 0.75

Table 2.3.5   Parameters of MF's (Sugeno-type)

Figure 2.3.3   Influence of Fuzzy Reasoning Type

2.3.4 Influence of Operations

In this situation, the piece-wise continuous MF's with WO = 14.29% (as shown in Table 2.3.3) and the Mamdani-type fuzzy inference system will be employed; and the defuzzification method is COA. The investigation will concentrate on the influences of both the aggregation operation of antecedents and the implication operation as discriminated by three cases (Table 2.3.6).
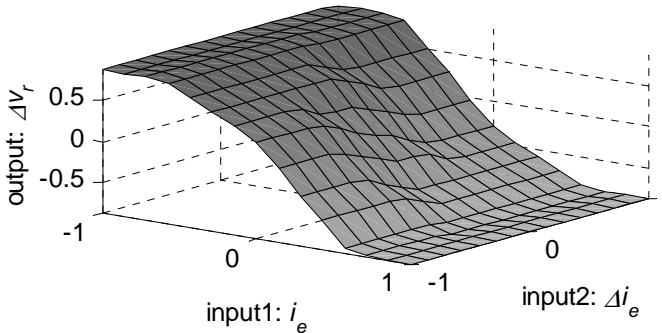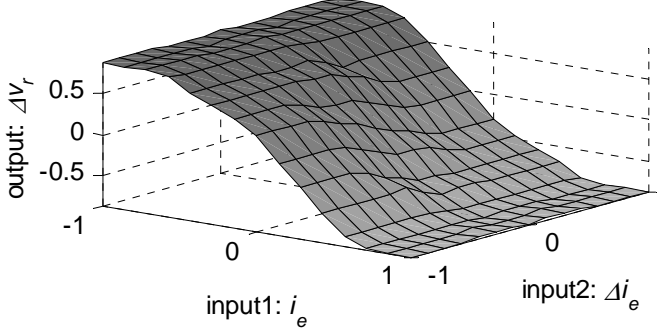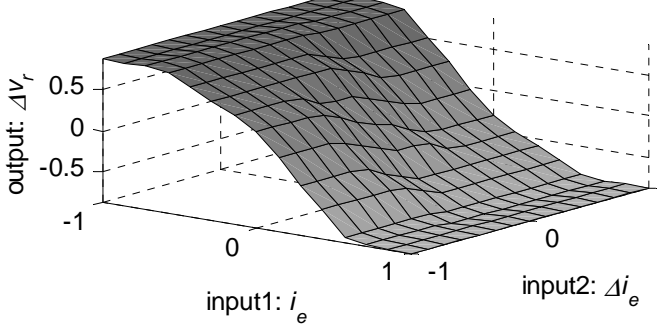
Control Surfaces	Operations
Case 1: *min + min*	**Case 1:** Aggregation operation for antecedent: min Implication operation: min
Case 2: *product + min*	**Case 2:** Aggregation operation for antecedent: product Implication operation: min
Case 3: *product + product*	**Case 3:** Aggregation operation for antecedent: product Implication operation: product

Table 2.3.6   Control Surfaces (Operation Comparison)

The simulation results are plotted in Figure 2.3.4, with the influences of the operations manifested in the subplot of the transient response. Using product as the implication operation can accelerate the initial response (case 3). As for the peak value, the min for both the aggregation for antecedent and the implication operation is much desirable (case 1).
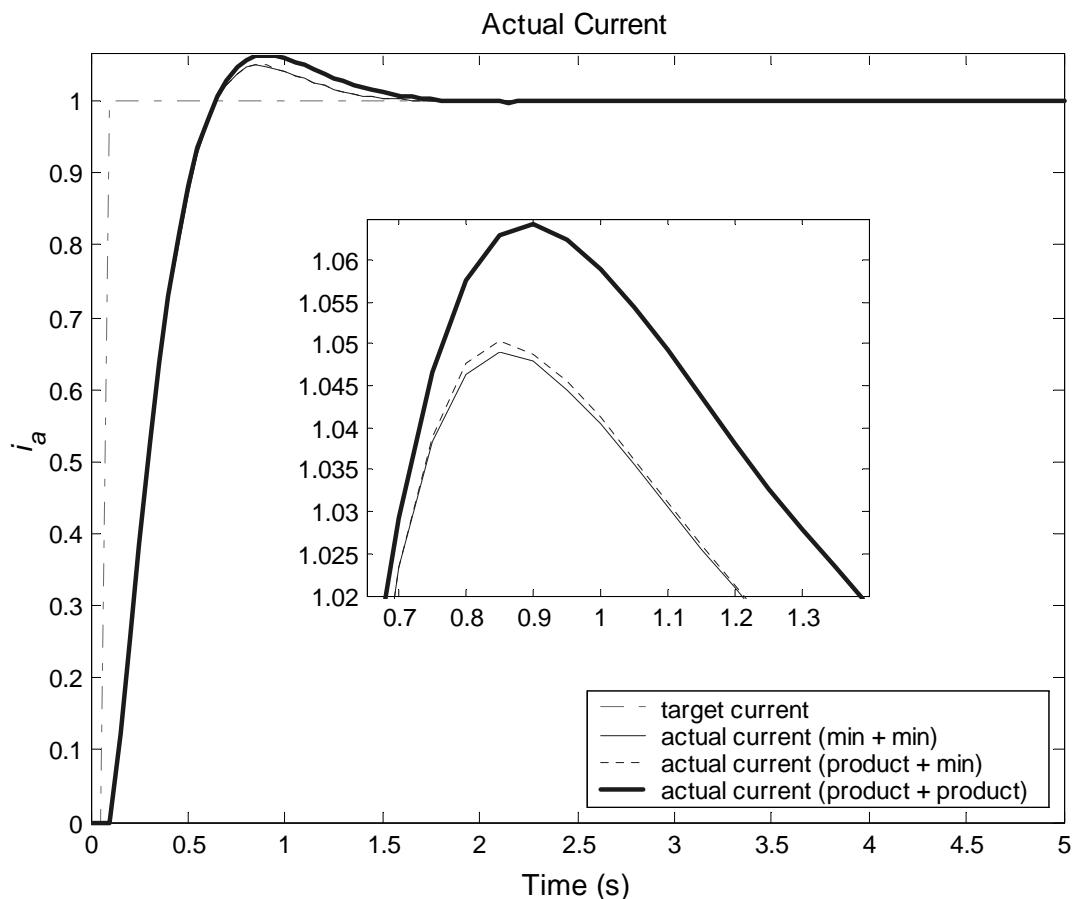
Figure 2.3.4   Influence of Operations

2.3.5   Influence of Defuzzification Method

In this scenario, the piece-wise continuous MF's with WO = 14.29% (as configuration in Table 2.3.3) and the Mamdani-type fuzzy inference system will be employed; and min is used for both the aggregation operation of antecedents and the implication operation. The investigation will concentrate on the comparison of performances for COA and MOM as the defuzzification methods as listed in Table 2.3.7.
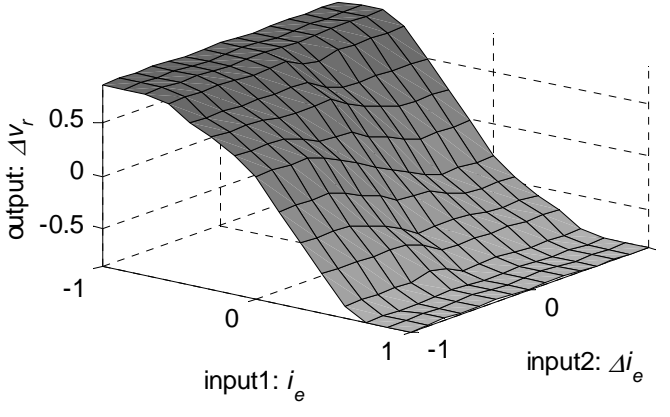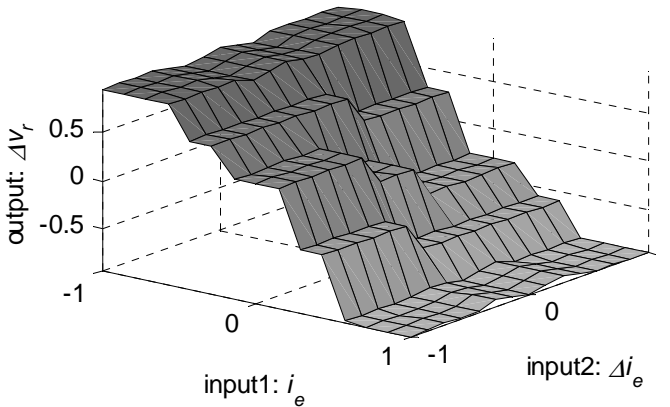
Control Surfaces	Defuzzification Methods
Case 1: *COA* 	**Case 1:** Defuzzification Method: COA
Case 2: *MOM* 	**Case 2:** Defuzzification Method: MOM

Table 2.3.7   Control Surfaces (Defuzzification Method Comparison)

The simulation results are shown in Figure 2.3.5. The plots verify that the FLC with COA has superior steady-state response and ensures stability, while MOM gives rapid initial response.
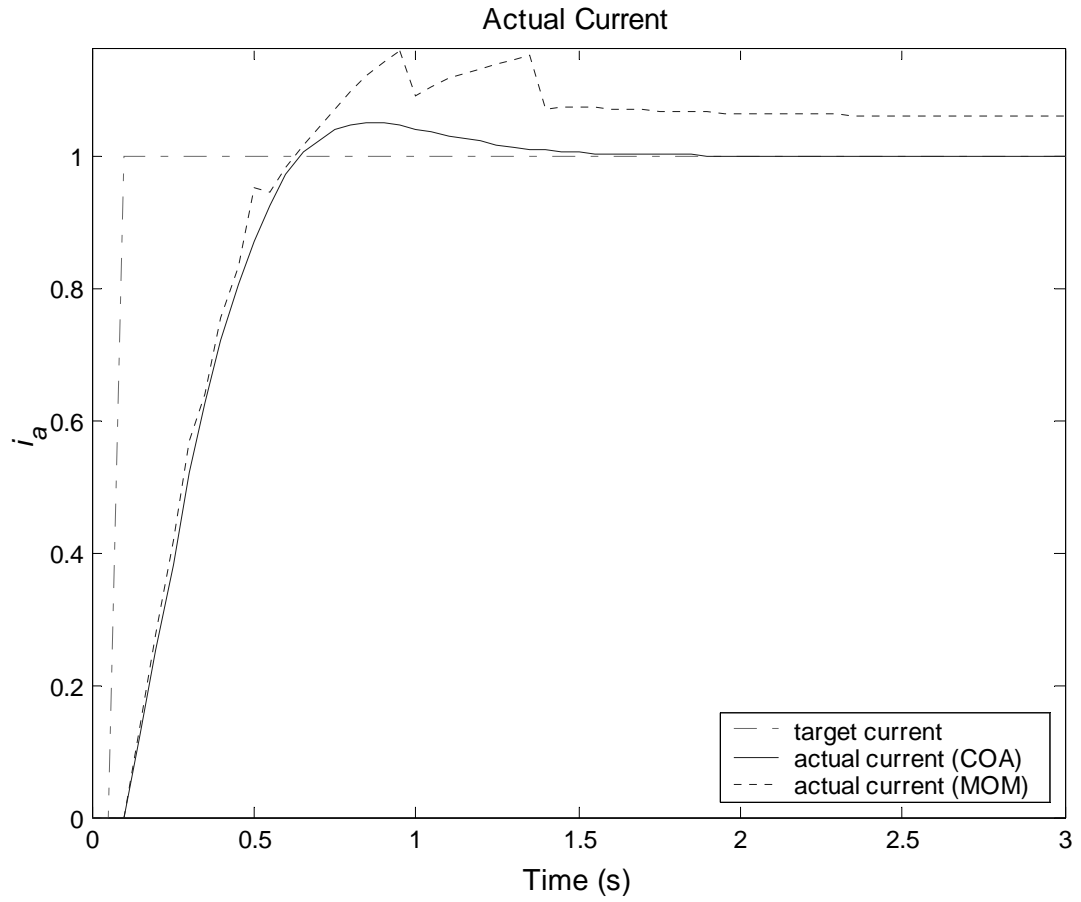
Figure 2.3.5 Influence of Defuzzification Method

2.4 Evaluation of Fuzzy Logic Controller

Following the aforementioned routine, the tension control system based on fuzzy logic can be designed without relying on much knowledge of the plant. As an example, the FLC is configured as in Table 2.3.5 and product is selected as both the aggregation operation of antecedents and the implication operation. Specifications, such as stability and steady state precision, can be achieved easily (Figure 2.4.1), however, the performance indices, such as response speed, overshoot, are not satisfactory. Compared with a conventional controller, it is difficult to optimize the FLC because of the large numbers of control elements involved. This can be seen by replacing the FLC with a PID controller in the tension control system. As demonstrated in Figure 2.4.1, the PID controller can be optimized by choosing P = 0.0005, I = 0.6 and D = 0.00002, despite the incorporation of the characteristic of the plant,. It is expected that some techniques may be used to optimize the FLC. In the rest of this work, some of these techniques will be exploited to configure the elements, such as SF's, MF's, rule base and operations.
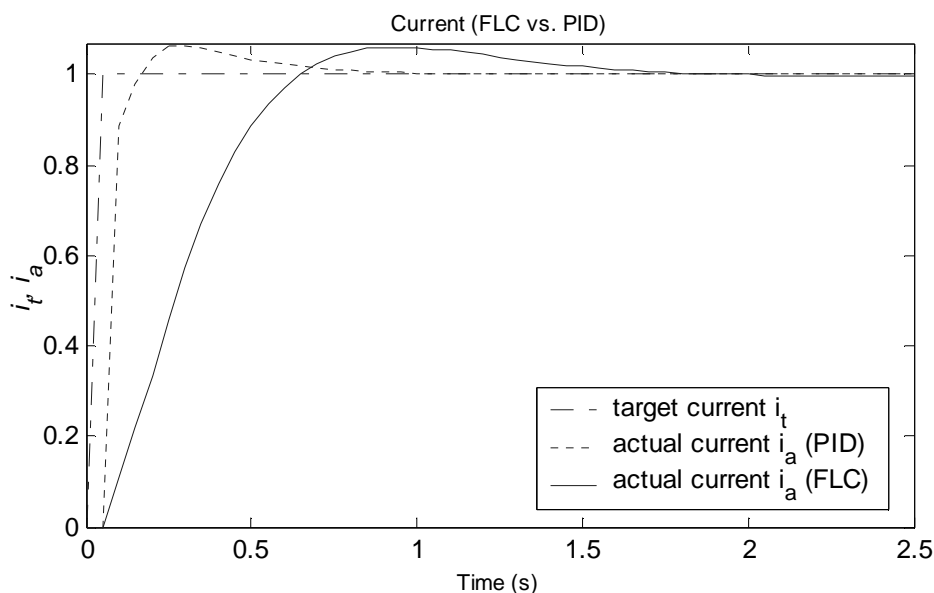
Figure 2.4.1   Unit Step Tension Response (FLC vs. PID)

Chapter 3

Dynamic Fuzzy Tension Controller

From the previous design and analysis of the fuzzy tension controller, it can be seen that different control elements perform well at different stages of response. One natural way to improve the performance of the FLC is dynamically switching the control elements during control activity, a scheme known as dynamic fuzzy controller.

3.1 Dynamic Control Elements

The dynamic switching of control elements associates with desired response criteria directly. Some conventional specifications defined for a unit step response of the actual current will be used here. Along with the specifications, the performances of the response are evaluated according to:

- Stability.
- Rapid response speed measured by the rising time, which is defined as the time required for the response of actual current i_a to rise from 0 to 90% final value, *i.e.*, 1; or equivalently, current error i_e decreases from 1 to 0.1.
- Small overshoot.
- Short transient period measured by the settling time, which is defined as the time required for current error i_e and current error variance Δi_e to settle within the area of ±3% and ±2% respectively, with 0 final values.

- Accurate steady state.

According to the arguments stated in the design issues, after summarizing the fuzzy partitions and establishing the rule base from engineering knowledge, these specifications are affected by the fuzzification methods, scaling factors, aggregation operations of antecedents, implication operations and defuzzification methods.

In this DFC, the input/output fuzzy variables and rule base are locked as listed in Chapter 2, Section 2.2.6. The fuzzy inference is of the Mamdani-type. Since only the membership functions with WO = 14.29% give satisfactory result, the MF's are configured as in Table 2.3.2. The scaling factors for input1 i_e is 1, *i.e.*, the maximum of the system output signal, because this value is safe and easier to determine than the other two. The dynamic elements are the scaling factors for input2 Δi_e and output Δv_f, the aggregation operations of antecedents, the implication operations and the defuzzification methods.

3.2 Dynamic Control Scheme

In last chapter, it was observed that:

(1)   Decreasing the SF for input2 Δi_e, increasing the SF for output Δv_f, product as both the aggregation operation for antecedent and implication and MOM as the defuzzification method accord with rapid initial response.

(2)   Increasing the SF for both input2 Δi_e and output Δv_f, min as both the aggregation operation for antecedent and implication and COA as the defuzzification method give small peak value and short transient period.

(3)   Decreasing the SF for output Δv_f and COA as defuzzification method guarantee steady-state precision.

To see the instant improvement, the parameters and operations tested in Chapter 2 will be used again. The DFC designed here is a three-stage controller in favor of the response indices and the trigger of different elements is divided into three phases during the unit step response of tension control system.

The response part before the rising time, *i.e.*, $|i_e| \geq 10\%$, is defined as phase 1 during which a rapid initial response is expected. Therefore, in phase 1, the scaling factors for input2 Δi_e and output Δv_f are 1, 0.25 respectively. Product is used as both the aggregation operation of antecedents and implication operation; and MOM is selected as the defuzzification method.

After phase 1, a rapid transient period with small peak value is desired which is identified as phase 2, *i.e.*, $3\% < |i_e| < 10\%$ and $|\Delta i_e| > 2\%$. Based on the previous discussion, the scaling factor for input2 Δi_e is changed to 3; min is used for both the aggregation operation of antecedents and implication operation. The defuzzification method COA supplants MOM.

The control system enters phase 3 after the settling time, *i.e.*, $|i_e| \leq 3\%$ and $|\Delta i_e| \leq 2\%$. To accelerate the process to turn into steady-state response with small final error, the scaling factors for output Δv_f is changed to 0.1, while other control factors remain the same as in phase 2.

3.3 Simulation of Dynamic Control

For comparison with the DFC, three simulations of non-dynamic FLC's with same configuration as phase1, 2 and 3 of the DFC respectively, are conducted (Table 3.3.1).

Case	SF for Input2 Δi_e	Aggregation Operation for Antecedents	Implication Operation	Defuzzification	SF for Output Δv_f
1	1	Product	Product	MOM	0.25
2	3	Min	Min	COA	0.25
3	3	Min	Min	COA	0.1

Table 3.3.1 Configurations of Non-dynamic FLC's

The four unit step responses of actual current i_a are plotted in Figure 3.3.1; the compartmentalization of the three phase areas is also labeled. It can be seen that the DFC combines the merits of the three cases: the DFC has the same initial tacking speed as case 1, undergoes short transient period, suppresses the overshoot dramatically and enters steady response earilier than all the other three cases.
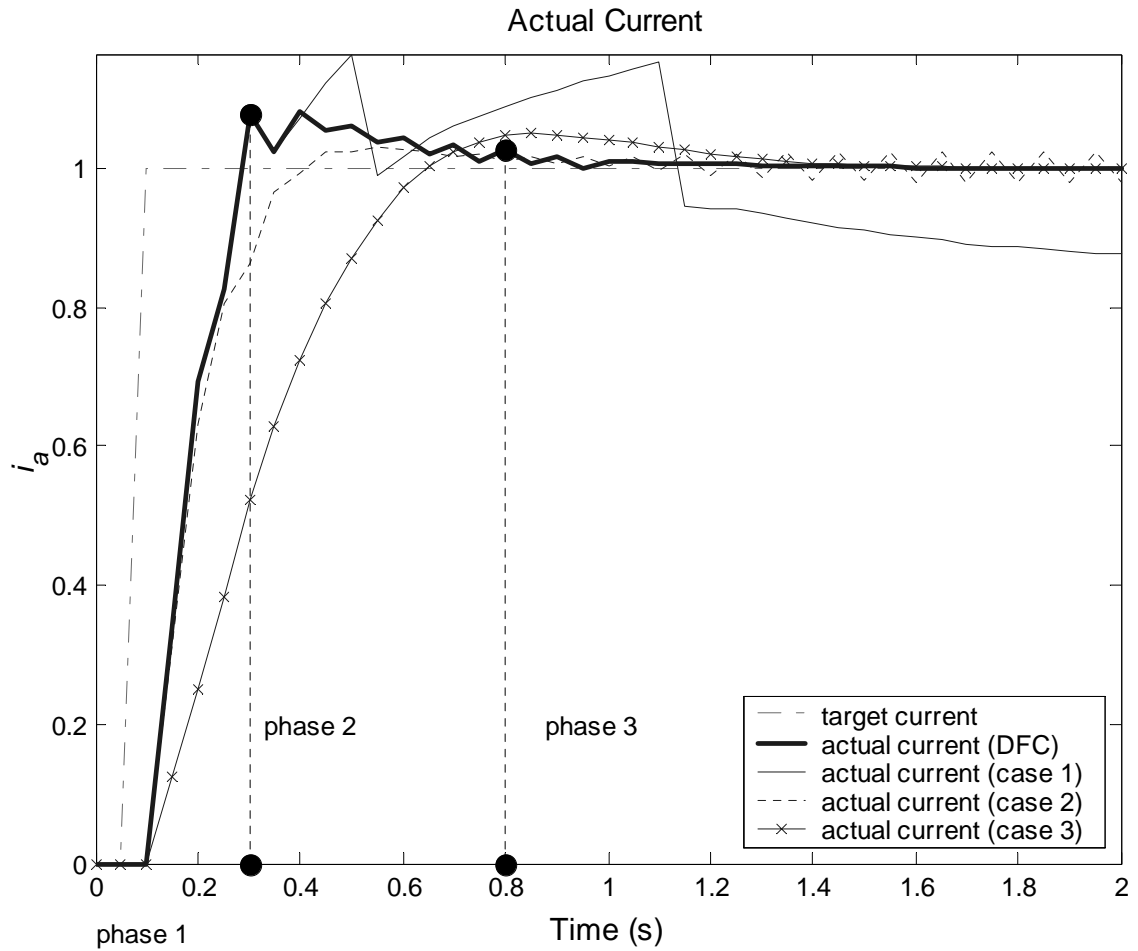
Figure 3.3.1   Response of DFC

However, the DFC will increase the memory overhead and involve more selection issues than the non-dynamic FLC. Its application and performance heavily rely on how the specifications are defined. At this stage, the DFC is used in a unit step response. For other circumstance, new trigger level should be prescribed. Furthermore, its suitability is plant-dependent and improvement to control activity is susceptible to noise.

Chapter 4

Learning and Tutoring

4.1 Self-organization Techniques

As with the previous analysis, the options of the control elements of the FLC are numerous and the selections are difficult to make. On the other hand, the performance of the FLC is plant-related. Due to the plant's complicated characteristics and its exposure to uncertain environments in reality, accurate control activity of the control system is not easy to predict. Another difficulty arises in the human matter: the control pattern of human operators, as engineering knowledge, resists systematic articulation, which is necessary in designing the FLC.

The learning power of the human brain and successful adaptation of species to natural environments evoke the effort to introduce artificial intelligence into the fuzzy controller. The objective is to provide the fuzzy controller with self-organization abilities to find the control elements either by training under supervision or evolving based on fitness in response to a dedicated environment.

In this study, two contemporary self-organization techniques, nerve net and natural selection will be applied in the self-learning of the FLC parameters. Using the fuzzy inference system as a framework, these two learning techniques

are accomplished via soft computation, *i.e.*, backward propagation and genetic algorithms.

According to artificial intelligence convention, these two self-organization techniques can be classified into two broad strategies respectively: tuning and searching strategy.

4.1.1 Tuning Strategy

In fuzzy logic, when the inference system is expressed as a set of interconnected processing elements and the parameters are visualized as the strength of the connection, then the fuzzy inference system can be conceived as a feed forward neural network. To endow the neural network the ability to learn and adapt, the backward propagation algorithm is employed. In this project, the neuro-fuzzy controller, by tuning the parameters of the FLC, works on the principle of function approximation. The training data in the form of a target function are generated by a tutor, which works with the tension control system synchronously.

4.1.2 Searching Strategy

The fuzzy inference system could be treated as an organism characterized by its parameters, which are conceived as genes, and engaged in the competition with rivals. By imitating the natural selection mechanism to search optimal parameters in a dedicated environment, the FLC with appealing performance can survive as the fittest. The evolution of the FLC can be realized through the genetic algorithms, which is an analogy of the evolution pattern in the biological domain, and the artificial environment is built up by the introduction of a tutor as exemplar and evaluation mechanism based on fitness.

4.1.3 Initial Configuration of FLC

For the purpose of testing the tuning and searching strategy, an intelligent tension control system is constructed as illustrated in Figure 4.1.1. The simulation system consists of:

- The target unit which can imitate the behavior of human operators to decide the target current value for the tension control system.

- The tutor unit which is in charge of supervision or construction of artificial environment in self-organization.

- The fuzzy controller unit which can adaptively generate speed reference variance Δv_f for driving motor control system ASR of the rolling mill stand.

- The rolling mill stand unit into which also incorporates the integration function of the control signals from FLC, *i.e.*, the speed reference variance.
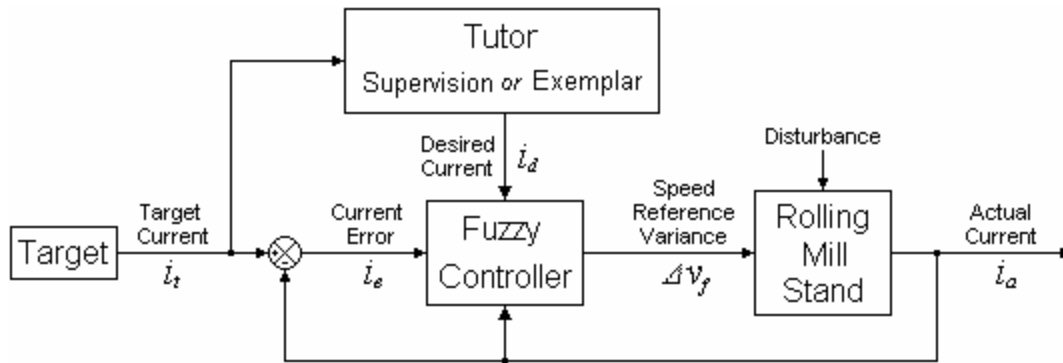
Figure 4.1.1   Intelligent Fuzzy Tension Control System

To compare the performance of these two self-organization techniques, the initial fuzzy inference system is configured as follows for both cases:

- Based on intuition and computation simplicity, the piecewise continuous membership functions, *i.e.*, triangle and trapezoid, and the Sugeno-type fuzzy inference system, *i.e.*, fuzzy singletons for the output linguistic variable, will be employed (Table 4.1.1).

- Product is used for both the aggregation operation of antecedents and implication operation since the backward propagation algorithm needs continuous transfer functions.
- The defuzzification method is weighted average.

Plot	Parameters
Membership Functions	**Input1** i_e **:** Scaling factor: 1 NB: Trapezoidal $\mu(-1, -1, -0.6, -0.3)$ NS: Triangular $\mu(-0.6, -0.3, 0)$ Z: Triangular $\mu(-0.3, 0, 0.3)$ PS: Triangular $\mu(0, 0.3, 0.6)$ PB: Trapezoidal $\mu(0.3, 0.6, 1, 1)$ **Input2** Δi_e **:** Scaling factor: 3 N: Trapezoidal $\mu(-1, -1, -0.5, 0)$ Z: Triangular $\mu(-0.5, 0, 0.5)$ P: Trapezoidal $\mu(0, 0.5, 1, 1)$ **Output** Δv_f **:** Scaling factor: 0.1 NB: -0.75 NM: -0.5 NS: -0.25 Z: 0 PS: 0.25 PM: 0.5 PB: 0.75
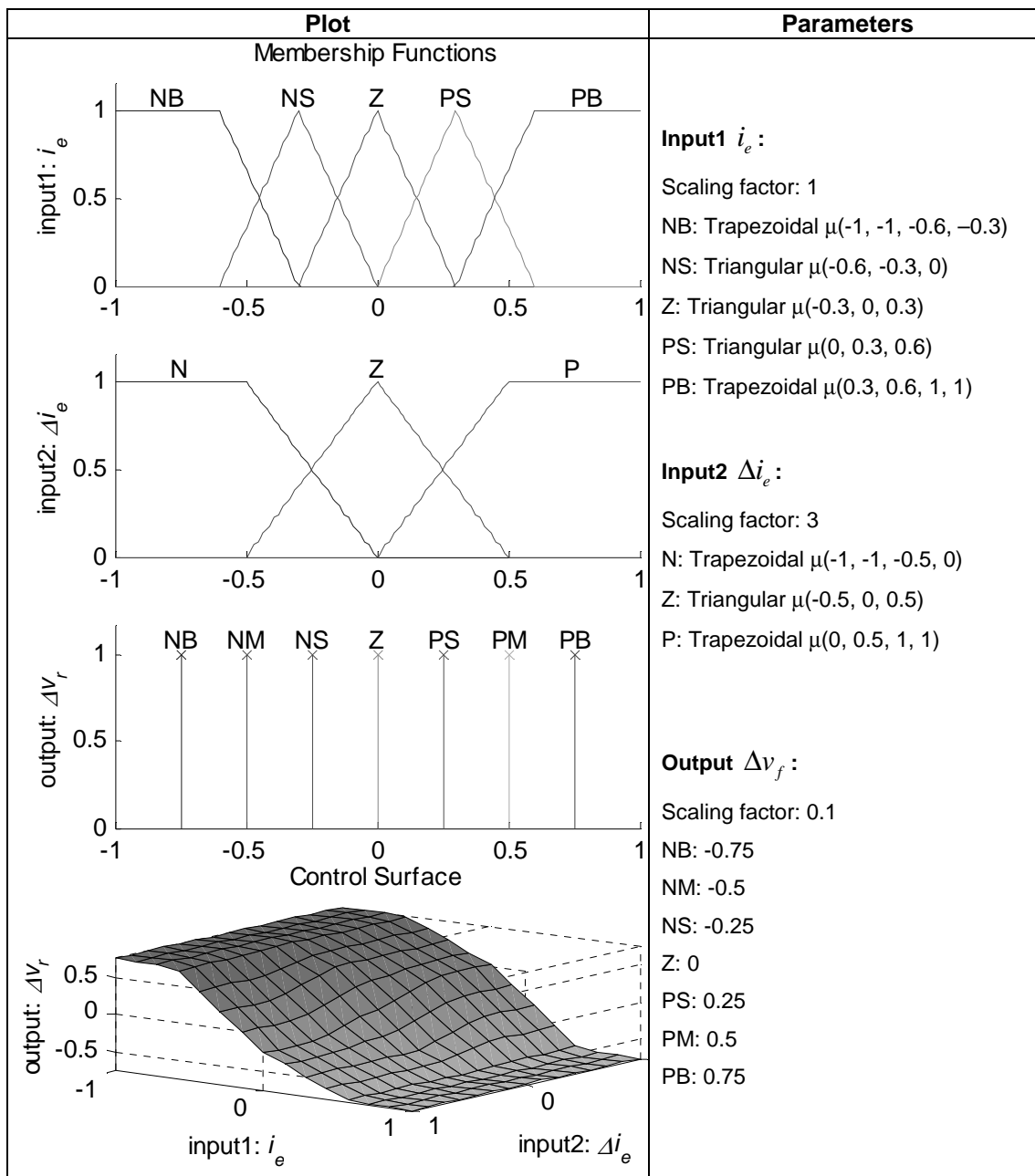
Table 4.1.1   Initial Parameters of FLC

After simulating with these initial settings, the current response and the square error between actual current i_a and desired current i_d, which is created by a tutor (*cf.* Section 4.2), before self-learning are plotted in Figure 4.1.2. An index used to assess the FLC performance is mean square error (MSE) defined as:

$$MSE = \frac{\sum_{k=1}^{n}(i_d(k) - i_a(k))^2}{n} \qquad (4\text{-}1\text{-}1)$$

where:

$n = \dfrac{running\ time}{settling\ time}$ is the sampling intervals.

In this scenario, the target settling time is 0.3s; the simulation runs for a period of 2.5s and the initial MSE is 0.025.
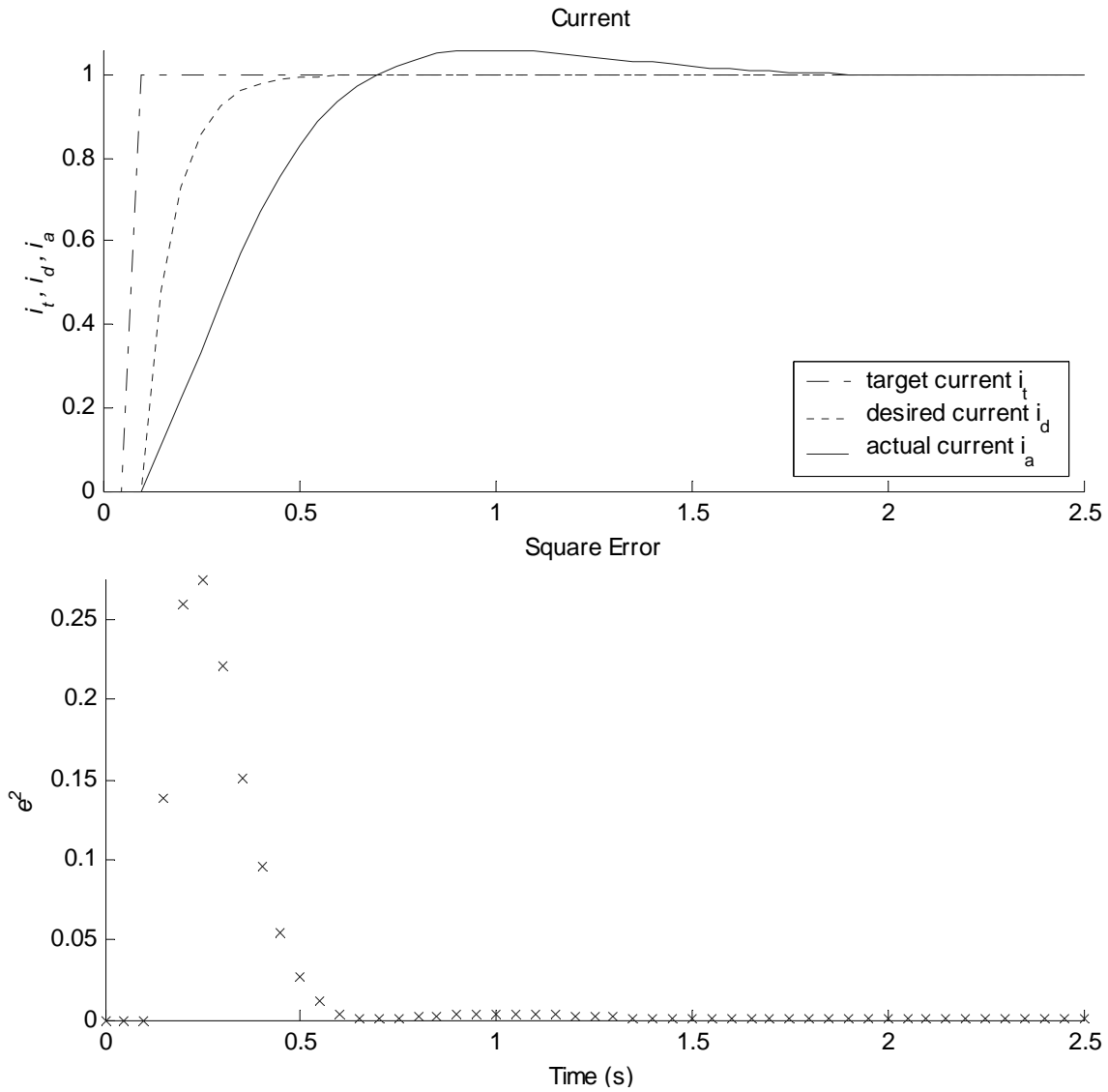
Figure 4.1.2   Response before Self-learning

4.1.4 Parameter Protection

During the tuning or searching, the shapes of membership functions must be kept reasonable for the fuzzy reasoning to proceed. Especially, the following inequalities must be satisfied:

1. $-1 \leq a \leq b \leq c \leq 1$ for the triangular membership function.

2. $-1 \leq a \leq b \leq c \leq d \leq 1$ for the trapezoidal membership function.

On the other hand, based on intuition, the relevant positions of the fuzzy variables should be maintained. Specifically, the following inequalities must be satisfied:

1. $-1 \leq b_{NB} \leq b_{NS} \leq b_Z = 0 \leq b_{PS} \leq b_{PB} \leq 1$ for the fuzzy variables of input1 i_e.

2. $-1 \leq b_N \leq b_Z = 0 \leq b_P \leq 1$ for the fuzzy variables of input2 Δi_e.

3. $-1 \leq NB \leq NM \leq NS \leq Z = 0 \leq PS \leq PM \leq PB \leq 1$ for the fuzzy singletons of output Δv_f.

The observation of these relations will prevent unreasonable tuning or searching that will violate intuition, moreover, can accelerate the parameter self-learning.

4.2 Tutor Selection

Since the training of the FLC needs supervision and the evolving of the FLC needs the evaluation in an environment, a tutor is employed for the following self-organization of the intelligent fuzzy tension controller. In the neuro-fuzzy controller, the tutor creates the desired response waveform into which the neuro-fuzzy controller would fit. In the genetic fuzzy controller, the tutor renders an environment in the form of exemplar of tension control behavior into which the genetic fuzzy controller would adapt.

According to the characteristics of the plant and demands on controller, three kinds of tutors are considered and classified in light of such criteria as overshoot and settling time:

(1)   Ideal tutor:

The ideal tutor is tension reference signal which the control system is expected to track. However, in most cases, such as the step input signal, the initial required tracking speed is infinite, which is not realistic, and thus these training data is not apporporiate as learning objective.

(2) First-order tutor:

If the overshoot is prohibited and the expected final value is 1, a first-order linear system [32]

$$G(s) = \frac{1}{Ts+1} \qquad (4\text{-}2\text{-}1)$$

is utilized to generate the desired tension control activity. The desired current i_d from the tutor is determined by the settling time and computed as:

$$i_d(t) = L^{-1}[\frac{1}{-S_t/log(0.02)s+1}I_t(s)] \qquad (4\text{-}2\text{-}2)$$

where:

S_t is the settling time which is defined as the time required for the output to settle to within 98% of its final value.

(3) Second-order tutor:

When the response speed is important and the overshoot is tolerable, a second-order linear system

$$G(s) = \frac{\omega_n}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad (4\text{-}2\text{-}3)$$

is utilized to generate the desired tension control behavior. According to the request on the settling time and overshoot, the desired current from the tutor is:

$$i_d(t) = L^{-1}[\frac{\varpi_n^2}{s^2 + 2\zeta\varpi_n + \varpi_n^2}I_t(s)] \qquad (4\text{-}2\text{-}4)$$

where:

$\zeta = |logOs|/\sqrt{\pi^2 + (logOs)^2}$ is the damping ratio

$\varpi_n = 16 \times (\pi^2 + (logOs)^2)/(St \times logOs)^2$ is the natural frequency

Os is the overshoot

St is the settling time as defined in Equation (4-2-2)

The sample curves from these three tutors are shown is Figure 4.2.1. The ideal tutor is the unit step signal. The other two curves are the unit step responses from the first and second order linear systems with the settling time 0.3s for both and the overshoot 6% for the latter.
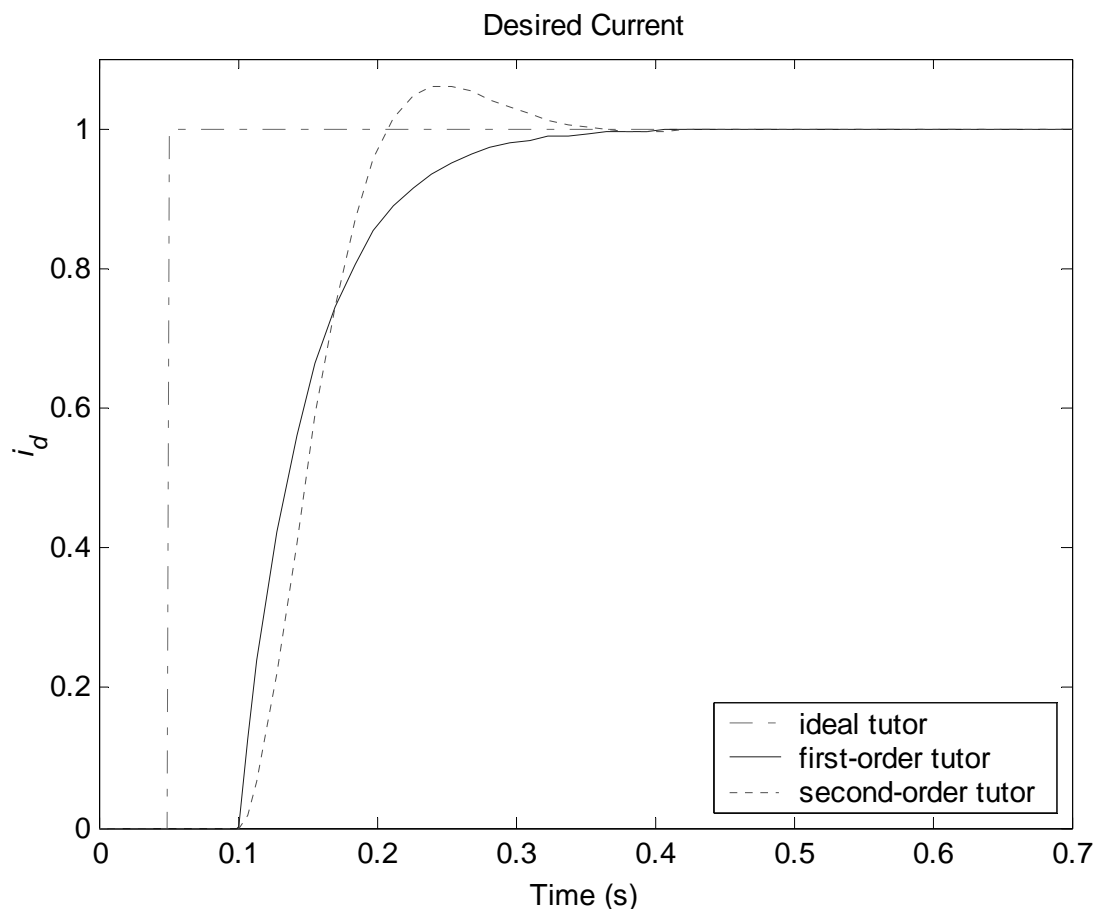
Figure 4.2.1   Desired Current from Tutor

Using the tutor would facilitate the self-learning of the fuzzy controller in four aspects:

(1)   The first and second systems are the ideal templates in the design of controllers. The specifications of the control system can be expressed as

conventional indices, computed from standard formulae and in turn configured easily.

(2)   The tutor shares the same input signal, *i.e.*, target current i_t, as the fuzzy controller. In this way, the tutor can work with the controller synchronously and, especially, offer real-time training data instantaneously for online tuning.

(3)   Since the amount of data for self-learning is determined by the running time and only limited by the sampling time, the instruction from the tutor is full of performance information over the whole response period with high resolution.

(4)   If target current i_t is changed, the instruction from the tutor will be modified accordingly in an autonomous manner, making the intelligent fuzzy tension controller a self-learning system.

For the tension comtrol system, the first-order tutor without overshoot will be used in the forthcoming self-learning.

Chapter 5

Neuro-Fuzzy Tension Controller

In the design of the FLC, lots of parameters, such as the scaling factors and points that define the MF's, are either set conservative to guarantee the stability of the control system or chosen based on intuition due to the lack of structured selection routines. Inspired by the applications of the backward propagation algorithm in neural networks, this self-organization technique is introduced into the design of the FLC to optimize these parameters. In a neuro-fuzzy controller, the fuzzy inference system is viewed as a neural network used to model the tutor's behavior with the same input signal. The weights between nodes and processing elements are fuzzy-based operations, computations and reasoning. The training of network is by resorting to the backward propagation algorithm with supervision [33]. The tunable elements are the scaling factors and membership function parameters for both the input and output signals of the FLC as well as the rule weights.

5.1 Neural Network Structure

To compute the derivatives in the backward propagation algorithm, the aggregation operation of antecedents is product and the membership functions for the output linguistic variable are fuzzy singletons, *i.e.*, the Sugeno-type fuzzy inference system is used. With this setting, the min and product implication operations are equivalent and expressed as min due to computation simplicity.

When the fuzzy logic controller is configured as in Chapter 4, the artificial version of the neural network can be constructed as Figure 5.1.1. The chief structural difference from the conventional artificial neural network is in its architecture, *i.e.*, the node and layer numbers, are determined by the fuzzy inference system, and the bonds among nodes, *i.e.*, the signal weighting, collecting and processing, are the parameters and operations for fuzzy reasoning.
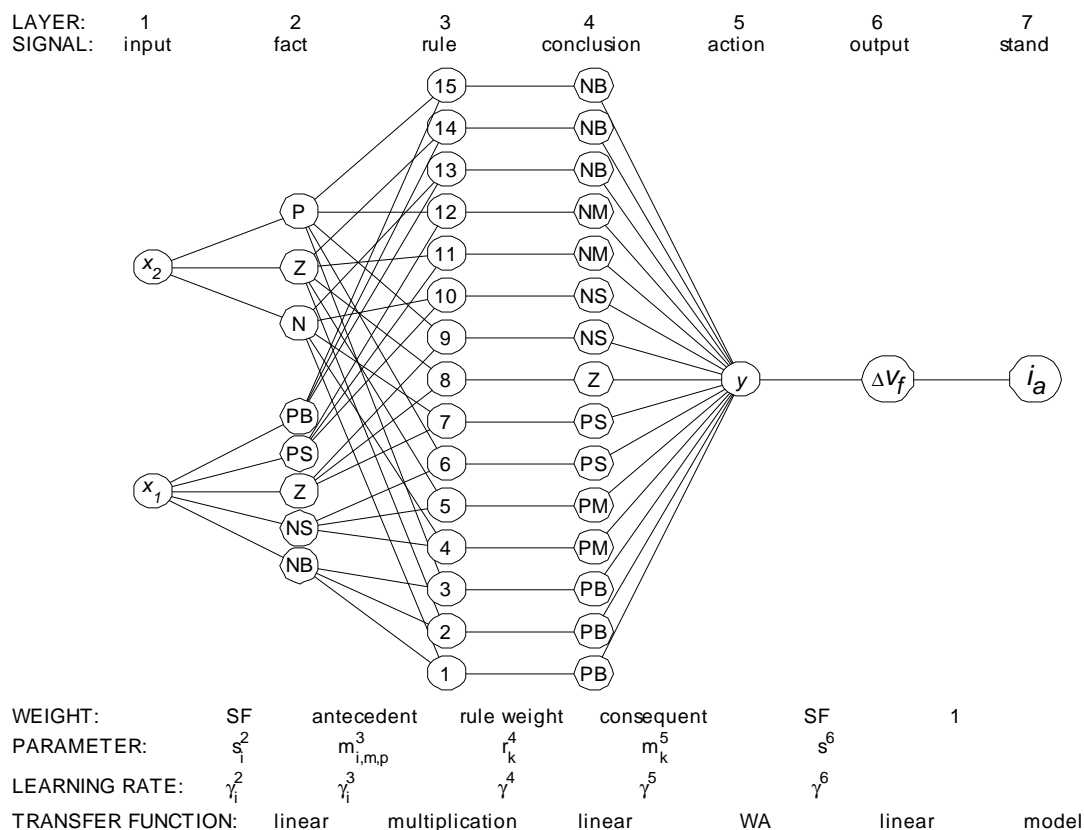
Figure 5.1.1   Neuro-fuzzy Controller Structure

Layer 1 is the input layer with 2 nodes. The inputs are input1 $x_1 = i_e(t)$, *i.e.*, current error, and input2 $x_2 = \Delta i_e(t)$, *i.e.*, current error variance. This layer receives and transmits the measures of the input signals.

Layer 2 is the fact layer with 8 nodes. The weights are the scaling factors for input1 i_e and input2 Δi_e. The transfer functions are linear. The outputs of this layer are the input signals mapped into the universe of discourse [-1, 1].

Layer 3 is the rule layer with 5×3 = 15 nodes. The weights are antecedents of the fuzzy rules. Different from conventional neural network, the combination of weighted inputs is the aggregation operation of antecedent, *i.e.*, product, instead of summation. The transfer functions are linear and the outputs of this layer are the firing strengths.

Layer 4 is the conclusion layer with 15 nodes. The connection strengths are the rule weights. The transfer functions are linear. The output of this layer is the weighted firing strengths.

Layer 5 is the action layer with one node. Since the zero-order Sugeno-type fuzzy reasoning system is employed, the weights are the consequents of the fuzzy rule, *i.e.*, the fuzzy singletons for the output linguistic variable. The output of this layer is the crisp control action computed by the defuzzification method, *i.e.*, WA.

Layer 6 is the output layer with one node. The weight is the scaling factor for output Δv_f and the transfer function is linear. The output signal is speed reference variance for ASR.

Layer 7 is the rolling mill stand layer with one node. The weight for this layer is 1. The transfer function is the unknown mathematical model of the rolling mill stand.

The data in the neural network flows in two directions: the control signal flows forwards from layer 1 to layer 7 and the error signal streams backwards from layer 7 to layer 1.

5.2 Forward Propagation Algorithm

In neuron-based fuzzy inference system, the reasoning procedure can be articulated by the forward propagation algorithm.

(1) Input layer

This layer receives the external input signals, its outputs are:

$$y_1^1(t) = x_1(t) = i_e(t) \tag{5-2-1}$$

$$y_2^1(t) = x_2(t) = \Delta i_e(t) \tag{5-2-2}$$

(2)   Fact layer

The layer inputs are weighted by the scaling factors for input1 i_e and input2 Δi_e respectively. The node net input is:

$$x_i^2(t) = s_1^2(t) \times y_1^1(t) \tag{5-2-3}$$

where:

i = 1,..., 5.

$s_1^2(t)$ is the scaling factor for input1 i_e.

$$x_j^2(t) = s_2^2(t) \times y_2^1(t) \tag{5-2-4}$$

where:

j = 6,..., 8.

$s_2^2(t)$ is the scaling factor for input2 Δi_e.

With the linear transfer function, the layer output is:

$$y_k^2(t) = x_k^2(t) \tag{5-2-5}$$

where:

k = 1,..., 8.

(3)   Rule Layer

The layer inputs are weighted with membership grades and combined through aggregation operation of antecedents. The node net inputs are:

$$x_k^3(t) = \mu_{1,i}^3(y_1^2(t), m_{1,i}^3(t)) \times \mu_{2,j}^3(y_2^2(t), m_{2,j}^3(t)) \qquad (5\text{-}2\text{-}6)$$

Where:

$\mu_{1,i}^3(y_1^2(t), m_{1,i}^3(t))$ is the degree of $y_1^2(t)$ in the ith fuzzy variable, which is designated by the kth rule, with MF parameters $m_{1,i}^3(t)$ for input1 i_e.

$\mu_{2,j}^3(y_2^2(t), m_{2,j}^3(t))$ is the degree of $y_2^2(t)$ in the jth fuzzy variable, which is designated by the kth rule, with MF parameters $m_{2,j}^3(t)$ for input2 Δi_e.

With the linear transfer function, the layer outputs, *i.e.*, the firing strengths, are:

$$y_k^3(t) = x_k^3(t) \qquad (5\text{-}2\text{-}7)$$

where:

k = 1, ..., 15.

(4)   Conclusion Layer

The layer inputs are weighted by rule weights and every node has one receiver. The node net inputs are:

$$x_k^4(t) = r_k^4(t) \times y_k^3(t) \qquad (5\text{-}2\text{-}8)$$

where:

$r_k^4(t)$ is the weight in the kth rule.

With the linear transfer function, the layer outputs are:

$$y_k^4(t) = x_k^4(t) \tag{5-2-9}$$

where:

k = 1, ..., 15.

(5)   Action Layer

The layer inputs are weighted by the fuzzy singletons for the output linguistic variable and summed up. The node net input is:

$$x^5(t) = \sum_{j=1}^{15} (m_i^5(t) \times y_j^4(t)) \tag{5-2-10}$$

where:

m_j^5 is the jth fuzzy singleton, which is designated by the jth rule.

With the defuzzification method WA as the transfer function, the layer output, *i.e.*, the crisp control action, is calculated from:

$$y^5(t) = \frac{x^5(t)}{\sum_{j=1}^{15} y_j^4(t)} \tag{5-2-11}$$

(6)   Output Layer

The crisp control action is weighted by the scaling factor for output Δv_f :

$$x^6(t) = s^6(t) \times y^5(t) \tag{5-2-12}$$

Where:

$s^6(t)$ is the scaling factor for output Δv_f .

With the linear transfer function, the layer output, *i.e.*, the control signal from the FLC, is:

$$y^6(t) = x^6(t) \tag{5-2-13}$$

(7)   Stand Layer

The output of this layer is actual current i_a determined by the characteristics of the rolling mill stand:

$$y^7(t) = y^a(t) = i_a(t) \tag{5-2-14}$$

Meanwhile, the tutor makes use of the same target current $i_t(t)$ and gives desired current $y^d(t) = i_d(t)$.

5.3 Backward Propagation Algorithm

With every pair of actual current $i_a(t) = y^a(t)$ and desired current $i_d(t) = y^d(t)$ as the training data, the FLC parameters are adjusted in the reverse layer to reduce the performance index. For every node, the error signal propagates from the last layer to the first layer and can be computed from sensitivity, which is defined as [33]:

$$J_n^l(t) = \frac{\partial P(i_d, i_a)}{\partial x_n^l(t)} \tag{5-3-1}$$

where:

$P(i_d, i_a)$ is the performance index.

$x_n^l(t)$ is the net input of the nth node in the lth layer, *i.e.*, the input of the transfer function.

5.3.1 Performance Index

Limited by the availiability of the probability distribution characteristics and in turn the expectation of the training data, the performance index is approximated by:

$$P(i_d, i_a) = e^2(t) \tag{5-3-2}$$

where:

$e(t) = y^d(t) - y^a(t)$ is the error between actual current $y^a(t) = i_a(t)$ and desired

current $y^d(t) = i_d(t)$.

And the parameter, as weight, is updated as:

$$W_{n,c}^l(t+1) = W_{n,c}^l(t) - \gamma_n^l \times \frac{\partial J_n^l(t)}{\partial W_{n,c}^l(t)} \tag{5-3-3}$$

where:

$W_{n,c}^l$ is the cth weight of the nth node in the lth layer.

γ_n^l is the learn rate for the nth node in the lth layer.

5.3.2 Parameter Updating

(1)   Stand layer

Sensitivity is:

$$J^7(t) = \frac{\partial P(i_d, i_a)}{\partial x^6(t)} = \frac{\partial e^2(t)}{\partial \Delta v_r(t)} = -2 \times e(t) \times \frac{\partial i_a(t)}{\partial \Delta v_r(t)} \tag{5-3-4}$$

There is no parameter to be tuned.

(2)   Output layer

The sensitivity is

$$J^6(t) = 1 \times 1 \times J^7(t) \tag{5-3-5}$$

The scaling factor for output Δv_f is updated as:

$$s^6(t+1) = s^6(t) - \gamma^6 \times J^6(t) \times y^5(t) \tag{5-3-6}$$

where:

γ^6 is the learning rate of the scaling factor for output Δv_f.

(3)   Action layer

The sensitivity is:

$$J^5(t) = \frac{1}{\displaystyle\sum_{j=1}^{15} y_j^4(t)} \times s^6(t) \times J^6(t) \tag{5-3-7}$$

The output fuzzy singletons, *i.e.*, NB, NM, NS, Z, PS, PM and PB, are updated as:

$$m_k^5(t+1) = m_k^5(t) - \gamma^5 \times J^5(t) \times \overline{y_j^4(t)} \tag{5-3-8}$$

where:

k = 1, ..., 7.

$\overline{}$ is the average operation over the rules with the k$^{\text{th}}$ fuzzy singleton in conclusion.

γ^5 is the learning rate of the fuzzy singleton for the output linguistic variable.

(4)   Conclusion layer

The sensitivities are:

$$J_k^4(t) = 1 \times m_i^5(t) \times J^5(t) \qquad \text{(5-3-9)}$$

The rule weights are updated as:

$$r_k^4(t+1) = r_k^4(t) - \gamma^4 \times J_k^4(t) \times y_k^3(t) \qquad \text{(5-3-10)}$$

where:

k = 1, ..., 15.

γ^4 is the learning rate of the rule weights.

(5) Rule layer

The sensitivities are:

$$J_i^3(t) = 1 \times r_i^4(t) \times J_i^4(t) \qquad \text{(5-3-11)}$$

Where:
i = 1, ..., 15.

The adjustable connection weights in this layer are the MF parameters $m_{i,j}^3(t)$ of the jth fuzzy variable for the ith input signal, which include a, b and c for the triangular MF's or a, b, c and d for the trapezoidal MF's.

The fuzzy variables that input1 i_e can take include NB, NS, Z, PS and PB. For the kth fuzzy variable, one of its parameter p is updated as:

$$m_{1,k,p}^3(t+1) = m_{1,k,p}^3(t) - \gamma_1^3 \times \frac{\partial \mu_{1,k}^3(y_1^2(t), m_{1,k}^3(t))}{\partial m_{1,k,p}^3(t)} \times \overline{J_i^3(t) \times \mu_{2,j}^3(y_2^2(t), m_{2,j}^3(t))} \qquad \text{(5-3-12)}$$

where:

k = 1, ..., 5.

γ_1^3 is the learning rate of input1 i_e.

$\overline{}$ is the average operation over the rules with the k$^{\text{th}}$ fuzzy variable in premise.

The fuzzy variables that input2 Δi_e can take include N, Z and P. For the k$^{\text{th}}$ fuzzy variable, one of its parameter p is updated as:

$$m_{2,k,p}^3(t+1) = m_{2,k,p}^3(t) - \gamma_2^3 \times \frac{\partial \mu_{2,k}^3(y_2^2(t), m_{2,k}^3(t))}{\partial m_{2,k,p}^3(t)} \times \overline{J_i^3(t) \times \mu_{1,j}^3(y_1^2(t), m_{1,j}^3(t))} \qquad (5\text{-}3\text{-}13)$$

where:

k = 1, ..., 3.

γ_2^3 is the learning rate of input2 Δi_e.

$\overline{}$ is the average operation over the rules with the k$^{\text{th}}$ fuzzy variable in premise.

(6)   Fact layer

For input1 i_e, the sensitivities are:

$$J_{1,k}^2(t) = \sum_{i=1}^{3} (1 \times \frac{\partial \mu_{1,k}^3(y_1^2(t), m_{1,k}^3(t))}{\partial y_1^2(t)} \times \mu_{2,i}^3(y_2^2(t), m_{2,i}^3(t)) \times J_j^3(t)) \qquad (5\text{-}3\text{-}14)$$

where:

k = 1, ..., 5.

For input2 Δi_e, the sensitivities are:

$$J_{2,k}^2(t) = \sum_{i=1}^{5} (1 \times \frac{\partial \mu_{2,k}^3(y_2^2(t), m_{2,k}^3(t))}{\partial y_2^2(t)} \times \mu_{1,i}^3(y_1^2(t), m_{1,i}^3(t)) \times J_j^3(t)) \qquad (5\text{-}3\text{-}15)$$

where:

k = 1, ..., 3.

The scaling factor for input1 i_e is updated as:

$$s_1^2(t+1) = s_1^2(t) - \gamma_1^2 \times \overline{J_{1,k}^2(t)} \times y_1^1(t)) \qquad (5\text{-}3\text{-}16)$$

where:

γ_1^2 is the learning rate of the scaling factor for input1 i_e.

$\overline{}$ is the average operation over the rules with antecedents input1 i_e.

The scaling factor for input2 Δi_e is updated as:

$$s_2^2(t+1) = s_2^2(t) - \gamma_2^2 \times \overline{J_{2,k}^2(t)} \times y_2^1(t)) \qquad (5\text{-}3\text{-}17)$$

where:

γ_2^2 is the learning rate of the scaling factor for input2 Δi_e.

$\overline{}$ is the average operation over the rules with antecedents input2 Δi_e.

5.4 Parameter Restriction

The FLC can be configured such that the fuzzy-based neural network is able to fit the desired activity waveform. However, since the amount of the training data is finite, two important issues in the training of the NFC must be addressed: generalization and overfit. Especially, the overfit can cause undesirable control activity and results in oscillation and final error. However, the structure of the network is stiff, *i.e.*, the node and layer numbers are determined by the fuzzy inference system. Preventive measures should be taken in case of the occurrence of irrational parameter tuning. This work circumvents these two problems from a response specification perspective: the protections are provided for the FLC parameters during tuning to ensure stability and final steady-state precision. Provided that the triangle/trapezoid membership functions are employed for the

input/output linguistic variables, it is observed that the parameters c of NS, d of N, b of Z, a of P and a of PS regulate the steady-state response. If the initial values are chosen as 0's, then these parameters are not supposed to be tuned.

Apart from these measures, other ways to ensure neural network's successful self-learning include deliberately designating learning rates and termination condition such as error tolerance as well as, in offline tuning, epochs to avoid overfit. With these efforts, the requirements of the parameter protection, as mentioned in Chapter 4 can also be fulfilled.

5.5 Offline Tuning

5.5.1 Simulation Configuration

In the following tuning, both the tension control system and the tutor run for 2.5 seconds using a unit step signal as target current i_t, and thus 50 pairs of [actual current i_a, desired current i_d] data are generated for training. The initial configuration and parameters of the FLC are described in Chapter 4. The tuning of the rule weights is turned off and the nodes in the rule layer are treated as dead neurons because their validity is questionable in preliminary tests.

In addition to the parameter restriction, two additional termination conditions are set to ensure the final convergence in the offline tuning. When either the prescribed epoch number or the error tolerance is reached, the tuning terminates.

The offline tuning procedure can be depicted as follows [34]:

```
//Neuro-fuzzy controller offline learning
Run simulation
Compute initial MSE
```

```
                    i = 1
                    while  (MSE >= error tolerance) and (i <= epochs)
                            Start simulation
                            while simulation continuing
                                    //forward propagation algorithm
                                    for l = 1 to 7
                                            Compute the node net inputs of layer l
                                            Compute the node outputs of layer l
                                    end l
                                    Collect the actual and desired output data
                                    Compute the performance index
                                    //Backward propagation algorithm
                                    for l = 7 to 1
                                            Compute the node sensitivities of layer l
                                            Update parameters
                                    end l
                            end while
                            Compute the MSE
                            l+ =1
                    end while
```

5.5.2 Tuning Result

Via trial-and-error, two groups of tuning parameters with satisfactory tuning results are selected as listed in Table 5.5.1 and Table 5.5.2 respectively.

Settling Time(s)	Running Time (s)	Epochs	Error Tolerance	Learning Rate						
				SF for Input1 i_e	SF for Input2 Δi_e	MF's for Input1 i_e	MF's for Input2 Δi_e	Rule Weights	MF's for Output Δv_f	SF for Output Δv_f
0.3	2.5	45	0.001	3	30	0.1	1	0	0.01	0.3

Table 5.5.1   Offline Tuning Parameters (Case 1)

Settling Time(s)	Running Time (s)	Epochs	Error Tolerance	Learning Rate						
				SF for Input1 i_e	SF for Input2 Δi_e	MF's for Input1 i_e	MF's for Input2 Δi_e	Rule Weights	MF's for Output Δv_f	SF for Output Δv_f
0.3	2.5	45	0.001							
				0.5	15	0.1	1	0	0.05	0.3

Table 5.5.2   Offline Tuning Parameters (Case 2)

Parameters (case 1)	Parameters (case 2)
Input1 i_e:	**Input1 i_e:**
Scaling factor: 1.3603	Scaling factor: 1.0827
NB: Trapezoidal μ(-1, -1, -0.6, −0.3)	NB: Trapezoidal μ(-1, -1, -0.6, −0.3)
NS: Triangular μ(-0.6, -0.2799, 0)	NS: Triangular μ(-0.6, -0.3042, 0)
Z: Triangular μ(-0.3, 0, 0.2907)	Z: Triangular μ(-0.2999, 0, 0.2910)
PS: Triangular μ(0, 0.2926, 0.6049)	PS: Triangular μ(0, 0.2969, 0.6112)
PB: Trapezoidal μ(0.2749, 0.5857, 1, 1)	PB: Trapezoidal μ(0.2525, 0.5669, 1, 1)
Input2 Δi_e:	**Input2 Δi_e:**
Scaling factor: 2.3511	Scaling factor: 2.4354
N: Trapezoidal μ(-1, -1, -0.4518, 0)	N: Trapezoidal μ(-1, -1, -0.4133, 0)
Z: Triangular μ(-0.6011, 0, 0.5012)	Z: Triangular μ(-0.6696, 0, 0.5010)
P: Trapezoidal μ(0, 0.5034, 1, 1)	P: Trapezoidal μ(0, 0.5033, 1, 1)
Output Δv_f:	**Output Δv_f:**
Scaling factor: 0.2490	Scaling factor: 0.2862
NB: -0.7517	NB: -0.7705
NM: -0.5013	NM: -0.5150
NS: -0.2251	NS: -0.2897
Z: 0	Z: 0
PS: 0.2499	PS: 0.2428
PM: 0.5035	PM: 0.4987
PB: 0.75	PB: 0.75

Table 5.5.3   Parameters of MF's after Offline Tuning
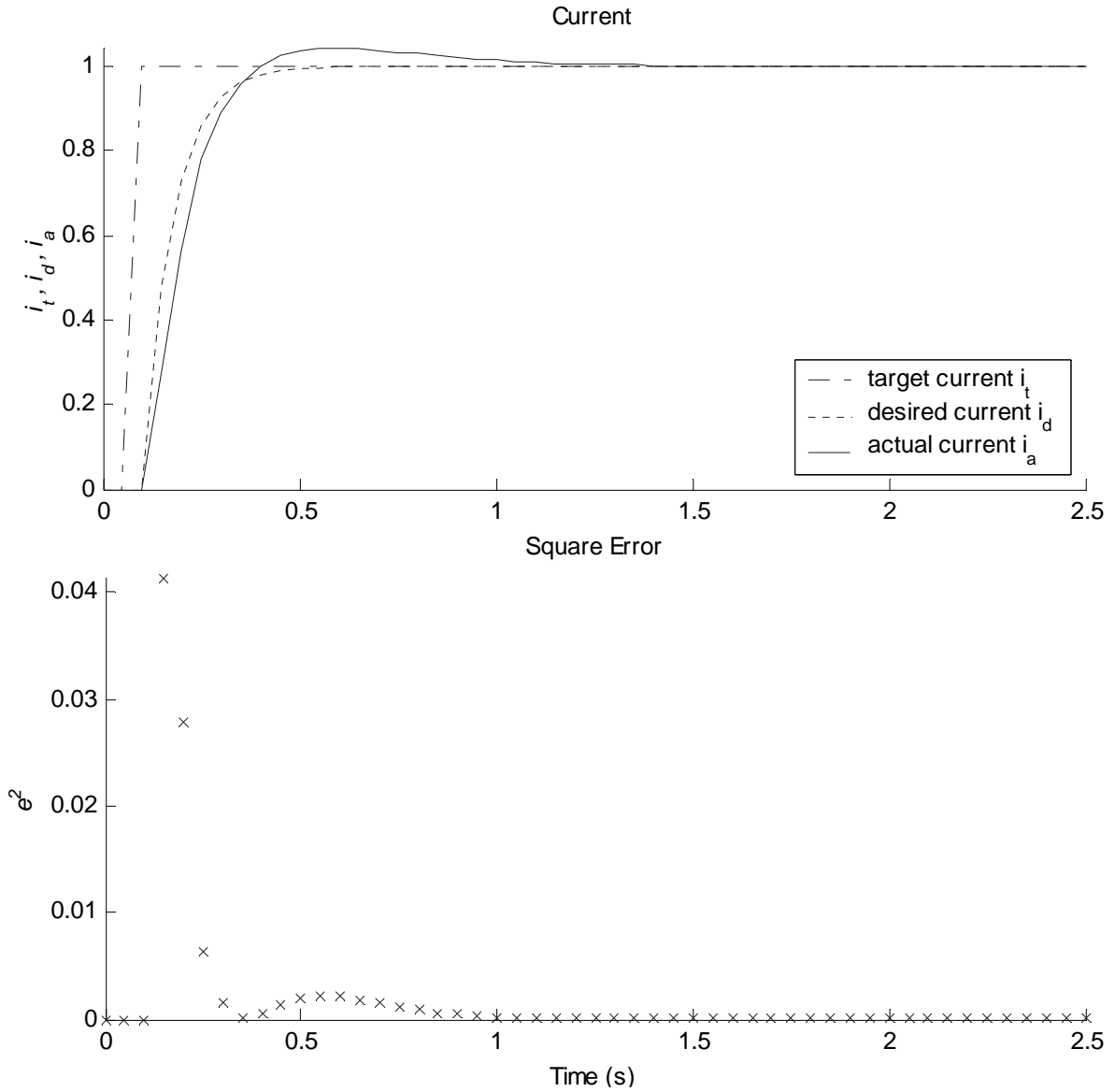
Figure 5.5.1   Response after Offline Tuning (Case 1)
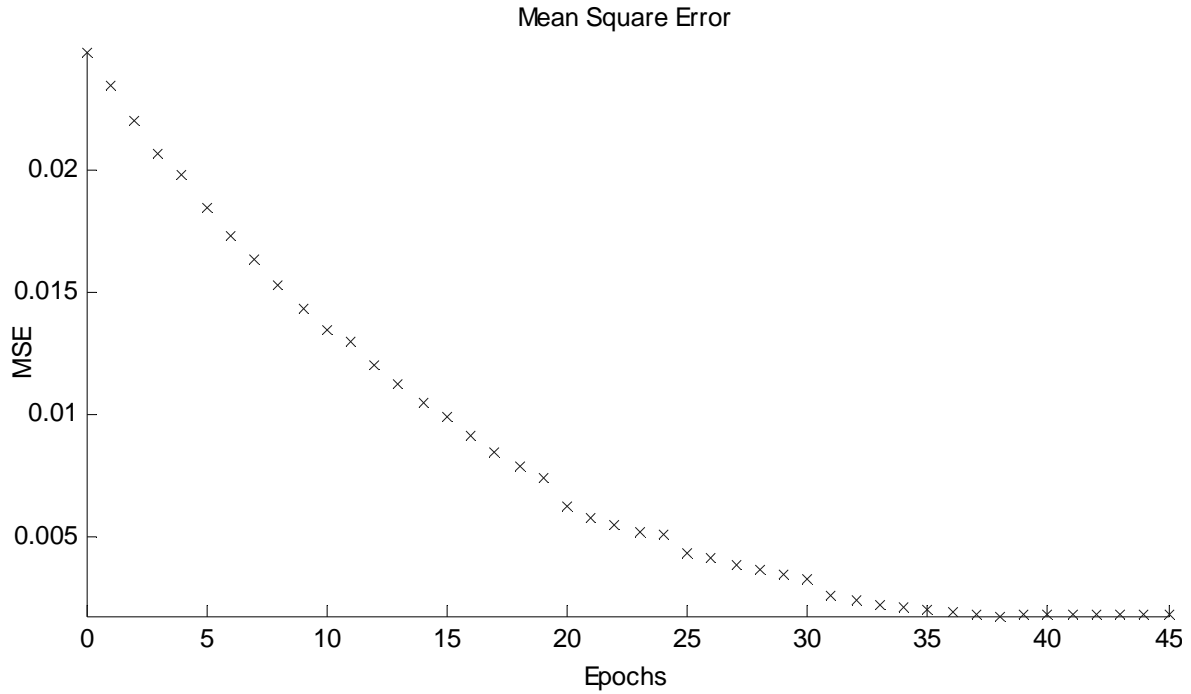
Figure 5.5.2   Mean Square Error of Offline Tuning (Case 1)

As case 1, the parameters of the FLC after tuning are listed in the first column of Table 5.5.3. The response of tension after tuning is plotted in Figure 5.5.1. The MSE plotted in Figure 5.5.2 shows that the backward propagation algorithm reduces the error all at each epoch under the tutor's guidance, and the tuning ceases at iteration 45 with MSE decreasing from initial 0.025 (Figure 4.1.2) to final 0.0018.
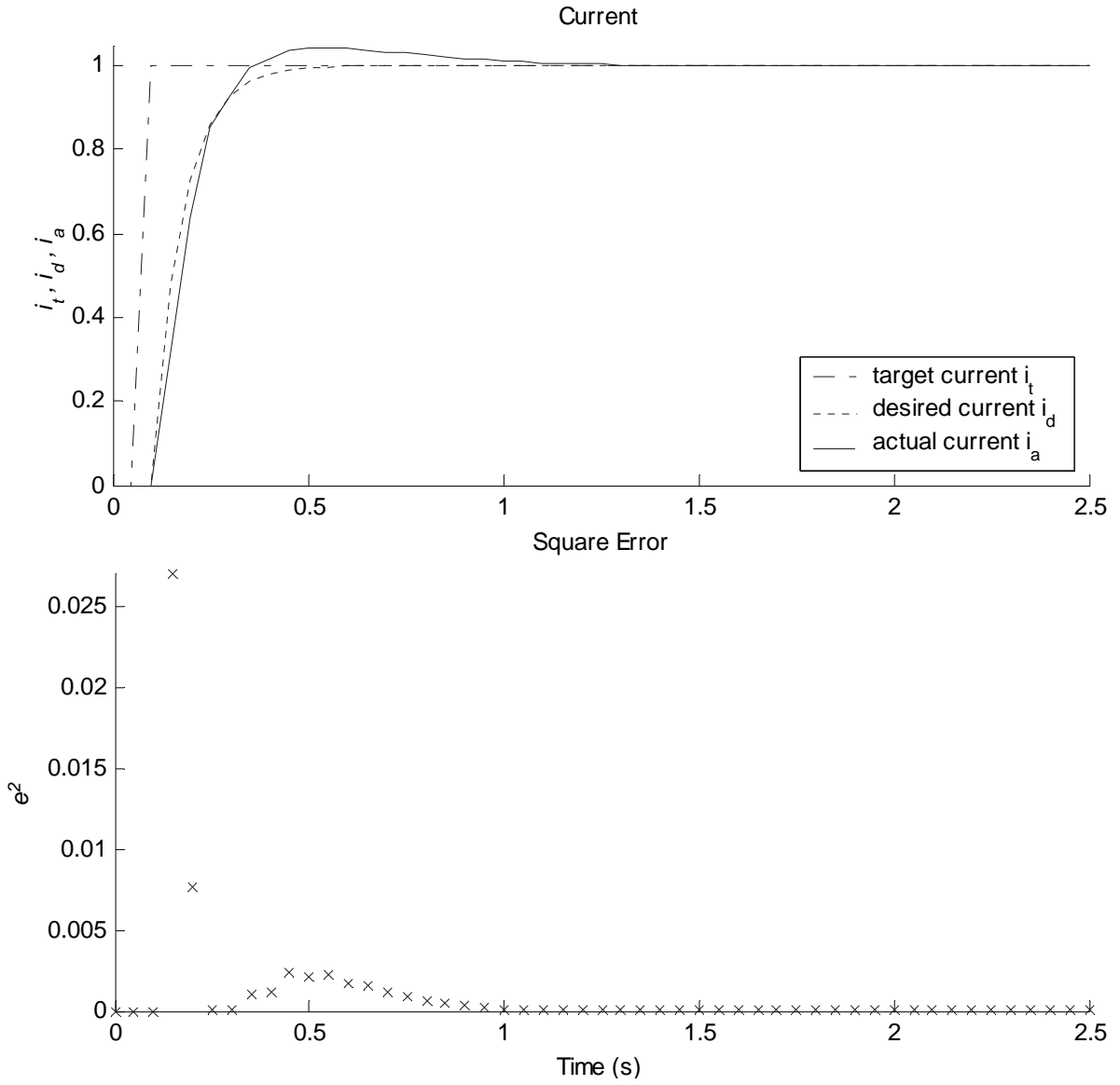
Figure 5.5.3   Response after Offline Tuning (Case 2)
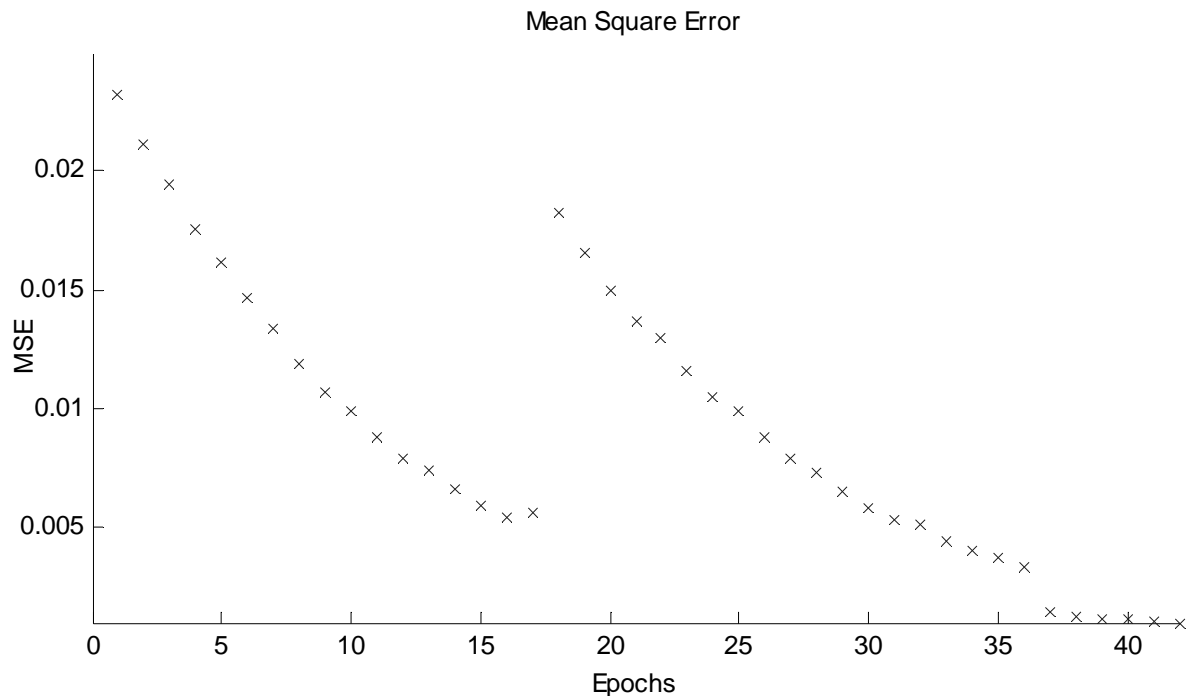
Figure 5.5.4   Mean Square Error of Offline Tuning (Case 2)

As case 2, the parameters of the FLC after tuning are listed in the second column of Table 5.5.3. The response of the tension after tuning is plotted in Figure 5.5.3. The MSE in Figure 5.5.4 demonstrates that the tuning process is not smooth, which stems from the bigger learning rates. The tuning is stopped by error tolerance condition when the MSE is equal to 0.00099684 at iteration 42.

5.6 Online Tuning

As a pre-tuning technique, the success of the offline learning depends on the representativity of the training data. The time-varying rolling behavior, uncertain operation conditions and the unpredictable target signal demand the real-time self-learning capability for the fuzzy tension controller. The main superiority of the backward propagation algorithm lies in its online application at a minor computational expense.

In the following investigation, the control elements and initial parameters of the FLC are configured as in Chapter 4; the termination condition is set through the error tolerance only. In reality, when a rolling stand enters normal mode, the human operator sets the reference, increases or decreases, for the ASR to manipulate the tension. This control signal is transferred into ramp function by an actuator. In simulation systems, this control pattern can by approximated by using a periodical trapezoidal signal as target current i_t. The trapezoidal tension response without tuning and square error between actual current i_a and desired one i_d are plotted in Figure 5.6.1. Initially, the FLC is set such that it is robust and has satisfactory tracking ability.
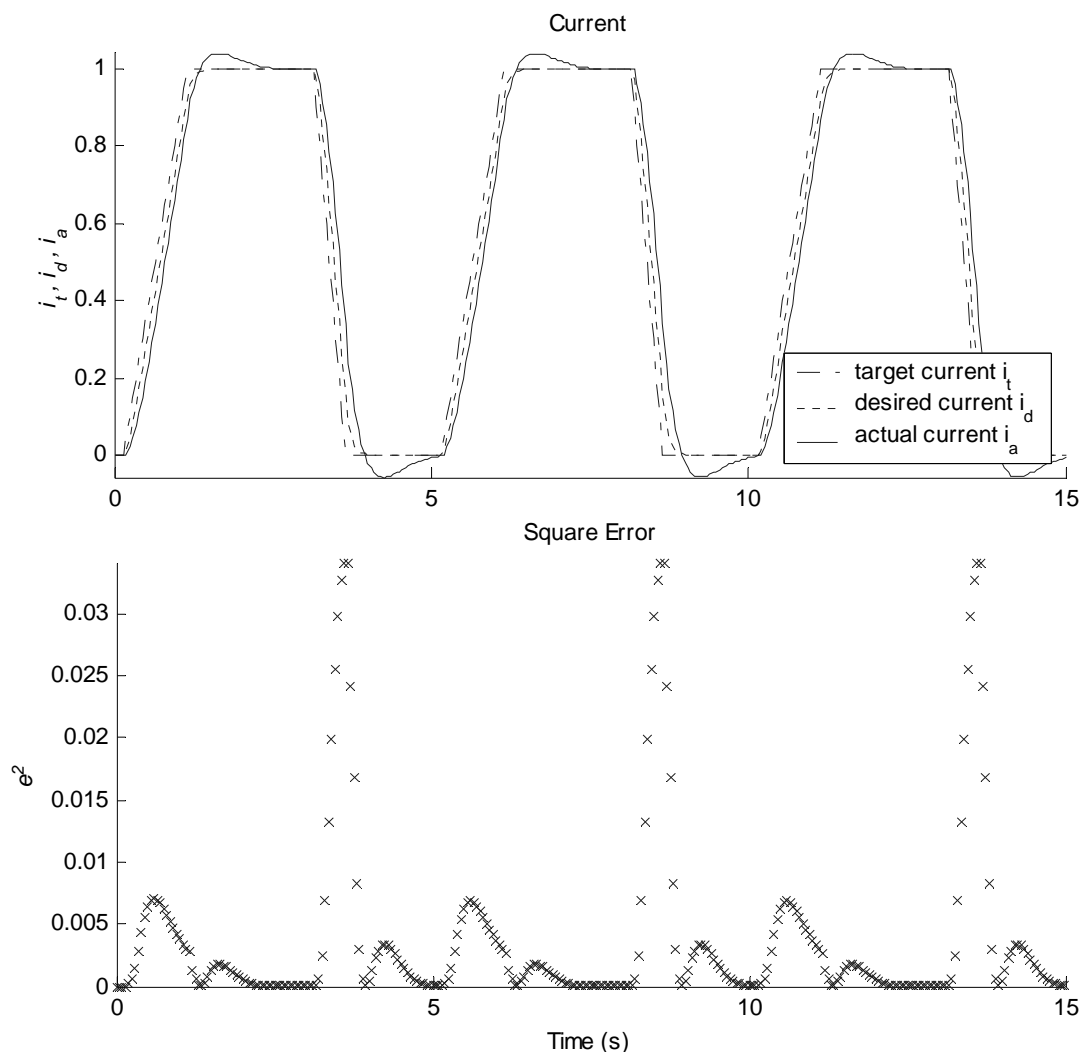
Figure 5.6.1   Response before Online Tuning

5.6.1 Simulation Configuration

The possibility of unstable learning from voluminous training data put a severe strain on the utilization of the NFC. In this respect, the instantaneous comparison of the error tolerance and the square error between the actual and desired current, *i.e.*, the performance index, at sampling instant is used as the termination condition for tuning.

The online tuning procedure can be depicted as follows [35]:

```
//Neuro-fuzzy controller online learning
Start simulation
while simulation continuing
        //forward propagation algorithm
        for l = 1 to 7
                Compute the node net inputs of layer l
                Compute the node outputs of layer l
        end l
        Collect the actual and desired output data
        Compute the performance index
        if (performance index >= error tolerance)
                //Backward propagation algorithm
                for l = 7 to 1
                        Compute the node sensitivities of layer l
                        Update parameters
                end l
        end if
end while
```

The online tuning needs elaborate parameter selections and tests. By trial-and-error, a group of tuning parameters is chosen as listed in Table 5.6.1. Similar to the off-line tuning, the rule weight tuning is turned off and the nodes in the rule layer are treated as dead neurons.

Settling time(s)	Error Tolerance	Learning rate						
		SF for Input1 i_e	SF for Input2 Δi_e	MF's for Input1 i_e	MF's for Input2 Δi_e	Rule Weights	MF's for Output Δv_f	SF for Output Δv_f
0.3	0.015	25	75	3	7.5	0	0.25	0.5

Table 5.6.1   Online Tuning Parameters

5.6.2 Tuning Result

The simulation is conducted for 60 seconds, *i.e.*, 1200 pairs of training data are generated; it is also assumed that the model of the rolling mill stand is time varying, *i.e.*, the coefficient a_1 (Section 2.1.5) increases 10% at the 20th second. It can be seen from the sweep chart (Figure 5.6.2) that the square error is reduced within one cycle after simulation system starting and within 7 cycles after the occurrence of a coefficient drift. As precision is reached, *i.e.*, the performance is less than the error tolerance = 0.015, the tuning is suspended and the rolling process continues with maximum square error no bigger than the error tolerance.

In both the online and offline tuning, the initial configuration of the FLC and the learning rates have strong influence on the learning performance of the NFC and their selections deserve scrutiny. The resulting parameters can be only locally optimal. This could be verified by the multiple local minima of MSE in the rugged tuning surface as roughly visualized in the tuning space of the scaling factors (Figure 5.6.3). For a stable learning, the initial FLC parameters should be set away from the local minima. In this simulation, the initial SF's are put in the flat part, *i.e.*, the far right side of the contour plot, and thus, not only can the global minima be approached, but also the big learning rates can be used. One of the more discernible properties is the interaction between the learning rates: increasing or decreasing a learning rate will magnify or minify the adjustment of adjacent layer weights, which, likewise, results from the irregularity of the tuning space.
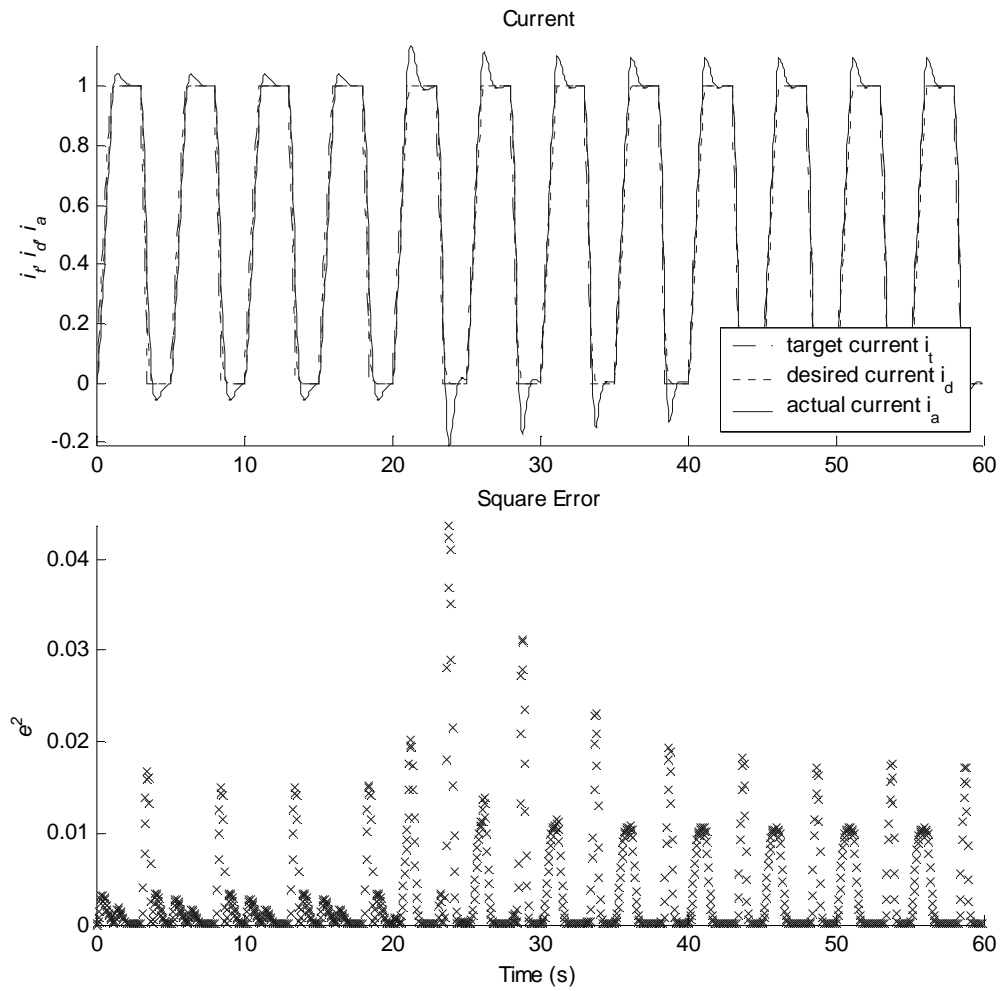
Figure 5.6.2   Response during Online Tuning
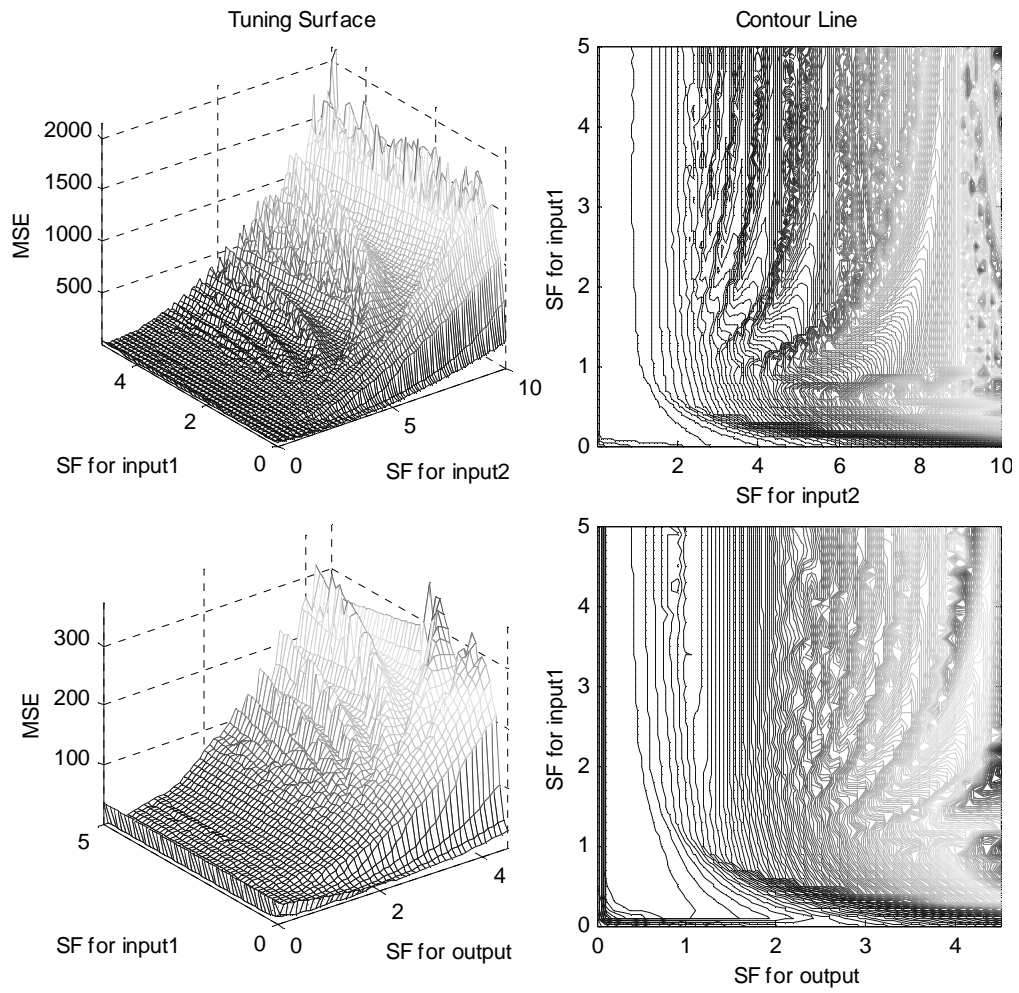
Figure 5.6.3 Tuning Space of SF's

Chapter 6

Genetic Fuzzy Tension Controller

Up to this point, the robustness of the FLC, the applicability of the DFC and the learning ability of the NFC rely on the initial parameter configuration significantly. However, the selection of both the controller and tuning parameters need many trials. As a pre-tuning technique, the versatility of genetic algorithms sheds light on this problem. Provided that simulation condition permits and encoding scheme is efficient, this can be achieved by exploring the application of the natural selection principle in the control area.

In genetic fuzzy controller, the parameters of fuzzy logic are encoded into bit string, *i.e.*, chromosome in biological parlance, viewed as phenotype of the fuzzy inference system and optimized by genetic manipulations. Imitating natural selection mechanics, the evolution of the fuzzy logic controller as a pseudo-organism through the natural selection can be depicted in Figure 6.1 [36 and 37].
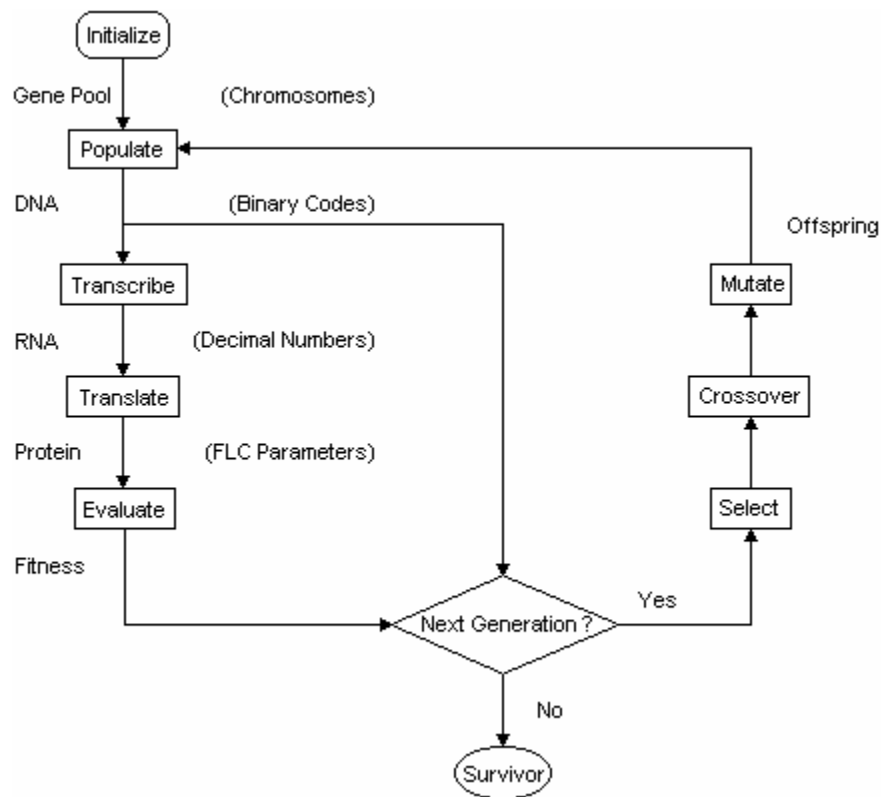
Figure 6.1   Evolution Flow of FLC

6.1 Initialization

The parameters to be searched, which carry out the performance of the FLC, include the scaling factors and the membership function parameters for input1 i_e, input2 Δi_e and output Δv_f, and the rule weights.

In the initialization part, a gene pool is set up in which all the possible candidates of every parameter within the upper and lower bound are generated and the resolution is limited by a pre-set error tolerance [35 and 38]. As the genes, the parameters' positions in the pool are encoded in the form of binary strings. As the DNA segments, these binary strings are concatenated to form chromosomes as demonstrated in Figure 6.1.1.

Phenotype	SF's for Input		MF's for Input1 i_e						MF's for Input2 Δi_e			Rule Weights			MF's for Output Δv_f			SF for Output Δv_f
Gene	$I1_{sf}$	$I2_{sf}$	Z_c	PS_b	PS_a	PS_c	PB_b	PB_a	Z_c	P_b	P_a	$R1_w$	...	$R15_w$	PS	PM	PB	O_{sf}
Locus	1	2	3	4	5	6	7	8	9	10	11	12	...	26	27	28	29	30

Figure 6.1.1   Chromosome

The binary coding will facilitate either deterministic or stochastic operations [38].

The construction of the initial gene pool will increase the overhead of the computer memory. Apart from the number of the FLCs' parameters to be optimized, the scale of the pool and the length of the chromosome are also determined by the number of the candidates of every parameter, which can be controlled by the error tolerance. However, since new individuals can be picked up from the searching space directly instead of coding repeatedly, this table-lookup scheme can not only simplify the creation of new population and, in turn, expedite the searching process, but also guarantee reasonable and viable DNA's, in other words, provide parameter protection for the FLC as depicted in Chapter 4.

In the gene pool, the genotypes of the FLC parameters are encoded and compressed as follows.

6.1.1   Scaling Factor Segments

In locus 1, the binary code $I1_{sf}$ is the gene of the scaling factor for input1 i_e in the range of [0, 5].

In locus 2, the binary code $I2_{sf}$ is the gene of the scaling factor for input2 Δi_e in the range of [0, 10].

In locus 30, the binary code O_{sf} is the gene of scaling factor for output Δv_f in the range of [0, 1].

6.1.2 Rule Weight Segments

From locus 12 to 26, the binary codes $R1_w \sim R15_w$ are the genes of *15* rule weights within the range of [0, 1].

6.1.3 Membership Function Segments

To simplify the structure of the chromosome and accelerate the searching, the fuzzy variables are made even symmetric, that is, symmetric about the ordinate, and thus for the triangle/trapezoid MF's, the parameters of the negative MFs' are not coded. For the zero and positive fuzzy variables, to make sure the final steady-state precision, the parameters with 0 or 1 value are locked. For the purpose of keeping the relevant reasonable positions, the parameters of the membership functions are coded as follows.

(1)　　For the membership function of input1 i_e:

In locus 3, the binary code Z_c is the gene of parameter c_z for fuzzy variable Z in the range of [0, 1] and equals to c_z.

In locus 4, the binary code PS_b is the gene of parameter b_{PS} for fuzzy variable PS in the range of [0, 1] and equals to $1-b_{PS}$.

In locus 5, the binary code PS_a is the gene of parameter a_{PS} for fuzzy variable PS in the range of [0, 1] and equals to $\dfrac{b_{PS} - a_{PS}}{b_{PS}}$.

In locus 6, the binary code PS_c is the gene of parameter c_{PS} for fuzzy variable PS in the range of [0, 1] and equals to $\dfrac{1 - c_{PS}}{1 - b_{PS}}$.

In locus 7, the binary code PB_b is the gene of parameter b_{PB} for fuzzy variable PB in the range of [0, 1] and equals to $1 - b_{PB}$.

In locus 8, the binary code PB_a is the gene of parameter a_{PB} for fuzzy variable PB in the range of [0, 1] and equals to $\dfrac{b_{PB} - a_{PB}}{b_{PB}}$.

(2)   For the membership function of input2 Δi_e :

In locus 9, the binary code Z_c is the gene of parameter c_Z for fuzzy variable Z in the range of [0, 1] and equals to c_Z .

In locus 10, the binary code P_b is the gene of parameter b_P for fuzzy variable P in the range of [0, 1] and equals to $1 - b_P$.

In locus 11, the binary code P_a is the gene of parameter a_P for fuzzy variable P in the range of [0, 1] and equals to $\dfrac{b_P - a_P}{b_P}$.

(3)   For the membership function of output Δv_f :

The gene of fuzzy singleton PS resides in locus 27 as binary code in the range of [0, 1] and equals to $1 - PS$.

The gene of fuzzy singleton PM resides in locus 28 as binary code in the range of [0, 1] and equals to $\dfrac{1-PM}{1-PS}$.

The gene of fuzzy singleton PB resides in locus 29 as the binary code in the range of [0, 1] and equals to $\dfrac{1-PB}{1-PM}$.

6.2 Population

This operation can generate initial individuals, for example, the one shown in Figure 6.2.1, from the gene pool in response to the pre-set population size.
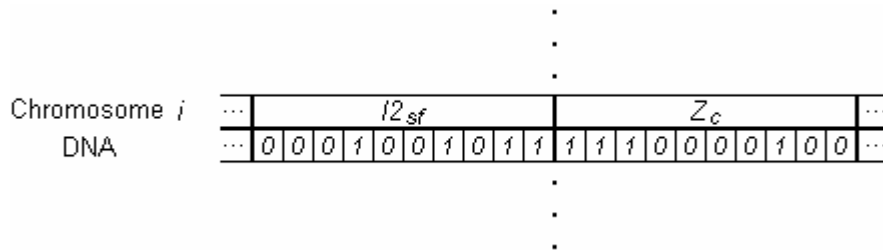
Figure 6.2.1   Population

To accelerate the searching, a group of reasonable parameters, as selected in Chapter 4, is encoded into chromosomes and put into the initial population.

6.3 Transcription

Transcription converts the DNA into the RNA as shown in Figure 6.3.1. In the artificial version, this operation is interpreted as converting the binary gene code into the decimal number which represents the position of the parameter candidate in the gene pool.
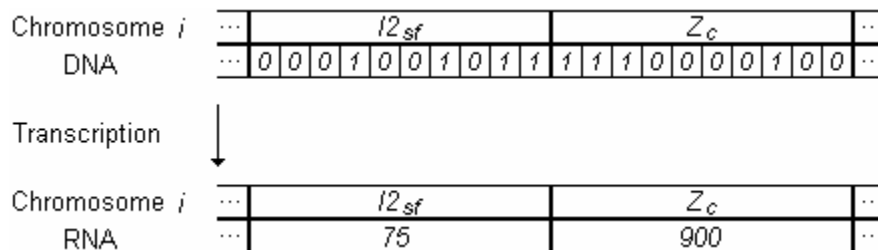
Figure 6.3.1   Transcription

6.4 Translation

The behavioral capacities with reference to the phenotypes of the genes are obtained by the synthesis of the protein from the RNA via translation operation, which includes the inversion of the genotype construction as illustrated in Figure 6.4.1. After translating, the parameters of the FLC are retrieved for ensuing evaluation.
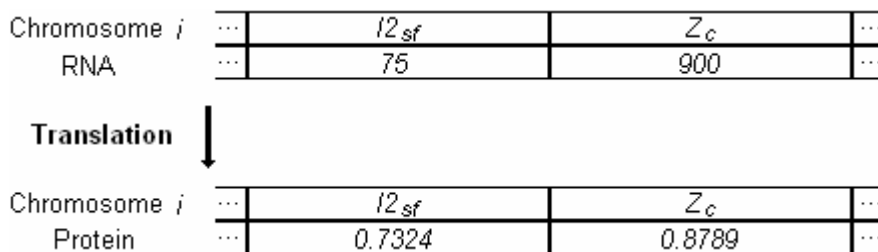
Figure 6.4.1   Translation

6.5 Evaluation

This operation is conducted in environment, which is composed of a tutor and a fitness index. The fitness $F(x_1, x_2)$, indicating the superiority of individual is defined as:

$$F(i_d, i_a) = \frac{1}{\sum\limits_{i=1}^{n} (i_d(i) - i_a(i))^2}$$ (6-5-1)

where:

$n = \dfrac{running\ time}{sampling\ time}$ is the number of sampling intervals.

With the parameters from the last operation, the fuzzy reasoning is carried out in simulation and the fitness is computed by comparing with the exemplar output from the tutor.

If the pre-set generation number is reached, the fittest is singled out as the final survivor, which is esteemed the best individual suited to the environment.

6.6 Generation

The descendants are generated by selection, mutation and crossover operations. The scale of new population maintains the same after these operations [39 and 40].

6.6.1 Selection

Selection method used in this working is Roulette Wheel [38]. The DNA's of individuals with fitness higher than average are copied into new generation for the next operation. The number of replicas is proportional to its fitness.

6.6.2 Crossover

The heritage of characteristic traits is embodied by the interchange of the DNA segments from two chromosomes (Figure 6.6.1). According to the pre-set

crossover number and probability, multiple point transposition occurs in alleles of two adjacent bit strings over the whole population.
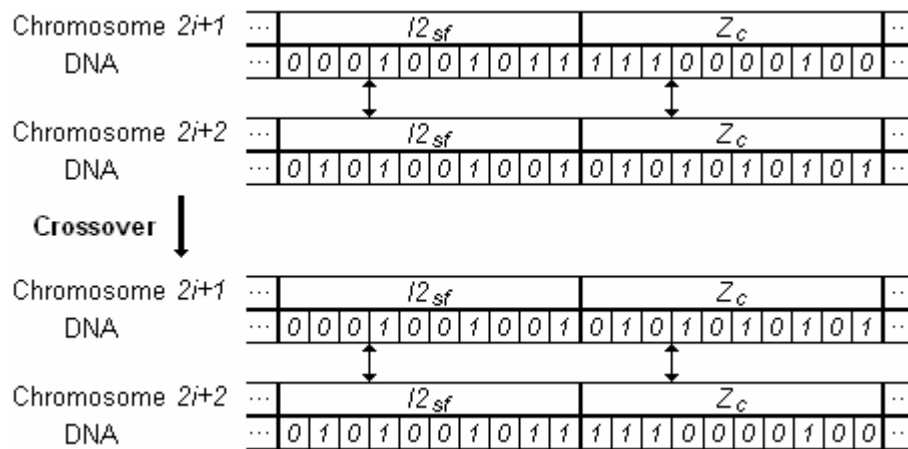
Figure 6.6.1   Crossover

6.6.3 Mutation

The beauty of genetic algorithm is the inherent capability of escaping the local optima. The occurrence of the random genetic mutation is given credit for this salient property because new DNA's are introduced by means of this operation. For binary chromosome, every bit, according to the pre-set mutation probability, randomly changes to its complement (Figure 6.6.2).
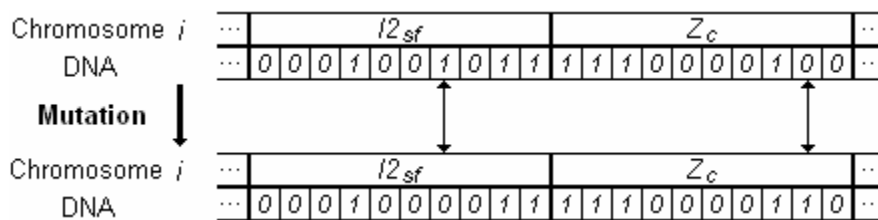
Figure 6.6.2   Mutation

6.7 Simulation Configuration

The searching procedure can be depicted as follows [37]:

```
//Genetic fuzzy controller evolution
Initialize
Generate initial population
for g = 1 to number_of_generation
        for p = 1 to number_of_population
                Transcribe
                Translate
                Execute Simulation
                Compute fitness
        end p
        Select
        Crossover
        Mutate
end g
Winner survives
```

In the following searching, the error tolerance 0.001 is used for all parameters, that is, the bit number of binary code is computed from $\log_2 \dfrac{1}{0.001} \approx 10$. Both the FLC and tutor run for 2.5 seconds using a unit step signal as target current i_t, and thus 50 pairs of [desired current i_d, actual current i_a] data are generated for fitness evaluation for each individual. The initial configuration of the FLC is described in Chapter 4.

Similar to the tuning parameters, the selection of searching parameters is largely a matter of trial-and-error too. A group of parameters with satisfactory searching result is listed in Table 6.7.1. The searching of the optimal rule weights is turned off and the genes of the rule weights are treated as recessive because their effect is vague.

Settling Time(s)	Running Time (s)	Generation	Population	Mutation Probability	Crossover Number	Crossover Probability
0.3	2.5	75	50	0.02	10	0.01

Table 6.7.1   Searching Parameters

6.8 Tuning Result

To see the randomness of genetic operations, two results with the same searching parameters are presented.

As one outcome, the resulting optimal FLC parameters are listed in Table 6.8.1; the response of current after searching is plotted in Figure 6.8.1. It can be seen that the response of the tension control system is improved dramatically compared with the one designed in Chapter 4.

Figure 6.8.3 and Figure 6.8.4 give another result. Although the distribution of the MF's are quite different, the response of the tension control system is satisfactory and, comparing with last result, is much smooth but with big overshoot.

Figure 6.8.2 and Figure 6.8.4 render the visual representation of the evolution process. These two plots verify that the genetic algorithm can avoid that the searching is trapped in local optima.

In each case, the actual current i_a coincides with desired one well with final mean square error *0.0004*. Another observation is that increasing the generation number is commensurate in searching performance with increasing population in each generation, but the former scheme is orders of magnitude faster in searching than the latter.

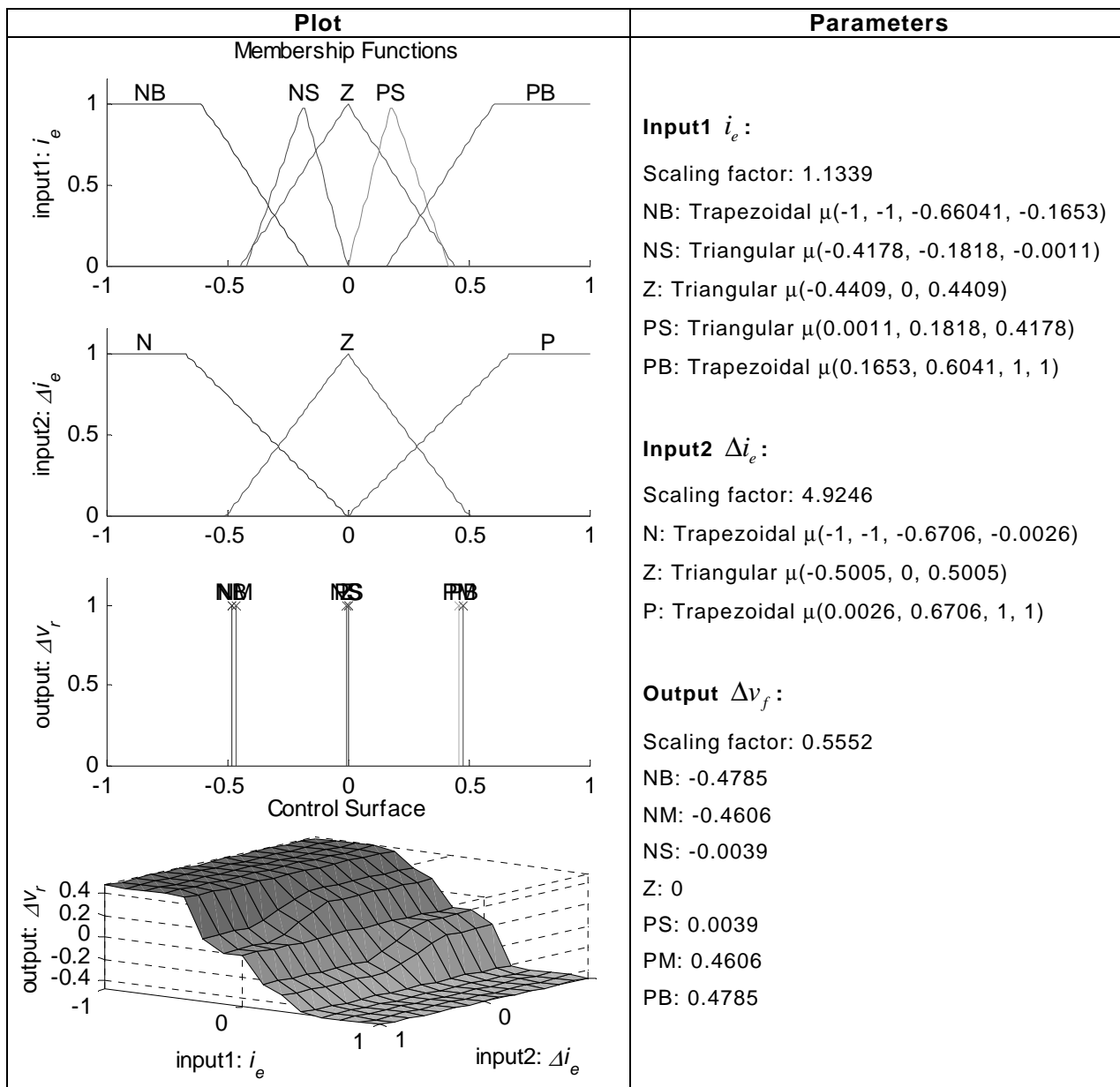| Plot	Parameters
Membership Functions *(graphs)*	**Input1** i_e **:** Scaling factor: 1.1339 NB: Trapezoidal μ(-1, -1, -0.66041, -0.1653) NS: Triangular μ(-0.4178, -0.1818, -0.0011) Z: Triangular μ(-0.4409, 0, 0.4409) PS: Triangular μ(0.0011, 0.1818, 0.4178) PB: Trapezoidal μ(0.1653, 0.6041, 1, 1) **Input2** Δi_e **:** Scaling factor: 4.9246 N: Trapezoidal μ(-1, -1, -0.6706, -0.0026) Z: Triangular μ(-0.5005, 0, 0.5005) P: Trapezoidal μ(0.0026, 0.6706, 1, 1) **Output** Δv_f **:** Scaling factor: 0.5552 NB: -0.4785 NM: -0.4606 NS: -0.0039 Z: 0 PS: 0.0039 PM: 0.4606 PB: 0.4785

Table 6.8.1   Parameters of MF's after Searching (Result 1)
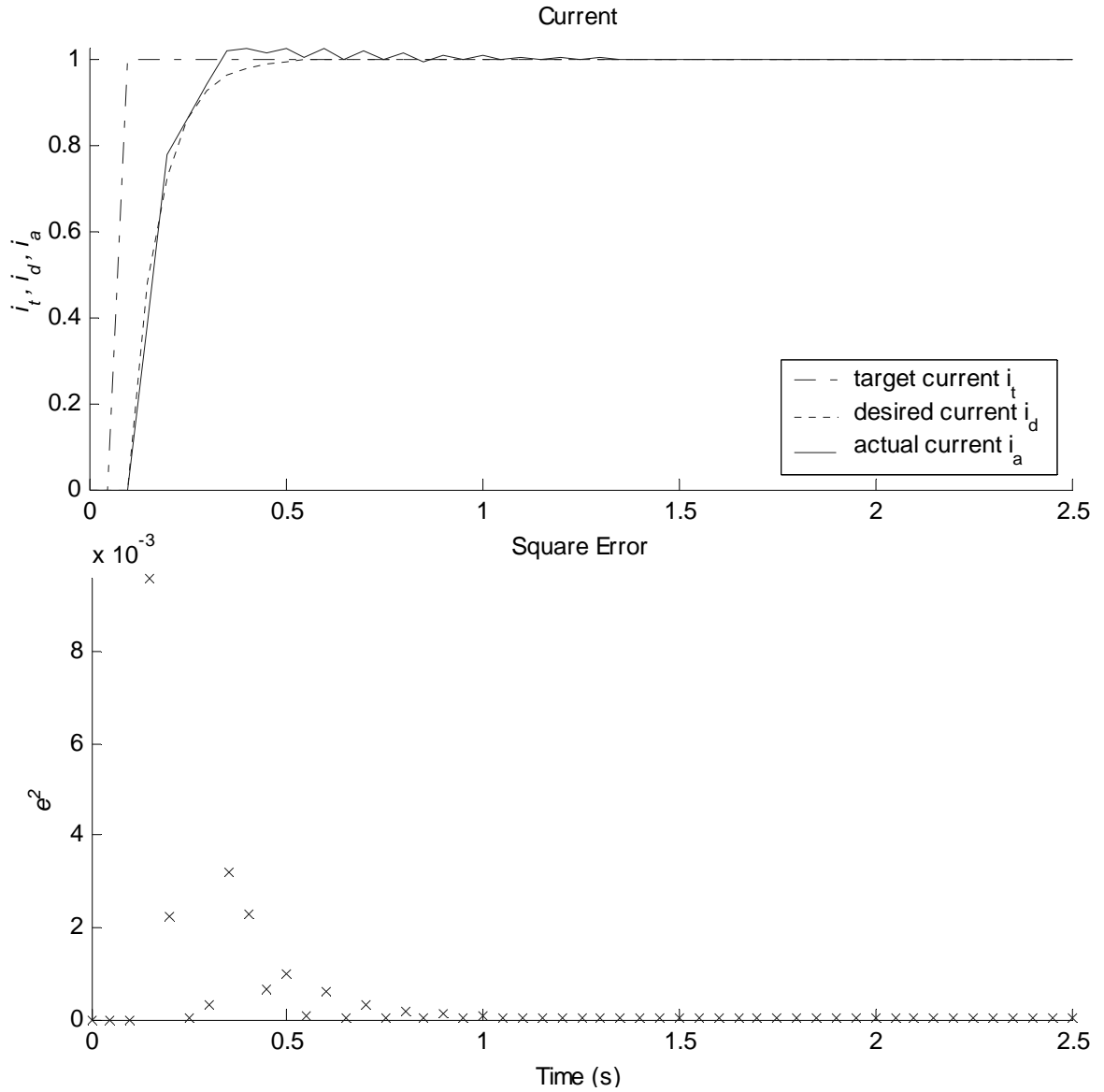
Figure 6.8.1   Response after Searching (Result 1)
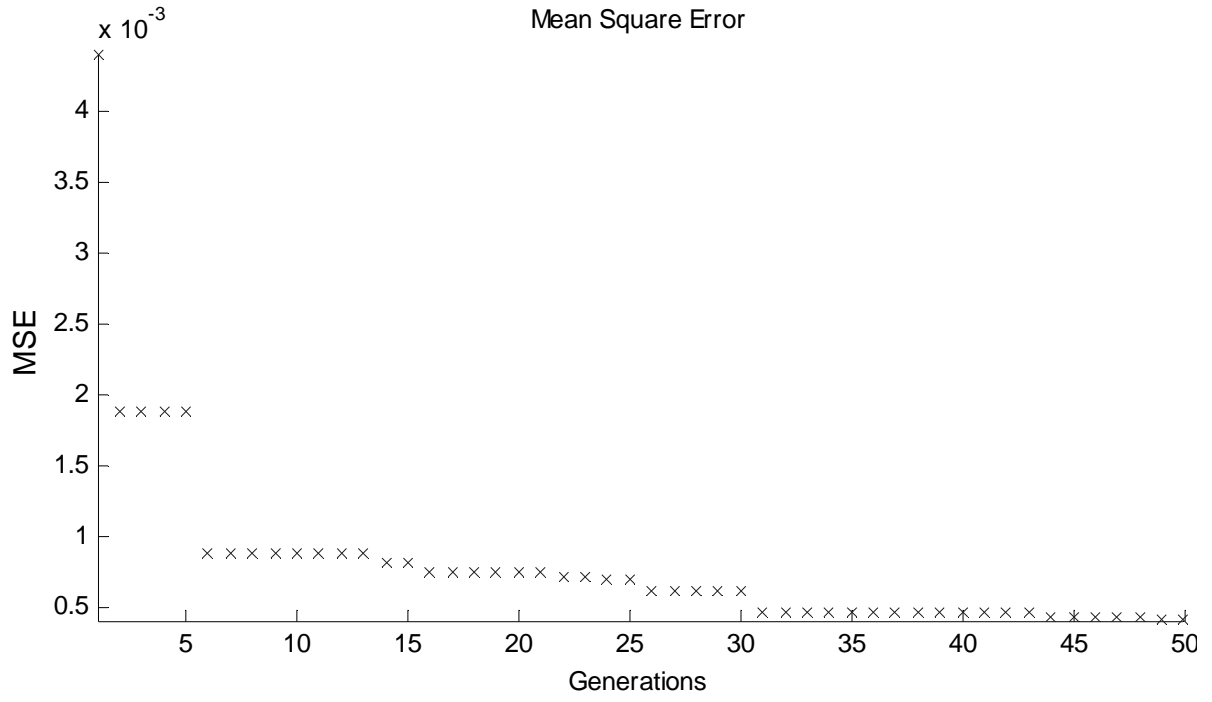
Figure 6.8.2   Mean Square Error of Searching (Result 1)

Plot	Parameters
Membership Functions input1: i_e — NB NS Z PS PB input2: Δi_e — N Z P output: Δv_r — NB NM NS Z PM PB **Control Surface** output: Δv_r vs input1: i_e and input2: Δi_e	**Input1** i_e **:** Scaling factor: 1.3099 NB: Trapezoidal $\mu(-1, -1, -0.4937, -0.2751)$ NS: Triangular $\mu(-0.7982, -0.2092, -0.0012)$ Z: Triangular $\mu(-0.8280, 0, 0.8280)$ PS: Triangular $\mu(0.0012, 0.2092, 0.7982)$ PB: Trapezoidal $\mu(0.2751, 0.4937, 1, 1)$ **Input2** Δi_e **:** Scaling factor: 7.7908 N: Trapezoidal $\mu(-1, -1, -0.8201, -0.6510)$ Z: Triangular $\mu(-0.7722, 0, 0.7722)$ P: Trapezoidal $\mu(0.6510, 0.8201, 1, 1)$ **Output** Δv_f **:** Scaling factor: 0.3480 NB: -0.7704 NM: -0.5707 NS: -0.5015 Z: 0 PS: 0.5015 PM: 0.5707 PB: 0.7704

Table 6.8.2   Parameters of MF's after Searching (Result 2)

Current

Square Error

Figure 6.8.3   Response after Searching (Result 2)

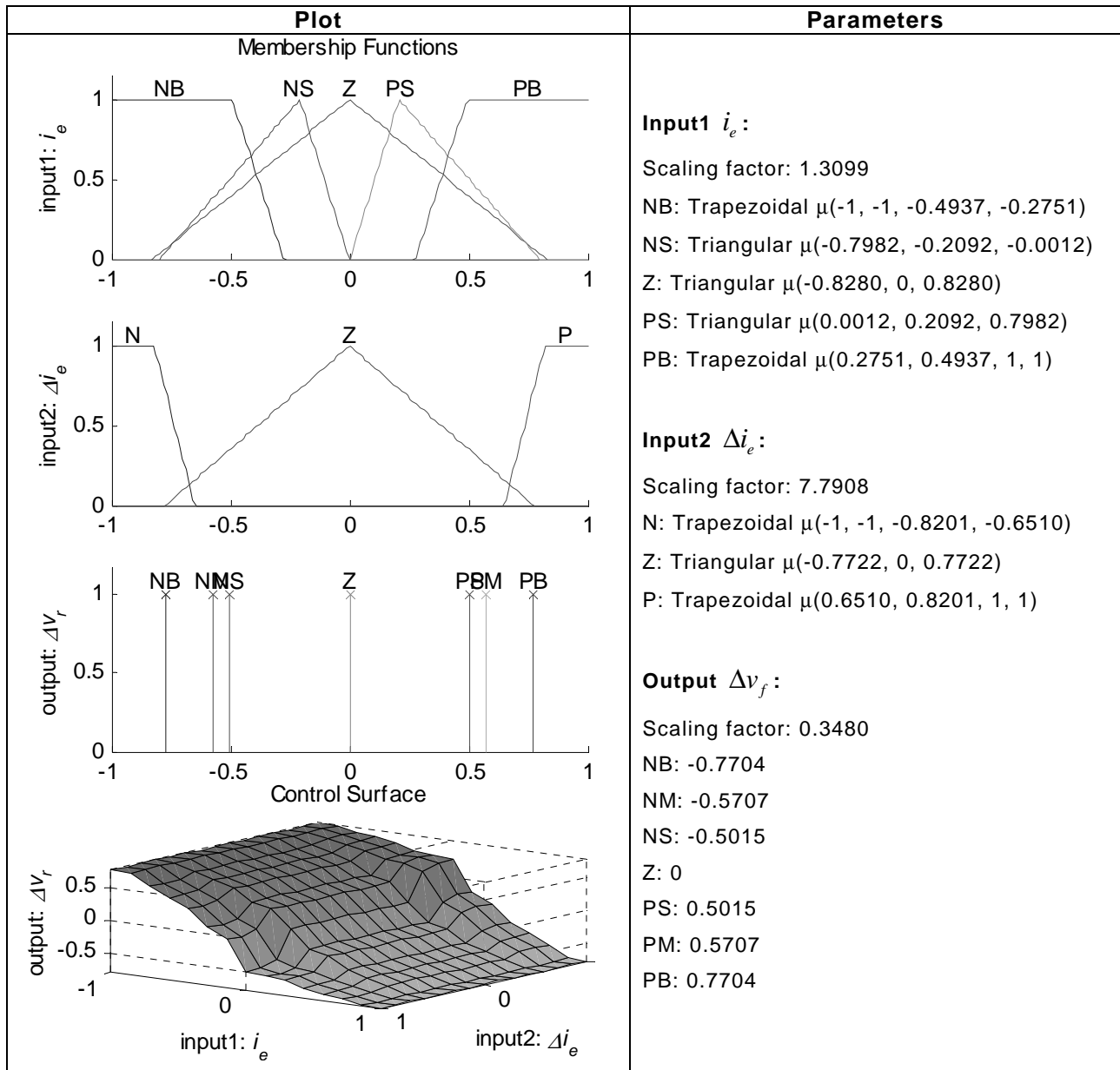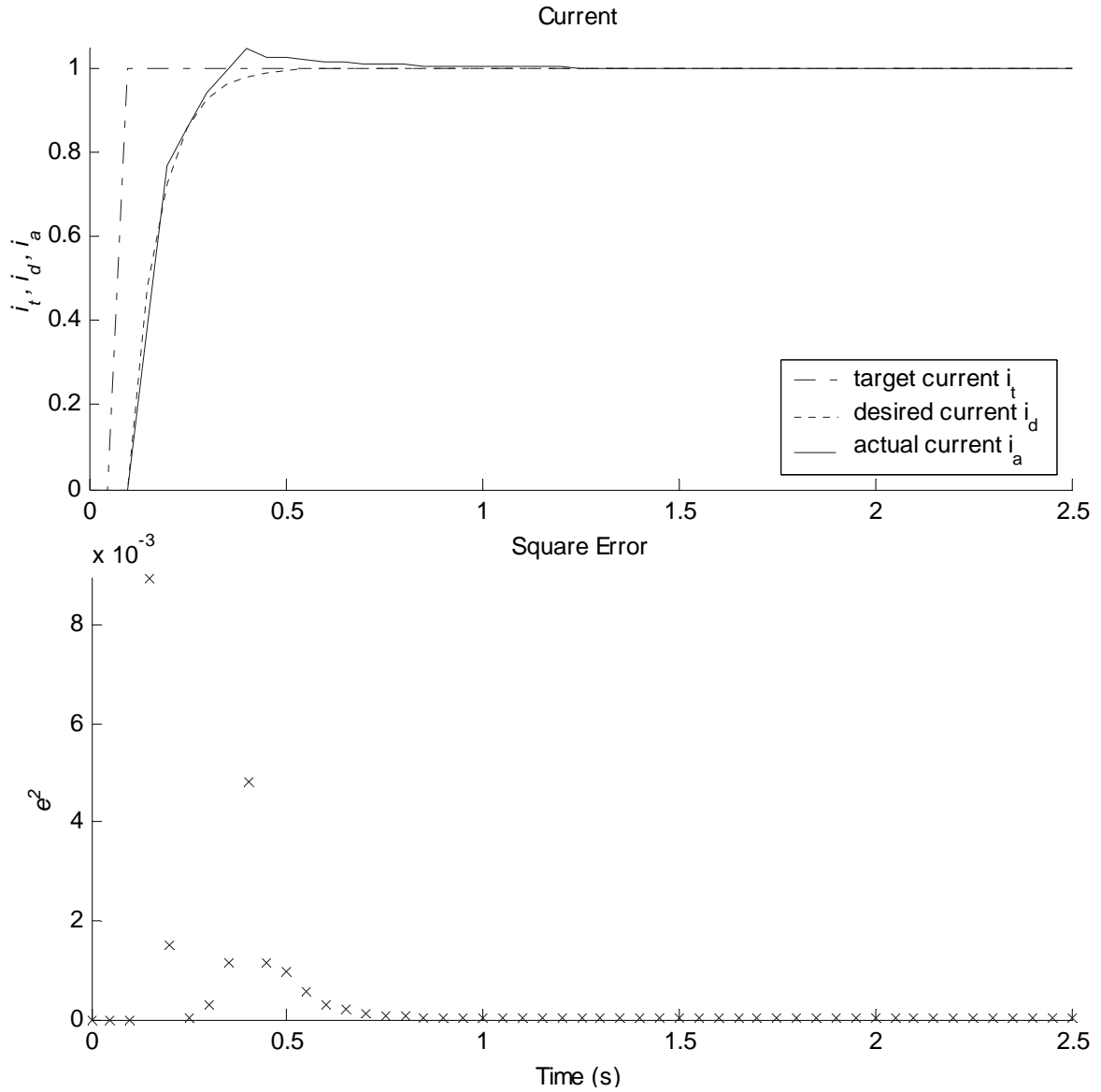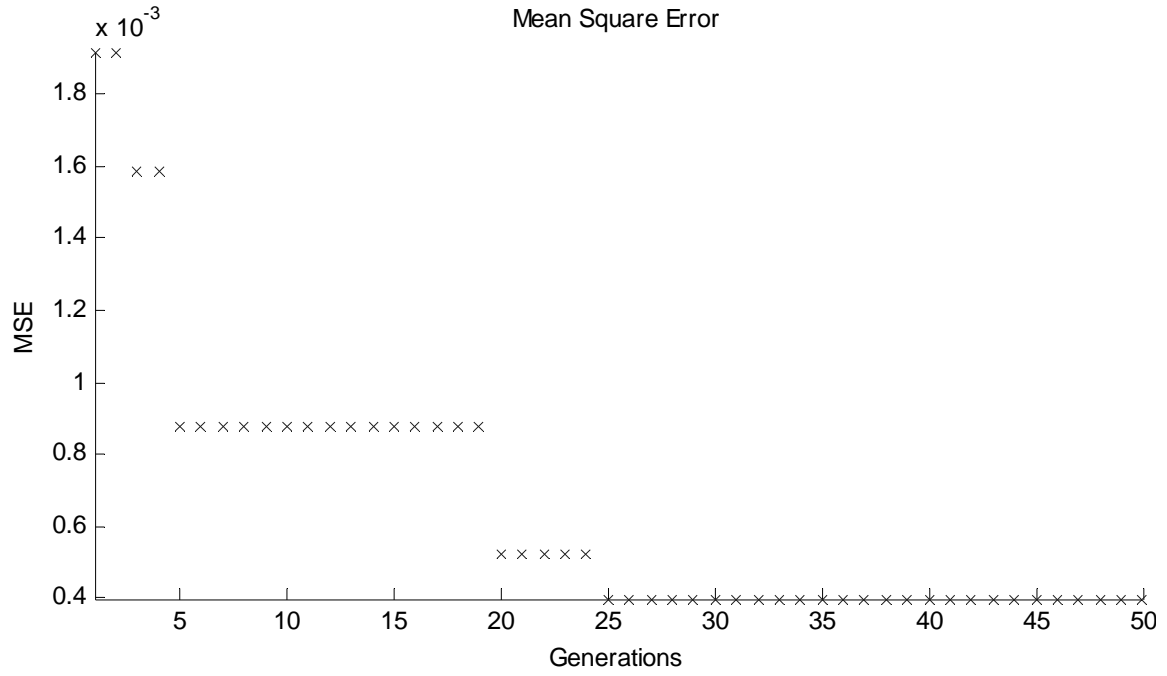Figure 6.8.4   Mean Square Error of Searching (Result 2)

Chapter 7

Conclusions

7.1 Summary

In this work, the application of intelligent techniques including the fuzzy reasoning, neural net and natural selection for the tension control in the tandem rolling mill was investigated. Preliminary simulations on a single rolling mill stand were provided to evaluate the performances of the above techniques. By means of a generalized least-squares method, the rolling mill stand together with the speed and gage control systems is identified as a linear discrete time invariant system. All these intelligent techniques, by resorting to emulating the biological mechanism, deal with incomplete plant knowledge, ambiguous operation information and uncertain environments.

The FLC's criteria, such as the overshoot, response speed and steady-state, accuracy can be satisfied through the incorporation of ample expertise and appropriate configuration of reasoning elements. In this work, engineering experience on the fuzzy partition and rule base are borrowed from looper control; the influence of such control elements as the parameters of the MF's, SF's, inference type, operations and defuzzification method are investigated. In general, the FLC performs well in rolling processes with expected humanoid reasoning ability. However, the design is essentially based on iterative trial-and-error and interrelated with the controlled plants.

The principle underlying the DFC is straightforward, that is, improving the FLC in terms of the specification requirement of tension response. According to the assertions pertain to the influences of the control elements on the unit step response, these elements are dynamically switched during control activity. The application of the DFC needs the provision of different definition of anticipated response performance indices for different target signals. On the other hand, the improvement relies on the characteristics of the rolling processes and is susceptible to noise.

On the platform of the fuzzy inference system and orchestrated by tutor, the simulation results of the NFC and GFC signify the successful applications of expected intelligence corresponding to the biological functions, neural net and natural selection respectively.

The selections of learning parameters for both the backward propagation and genetic algorithms are indispensable and are a trade-off between learning speed and performance. In contrast, the selection of the tuning parameters in the NFC is more subtle and strenuous in relation to convergence than that of the searching parameters in the GFC.

Efficient tuning of the NFC takes a combination of initial FLC configuration and selection of the learning parameters. As a serious disadvantage of tuning, the obtained fuzzy parameters can only be the best ones around the nominal values. In stark comtrast, the parameters acquired from the GFC are optimal over the whole searching space and irrelevant to initial FLC setting.

Compared with the GFC, a major disadvantage of the NFC is the derivative computation and continuity of the control space that the backward propagation algorithm calls for. These two stipulations narrow the options of the fuzzy control elements: the inference system is confined to the Sugeno-type and the aggregation operation of antecedents is continuous function: product.

Notwithstanding limited options for the NFC, the real-time online tuning of the NFC is an appealing property, which favors the time-varying rolling processes, as long as the overhead of the computation time is trivial. By contrast, the genetic algorithm needs the emulation of a variety of individuals and can only be a means to configure the FLC before being put into practice.

Apart from biological mechanism, another aspect that substantially differentiates the NFC from the GFC is that the tuning of the backward propagation algorithm is deterministic, while the searching of the genetic algorithm is stochastic and less application-specific.

The supervision for the NFC and the exemplar for the GFC from the tutor mark the boundaries of effective training or evolving. One of the most beneficial consequences of introducing a tutor into either the on/off line tuning or searching is that all the specifications of the tension control performance are taken into account. Going still further, the guidance can adapt to different reference signal automatically.

Among the DFC, NFC and GFC, both the initial response speed and final steady-state precision can be improved by these learning schemes easily. At this stage, it can be seen that the NFC guarantees the smoothness of the fuzzy control action and the DFC is the worst. A promising comprehensive self-organization scheme is constructing a GFC and searching the fuzzy parameters in whole control space with respect to desired criteria. With the resulting global optimal parameters, construct a NFC and putting it into practice with the ability to approximate the best parameters online.

7.2 Future Work

The rolling are complex non-linear processes with harsh noise. In this study, the simulations are conducted in a simplified operation situation and the model is also linearized. These results have to be justified by experiments. Furthermore, the FLC along with its intelligence variations need testing in experiments and industrial practice. An ongoing real time simulation plan is schematically pictured in Figure 7.1. The model is constructed in Labview® and tension control scheme is implemented by IEC1131 programming software. These components together with other auxiliary systems are assembled via Windows NT® workstation in conjunction with Modbus® protocol.
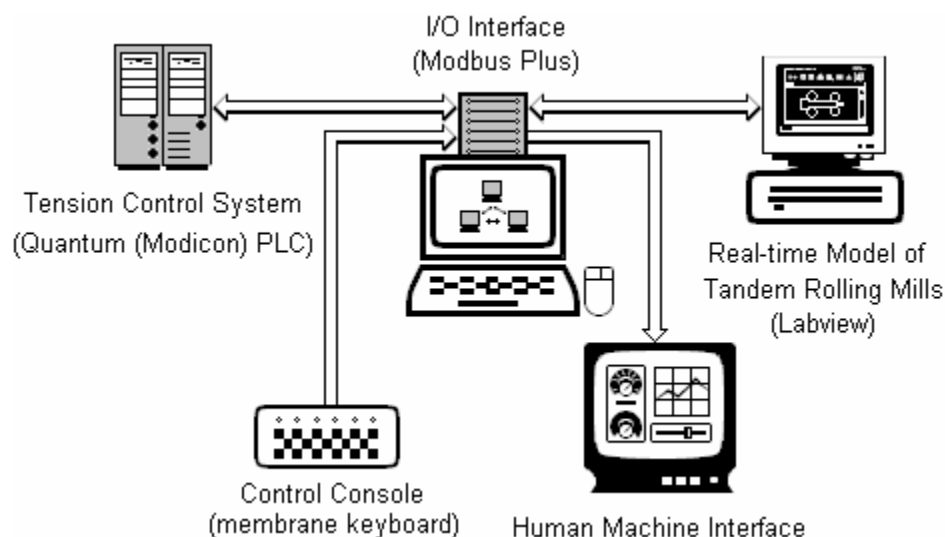
Figure 7.1   Real Time Simulation System

In the DFC, if multiple groups of MF's with different fuzzification configurations, such as partition, WO and corresponding rule base, are provided and dynamically switched during control activity, dramatic improvement to the FLC is also anticipated. As another potential utilization, the seamless conversion of controllers for consecutive normal and run-out modes can be achieved based on the concept of the DFC.

An unsolved crux is the perplexing property of the rule weights. Although the tuning and searching of the rule weights are considered in either the NFC or GFC, their self-learning is turned off because the results are not satisfied as long as they deviate from the nominal value 1's.

Some issues are not discussed in this project. One is the influence of the fuzzy partition and the corresponding rule base. Detailed behavior analysis of human operators is necessary for the selection of the MF's and the construction of the fuzzy rule base.

For the neuro-fuzzy controller, learning parameters, such as learning rates, epochs and the error tolerance, influence the NFC's precision, convergence speed and stability of the FLC dramatically. For the genetic fuzzy controller, the parameters of generation, population, mutation and crossover also determine the convergence. These parameters are selected by repeated trials here. Whereas the genetic algorithm can be used in a variety of applications, these issues may lead to a possible development direction of intelligence application within the context of the fuzzy reasoning. The self-organization of the fuzzy partition and rule base, as phenotypes, can be carried out by encoding these elements into chromosomes and optimizing via the genetic algorithm. With the synthesization of the backward propagation and genetic algorithms in the neural network, the structure of the neural network can be flexible; such elements as the learning rates, node number in antecedent, rule and consequent layers, which are determined by the fuzzy partition and rule base, can be optimized by evolutionary searching, and thus the backward propagation algorithm can take over fine-tuning.

From the GFC's two searching results, it can be seen that the two resulting FLC's with the same fitness can have completely different MF's patterns. This gives the space to apply multiple-fitness to the evaluation and evolution. For example, using the smoothness of the response as another fitness will increase

the survival possibility of the individual with less stair-stepping effects, specifically, result 2.

In the current genetic algorithm, the stabilizing selection was used, that is, the fuzzy inference system evolves in a constant environment. Considering the volatile characteristics of the rolling condition, evolution in inconstant environments should be taken into account in future applications.

As for the tutor, both the transient and steady-state response indices are taken into consideration evenly. In practical application, the tutoring signal can be weighted to emphasize a certain performance specification of the system response.

Bibliography

[1] V. B. Ginzburg, "Steel-rolling Technology: Theory and Practice," New York, M. Dekker, 1989.

[2] QUAD Engineering, "An intelligent tension control system for rolling mills," *Project #934.2*, 1998.

[3] V. B. Ginzburg, "High-Quality Steel rolling, Theory and practice," New York, Marcel Dekker, 1993.

[4] Nippon Kokan K. K, "New tension control method for hot strip mills," *Transactions of the Iron and Steel Institute of Japan*, vol. 26, no. 3, pp. 256-256, 1986.

[5] G. V. Bass and R. Hartmann, "Minimum Tension Control in Finishing Train of Hot Strip Mills," *Iron and Steel Engineer*, pp. 48-53, Nov. 1987.

[6] Y. Noguchi, K. Baba, H. Ogai, H. Ishii, T. Oka and M. Baba, "Multivariable control for bar rolling and precision bar rolling system," *Proceedings of the 1995 IEEE IECON 21st International Conference on Industrial Electronics, Control, and Instrumentation*, vol. 2, pp. 780-785, 1995.

[7] S. Duysters, J. A. J. Goversvan and A. J. J. D. Weiden, "Process interactions in a hot strip mill; possibilities for multivariable control?" *Proceedings of the Third IEEE Conference on Control Applications*, vol.3, pp. 1545-1550, 1994.

[8] A. J. Pollmann, "Control strategies for rolling mills," *Industry Applications Society Annual Meeting, Conference Record of the 1993 IEEE*, vol.3, pp. 2420-2425, 1993.

[9] Y. Kadoya, T. Ooi, Y. Washikita and Y. Seki, "Strip gage and tension control at cold tandem mill based on ILQ design theory," *Proceedings of*

the 1999 IEEE International Conference on Control Applications, vol. 1, pp. 23-28, 1999.

[10]   S. S. Garimella and K. C. Srinivasan, "Application of iterative learning control to coil-to-coil control in rolling," *IEEE Transactions on Control Systems Technology*, vol. 6, no. 2, pp. 281-293, Mar. 1998.

[11]   I. Hoshino, Y. Okamura and H. Kimura, "Observer-based multivariable tension control of aluminum hot rolling mills," *Proceedings of the 35th IEEE Conference on Decision and Control*, vol.2, pp. 1217-1222, 1996.

[12]   Y. Seki, K. Sekiguchi, Y. Anbe, K. Fukushima, Y. T. and S. Ueno, "Optimal multivariable looper control for hot strip finishing mill," *IEEE Transactions on Industry Applications*, vol. 27, no. 1, part 1, pp. 124-130, Jan.-Feb. 1991.

[13]   H. Miura, S. Nakagawa, S. Fukushima and J. Amasaki, "Gauge and tension control system for hot strip finishing mill," *Proceedings of the IECON '93, International Conference on Industrial Electronics, Control, and Instrumentation*, vol.1, pp. 463-468, 1993.

[14]   I. Hiroyuki, M. Youichi and S. Kunio, "Looper H-infinity control for hot-strip mills," *IEEE Transactions on Industry Applications*, vol. 33, pp. 790-796, May/Jun. 1997.

[15]   M. Takano, K. Takeda, K. Kurotani and S. Hosaka, "Loop control system of wire rolling mill using H-infinity/control design method," *Proceedings of the 1992 International Conference on Industrial Electronics, Control, Instrumentation, and Automation, 1992. Power Electronics and Motion Control*, vol.3, pp.1141-1146, 1992.

[16]   M. Shioya, N. Yoshitani and T. Ueyanma, **"**Noninteracting control with disturbance compensation and its application to tension-looper control for hot strip mill," *Proceedings of the 1995 IEEE IECON 21st International Conference on Industrial Electronics, Control, and Instrumentation*, vol. 1, pp. 229-234, 1995.

[17]   T. Hesketh, Y. Jiang, D. J. Clements, David H. Butler and R. V. D. Laan, "Controller design for hot strip finishing mills," *IEEE Transactions on Control Systems Technology*, vol. 6, no. 2, pp. 208-219, Mar. 1998.

[18]   N. Hur and K. Nam, "A robust load-sharing control scheme for parallel-connected multisystems," *IEEE Transactions on Industrial Electronics*, vol. 47, no. 4, pp. 871-879, Aug. 2000.

[19]   S. H. Jeon, J. Kim, K. Jung, S. Sul and J. Y. Choi, "Decoupling control of bridle rolls for steel mill drive system," *Industry Applications Conference, Thirty-second IAS Annual Meeting, IAS '97, Conference Record of the 1997 IEEE*, vol. 3, pp. 2144-2150, 1997.

[20]   H. Ogai, A. Fujii, K. Baba, S. Kakimoto and T. Harakawa, "Multidimensional size control in rod bar rolling and cold strip rolling by using fuzzy method," *American Control Conference, Proceedings of the 1998*, vol. 6, pp. 3822-3823, 1998.

[21]   C. T. Lin and C. S. George Lee, "Neural Fuzzy Systems, A Neuro-Fuzzy Synergism to Intelligent Systems," Upper Saddle River: Prentice Hall, Inc., 1996.

[22]   F. Janabi-Sharifi and J. Fan, "Self-tuning fuzzy looper control for rolling mills," *Proceedings of the 39th IEEE Conference on Decision and Control, 2000*, vol. 1, pp. 376-381, 2000.

[23]   L. Ljung, "System Identification Toolbox," The MathWorks Inc., 1999.

[24]   L. Ljung, "System Identification: Theory for The User," Upper Saddle River, NJ Prentice Hall PTR, 1999.

[25]   J. P. Norton, "An Introduction to Identification," *Academic Press*, 1986.

[26]   L. Reznik, "Fuzzy controller design: recommendations to the user," *Second Internation Conference on Knowledge-Based Intelligent Electronic Systems, Adelaide, Australia*, pp. 609-616, 1998.

[27]   L. Reznik, "Fuzzy Controller," Oxford: Butterworth-Heinemann Linacre House, 1997.

[28] J. V. D. Oliveira, "Design methodology for fuzzy system interfaces," *IEEE Transactions on Fuzzy Systems*, vol. 3, no. 4, pp. 404-414, Nov. 1995.

[29] D. Driankov, H. Hellendoorn and M. Reinfrank, "An Introduction to Fuzzy Control," New York: Springer-Verlag Berlin Heidelberg, 1996.

[30] E. Gur and D. Mendlovic, "Optical fuzzy controllers," *Proceedings IEEE Convention of Electrical and Electronics Engineers in Israel, IEEE,* Piscataway, NJ, USA, 96TH8190, pp. 375-378, 1996.

[31] R. Jager, H. B. Verbruggen and P. M. Bruijn, "The role of defuzzification methods in the application of fuzzy control," *IFAC Intelligent Components and Instruments for Control Applications, Malaga, Spain,* pp. 75-80, 1992.

[32] C. L. Phillips, "Feedback Control Systems," Cliffs, N.J.: Prentice Hall, 1996.

[33] M. T. Hagan, H. B. Demuth and Mark Beale, "Neural Network Design," Boston: PWS Pub., 1996.

[34] J. W. Hines and L. H. Tsoukalas, "MATLAB Supplement to Fuzzy And Neural Approaches in Engineering," New York: Wiley, 1997.

[35] D. A. White and D. A. Sofge, "Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches," New York: Van Nostrand Reinhold, 1992.

[36] W. Banzhaf, P. Nordin, R. E. Keller and F. D. Francone, "Genetic Programming: An Introduction on The Automatic Evolution of Computer Programs And Its Applications," San Francisco, Calif.: Morgan Kaufmann Publishers; Heidelburg: Dpunkt-verlag, 1998.

[37] K. F. Man, K. S. Tang and S. Kwong, "Genetic Algorithms," Springer-Verlag London Limited, 1999.

[38] L. Chambers, "Practical Handbook of Genetic Algorithms," Chapman & Hall/CRC, 2001.

[39] Z. Michalewicz, "Genetic Algorithm+Data Structures = Evolution Programs, 2nd," Springer-Verlag New York Limited, 1994.

[40] D. E. Goldberg, "Genetic Algorithms in Search, Optimazation, and Machine Learning," Addison-Wesley, MA, 1989.

[41] H. B. Schwefel and T. Back, "Evolutionary computation: an overview," *Proceedings of IEEE International Conference on Evolutionary Computation*, pp.20-29, 1996.

[42] T. Back, U. Hammel and H. Schwefel, "Evolutionary computation: comments on the history and current state," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 3-17, Apr. 1997.

[43] S. L. Hung and H. Adeli, "A parallel genetic/neural network learning algorithm for MIMD shared memory machines," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 900-909, Nov. 1994.

[44] C. L. Karr and E. J. Gentry, "Fuzzy control of pH using genetic algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 1, pp. 46-53, Feb. 1993.

[45] D. Delahaye, J. M. Alliot, M. Schoenoauer and J. L. Farges, "Genetic algorithms for partitioning air space," *Proceedings of the Conference on Artificial Intelligence Application*s, pp. 291-297, 1994

[46] M. Juric, "Optimizing genetic algorithm parameters for multiple fault diagnosis applications," *IEEE Proceedings of the 10th International Conference on AI for Application*s, pp. 434-440, 1994.

[47] Y. Jessen, M. A. Keane, and J. R. Koza, "Automatic design of both topology and tuning of a common parameterized controller for two families of plants using genetic programming," *IEEE International Symposium on Computer-Aided Control System Design*, pp. 234-242, 2000.