Learning of Type-2 Fuzzy Logic Systems using Simulated Annealing

by

Majid Almaraashi

A thesis submitted in partial fulfilment for the
degree of Doctor of Philosophy in Artificial Intelligence

DE MONTFORT UNIVERSITY

September 2012

Declaration of Authorship

The work contained within this thesis is purely that of the author unless otherwise stated. This thesis has not been submitted as a whole for any examination, qualification or publication.

Majid Almaraashi

"Science is wonderfully equipped to answer the question "How?" but it gets terribly confused when you ask the question "Why?"

Erwin Chargaff

DE MONTFORT UNIVERSITY

Abstract

Faculty of Technology
Department of Informatics

by **Majid Almaraashi**

This thesis reports the work of using simulated annealing to design more efficient fuzzy logic systems to model problems with associated uncertainties. Simulated annealing is used within this work as a method for learning the best configurations of type-1 and type-2 fuzzy logic systems to maximise their modelling ability. Therefore, it presents the combination of simulated annealing with three models, type-1 fuzzy logic systems, interval type-2 fuzzy logic systems and general type-2 fuzzy logic systems to model four bench-mark problems including real-world problems. These problems are: noise-free Mackey-Glass time series forecasting, noisy Mackey-Glass time series forecasting and two real world problems which are: the estimation of the low voltage electrical line length in rural towns and the estimation of the medium voltage electrical line maintenance cost. The type-1 and type-2 fuzzy logic systems models are compared in their abilities to model uncertainties associated with these problems. Also, issues related to this combination between simulated annealing and fuzzy logic systems including type-2 fuzzy logic systems are discussed.

The thesis contributes to knowledge by presenting novel contributions. The first is a novel approach to design interval type-2 fuzzy logic systems using the simulated annealing algorithm. Another novelty is related to the first automatic design of general type-2 fuzzy logic system using the vertical slice representation and a novel method to overcome some parametrisation difficulties when learning general type-2 fuzzy logic systems. The work shows that interval type-2 fuzzy logic systems added more abilities to modelling information and handling uncertainties than type-1 fuzzy logic systems but with a cost of more computations and time. For general type-2 fuzzy logic systems, the clear conclusion that learning the third dimension can add more abilities to modelling is an important advance in type-2 fuzzy logic systems research and should open the

doors for more promising research and practical works on using general type-2 fuzzy logic systems to modelling applications despite the more computations associated with it.

Acknowledgements

In the name of Allah, the Most Gracious and the Most Merciful. All praises to Allah for the strengths and his blessing in completing this thesis.

It would not have been possible to write this thesis without the help and support of the kind people around me. It is to them that I owe my deepest gratitude.

- It gives me great pleasure in acknowledging the support and help of my first supervisor Professor Robert John. His wisdom, knowledge and commitment to the highest standards inspired and motivated me during these years. Special thanks to the members of my supervision team; Dr. Samad Ahmadi, Professor Adrian Hopgood and my advisor Dr. Simon Coupland. My supervisors' encouragement, guidance and support from the initial to the final level enabled me to develop valuable research and scientific skills.

- I wish to express my love and gratitude to my beloved families; my father Saeed, mother Fatima, brothers and sisters for their understanding and endless love through the duration of my studies.

- I would like to show my gratitude to my wife, Safiah, without whom this effort would have been worth nothing. For her love, support and patience since we came together to UK to get my postgraduate degrees. For looking after our little children; Osama, Logain and Aljoori.

- Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the thesis.

Contents

List of Figures

List of Tables

Abbreviations

FLS	Fuzzy Logic System
T1FLS	Type-1 Fuzzy Logic System
T2FLS	Type-2 Fuzzy Logic System
IT2FLS	Interval Type-2 Fuzzy Logic System
GT2FLS	General Type-2 Fuzzy Logic System
FOU	Footprint of Uncertainty
SMF	Secondary Membership Function
VSCTR	Vertical-Slices Centroid Type-reduction
SA	Simulated Annealing
SA-T1FLS	Simulated Annealing with Type-1 Fuzzy Logic System
SA-T2FLS	Simulated Annealing with Type-2 Fuzzy Logic System
SA-IT2FLS	Simulated Annealing with Interval Type-2 Fuzzy Logic System
SA-GT2FLS	Simulated Annealing with General Type-2 Fuzzy Logic System
GT2FLS-Sampling	General Type-2 Fuzzy Logic System with Sampling Type-reduction
GT2FLS-VSCTR	General Type-2 Fuzzy Logic System with Vertical-Slices Centroid Type-reduction

Dedicated To My Family...

Chapter 1

Introduction

1.1 Overview

This thesis presents a novel combination of fuzzy logic systems (mainly type-2 fuzzy logic systems) with the simulated annealing algorithm to design high performing systems to model uncertainties. The combination is applied in cases where the exact systems are difficult to acquire by experts due to the lack of knowledge and the existence of uncertainties associated with these problems. Simulated annealing is used within this work as a method for learning the best configurations of fuzzy logic systems to maximise their modelling ability. For this purpose, it presents the combination of simulated annealing with three models, type-1 fuzzy logic systems, interval type-2 fuzzy logic systems and general type-2 fuzzy logic systems to model four bench-mark problems including real-world problems.

1.2 Context

The main methods used in this work are concerned with type-1 fuzzy logic systems, type-2 fuzzy logic systems and simulated annealing. To introduce the context of this thesis, the three methods are highlighted here followed by the research motivation.

1.2.1 Fuzzy logic systems

Fuzzy logic systems are rule-based systems that use the theory of fuzzy sets and fuzzy logic proposed by Zadeh in (Zadeh, 1965). It is one of the most important areas of fuzzy set theory. Nowadays, many applications are using fuzzy logic systems to represent knowledge in a closer way to how human are thinking. In a fuzzy set, any element in the set is given a degree of membership of this set as opposed to the ordinary crisp set where its membership is characterised by two values only (0 or 1). The most used form of fuzzy logic systems is based on using type-1 fuzzy sets and so that it is called type-1 fuzzy logic system. In general, a fuzzy logic system is a process involves fuzzifying input crisp values followed by inference engine to apply fuzzy rules and ends by defuzzifying the results into crisp values as outputs.

1.2.2 Type-2 fuzzy logic systems

Type-2 fuzzy logic systems are rule-based fuzzy systems that are similar to type-1 fuzzy logic systems in terms of the structure and components (Karnik et al., 1999). The only differences are that type-2 fuzzy logic system uses type-2 fuzzy sets and it has an extra output process component called the type-reducer before defuzzification. In type-2 fuzzy sets, the memberships are type-1 fuzzy sets rather than a crisp value between 0 and 1. There are two main forms of type-2 fuzzy sets: interval and general type-2 fuzzy sets. In the interval form, the third dimension is fixed to normality for the whole fuzzy set that is used as a membership function. On the other form, the third dimension is free

fuzzy set and so that it is called general. In practice, the interval form is the most used form in type-2 fuzzy sets due to its simplicity and ease of computations compared to the generalised form. Type-2 fuzzy logic systems normally require more computations than type-1 fuzzy logic systems but they offer more flexibilities and freedom in representing knowledge. Both interval and general type-2 fuzzy logic systems models (IT2FLS and GT2FLS) will be used in this thesis.

1.2.3 Simulated Annealing

The simulated annealing algorithm is a simple and general optimisation algorithm for finding global minima introduced in (Kirkpatrick et al., 1983). It has been used widely to search for optimal or nearly optimal solutions in a wide range of optimisation problems. In this work, it acts as a learning algorithm to automatically design fuzzy logic systems by searching for the best configurations of these systems.

1.3 Motivation

Fuzzy logic systems have been applied successfully to a broad range of problems in different application domains. One such type of applications is concerned with using fuzzy logic systems for system modelling and approximation where a fuzzy inference system is used to model human knowledge or to approximate non-linear and dynamic systems. However, existence of uncertainties and lack of information in many real world problems makes it difficult to model such problems using expert knowledge only. Examples of such problems include identifying systems with no known rule-base and systems with only historic data observation. It becomes clear that when designing a simple fuzzy logic system with few inputs, the experts may be able to provide efficient rules but as the complexity of the system grows, the rule base and membership functions become difficult to acquire. Therefore, some automated tuning and learning methods

are often used to cope with such situations. The objective of these methods is to get parametrised functions that best model these problems according to chosen criterion. The use of automated methods to design fuzzy logic systems helped to model many real world problems that are difficult to understand by experts and becomes one of the well known methodologies for modelling and approximation applications.

Although, type-1 fuzzy logic systems are the most well known model of fuzzy logic systems that have been given attention for decades, the advancement on research of type-2 fuzzy sets and systems and their reported success over type-1 fuzzy logic systems encouraged many researchers to apply type-2 fuzzy logic systems to model problems. In fact, the type-1 fuzzy logic approach has problems when faced with environments that have some kinds of uncertainties which exist in a large number of real world applications. All these uncertainties translate into uncertainties about membership functions (Mendel and John, 2002). Type-1 fuzzy Logic can not fully handle these uncertainties because type-1 fuzzy logic is precise in nature and for many applications it is unable to model knowledge adequately where type-2 fuzzy logic offers a higher level of imprecision (John and Coupland, 2006). The extra dimension and parameters in type-2 fuzzy sets are supposed to provide more design freedom and flexibility than type-1 fuzzy sets. To benefit from this feature, the use of some automated learning methods becomes a valuable choice as complexity grows when designing type-2 fuzzy logic systems. In consequence, finding good learning methods for learning type-2 fuzzy logic systems helps to improve their performance in modelling difficult problems.

The ability of fuzzy logic systems to be hybridised with other methods extended the usage of fuzzy logic systems in many application domains. Many approaches have been proposed to learn and tune fuzzy logic systems including using search algorithms such as genetic algorithms and simulated annealing as well as many local search algorithms and classical learning methods such as the least-squares method. Although optimisation search algorithms such as genetic algorithms were not specifically designed for learning,

they can offer some advantages for machine learning (Herrera, 2005). In fact, genetic algorithms are one of the most common search algorithms used with fuzzy logic systems that has been applied to a wide range of problems such as control system design, decision making, classification, modelling and information retrieval (Hoffmann, 2001). On the other hand, fewer number of researchers have studied the use of simulated annealing to learn type-1 fuzzy logic systems (Drack and Zadeh, 2006). Amongst those who have studied this combination are (Huyghe and Hamam, 1995; Garibaldi and Ifeachor, 1999; Liu and Yang, 2000; Cordón et al., 2000; Drack and Zadeh, 2006; Yanar and Akyrek, 2011). For type-2 fuzzy logic systems, to the author's knowledge, no work was reported in the open literature using simulated annealing to design type-2 fuzzy logic systems, despite that many of the search and learning algorithms have been used with interval type-2 fuzzy logic systems. Accordingly, the questions of how far simulated annealing can assist designing type-2 fuzzy logic systems and how they are combined worth being investigated.

Another motivation comes from the lack of applications using general type-2 fuzzy logic systems. Although, type-2 fuzzy logic is a growing research topic with much evidence of successful applications (John and Coupland, 2007), up to now, almost all developments of type-2 fuzzy logic systems were based on interval type-2 fuzzy logic systems with some exceptions related to different representations of type-2 sets and systems other than the first and main representation known as vertical-slices. The heavy computations associated with the generalised form of type-2 sets is the main driver for the lack of applications of general type-2 fuzzy sets and systems compared to the interval model. In fact, learning and optimisation of general type-2 fuzzy logic systems is an open area for more research as well as the ongoing research on how to reduce the complexity of general type-2 fuzzy logic systems especially on the type-reduction phase of the system. Hence, the large number of methods used to design type-1 and interval type-2 fuzzy logic systems can be potential candidates for general type-2 fuzzy logic systems and some of them might uncover the potential for general type-2 fuzzy logic systems in

modelling more uncertainties. However, in the last few years, some advances in general type-2 fuzzy logic systems research achieved more steps towards practicality. These advances includes new representations, optimised operations and faster type-reduction methods. Despite the higher amount of computations associated with general type-2 fuzzy sets, its extra free dimension can bring more abilities to modelling than the restricted uniform dimension in interval type-2 fuzzy sets. This ability can be unveiled using automated designing methods rather than being chosen by the designer manually. Although, automated methods can add more fine tuning to expert designs, there is no rational basis for choosing secondary membership functions in general type-2 fuzzy sets (Mendel, 2001, p.302). This issue enforces the need for automated methods in such designs. The other factor affecting the usage of general type-2 fuzzy logic systems is the lack of parametrisation methods to handle the third dimension in general type-2 fuzzy sets. In general, the general type-2 fuzzy logic system has a potential to model more uncertainties despite the large amount of computations associated with it especially when applied to non real-time applications. In consequence, the question of how much general type-2 fuzzy logic systems can add to modelling performance over interval type-2 fuzzy logic systems is another issue that worths investigation.

1.4 Objectives

The main aim of this investigation is to use simulated annealing to design high performing type-2 fuzzy logic systems automatically to model information with uncertainties. This aim is fulfilled by achieving the following objectives :

1. Firstly, to design and evaluate the combination of simulated annealing with type-1 fuzzy logic systems to model uncertain information. Although, this combination (simulated annealing and type-1 fuzzy logic systems) has been applied before, this objective has been chosen to show how the combination between simulated

annealing and fuzzy logic systems is achieved. Also, to ground the basic relationships and issues of this combination before using higher types of fuzzy logic systems.

2. To design interval type-2 fuzzy logic systems automatically using simulated annealing to model uncertain information. In order to get reliable conclusions, the applications of modelling four bench-mark problems including real-world data have been chosen. These problems are: noise-free Mackey-Glass time series forecasting (Mackey and Glass, 1977), noisy Mackey-Glass time series forecasting (Mackey and Glass, 1977) and two real world problems which are the estimation of the low voltage electrical line length in rural towns and the estimation of the medium voltage electrical line maintenance cost (Cordón et al., 1999).

3. To design general type-2 fuzzy logic systems automatically using simulated annealing to model uncertain information. This means that the third dimension in general type-2 fuzzy sets should be exploited in the learning process. The four problems mentioned above will be applied to test this model.

4. To develop a practical approach to design the third dimension of general type-2 fuzzy sets easily and with reasonable computational cost.

5. To compare type-1 and type-2 models of fuzzy logic systems in their abilities to model uncertainties associated with problems. This objective helps to answer the question of weather interval and general type-2 fuzzy logic systems can add extra abilities to modelling.

1.5 Thesis Contribution

The thesis presents a novel design of fuzzy logic systems mainly type-2 fuzzy logic systems using simulated annealing algorithm to model problems with higher levels of

uncertainty. The combination of simulated annealing with three models: type-1 fuzzy logic system, interval type-2 fuzzy logic system and general type-2 fuzzy logic system have been applied. These three models are applied in modelling four bench-mark problems. The three models are compared in their abilities to model uncertainties associated with these problems. Also, issues related to this combination between simulated annealing and fuzzy logic systems are discussed.

Regarding contribution to knowledge, this thesis presents three kinds of novelties. The first is a novel approach to design interval and general type-2 fuzzy logic systems using simulated annealing as no work has been reported in the open literature using such combination. The second one is the automatic design of first general type-2 fuzzy logic system using the vertical slice representation. In addition, the thesis proposed a new novel method to overcome some parametrisation difficulties when learning general type-2 fuzzy logic systems. This thesis helps to open the doors for more practical work in general type-2 fuzzy logic systems. The clear conclusion that learning the third dimension can add more abilities to modelling is an important advance in type-2 fuzzy logic system research and should open the doors for more promising research on using general type-2 fuzzy logic systems. Hence, the proposed parametrisation method allow other learning methods used to design interval type-2 fuzzy logic systems to be potential candidates for general type-2 fuzzy logic systems design.

Parts of this thesis have been published or submitted to a number of publications (Almaraashi, 2010; Almaraashi et al., 2010; Almaraashi and John, 2011*b*, 2010, 2011*a*; Almaraashi, John and Ahmadi, 2012*b*,*a*; Almaraashi, John and Coupland, 2012). One of these publications has won the best student paper award in a conference (Almaraashi and John, 2011*a*). These publications are attached in the thesis appendices.

1.6 Thesis Structure

The rest of the thesis is divided into six chapters as follows:

Second Chapter reports the literature of the methods and concepts used within the thesis. The most relevant aspects of these methods are concerned with type-1, type-2 fuzzy logic systems, learning of fuzzy logic systems and the simulated annealing algorithm as a method that carries out the learning.

Third Chapter presents the work of learning type-1 fuzzy logic systems using simulated annealing with two fuzzy logic systems models (Mamdani and Takagi-Sugeno-Kang (TSK)) and under two fuzzification models (singleton and non-singleton fuzzifications). Theses models are used to predict the well known Mackey-Glass time series with six different noise levels. The results of the proposed methods are compared by their ability to handle uncertainty as well as comparing the noisy free time series with other works in the literature. In this chapter, a description of the problems applied in the thesis including next chapters is presented. Another work on type-1 fuzzy logic systems will be presented in the fourth chapter for comparison and consistency purposes.

Fourth Chapter presents the work of learning interval type-2 fuzzy logic systems using simulated annealing . This chapter discusses some of the issues of this combination. The proposed method is applied on four bench-mark problems including real world problems. These are noise-free and noisy Mackey-Glass time series forecasting (Mackey and Glass, 1977) and two real world problems which are the estimation of the low voltage electrical line length in rural towns and the estimation of the medium voltage electrical line maintenance cost (Cordón et al., 1999). The results of the interval type-2 fuzzy logic systems are compared to the results of a type-1 fuzzy logic systems in these experiments.

Fifth Chapter presents the work for designing general type-2 fuzzy logic systems with the aid of interval type-2 fuzzy logic systems and simulated annealing. Issues and difficulties of manual and automatic design of general type-2 fuzzy logic systems are discussed as well as proposing a practical solution to some parametrisation issues of the third dimension in general type-2 fuzzy logic systems. The proposed method is applied on the four bench-mark problems mentioned above and their results and analysis are discussed.

Sixth Chapter reports the work for learning all parameters of general type-2 fuzzy logic systems using simulated annealing . The proposed method is applied on the four bench-mark problems mentioned above and their results and analysis are discussed.

Seventh Chapter provides the conclusion and some potential future work. It summaries the contribution achieved in this work and its impacts on the field. Then, a potential future work is suggested based on what achieved in this thesis.

Chapter 2

Background

2.1 Introduction

This chapter presents needed literature for the methods and concepts that are used within the thesis. The most relevant aspects of these methods and concepts are concerned with type-1 and type-2 fuzzy logic systems, learning of fuzzy logic systems and the simulated annealing algorithm as a method that carries out the learning. The chapter presents a description of these components that are needed to understand the work in the following chapters in three sections. Firstly, it starts by reviewing the main parts in the theory of type-1 fuzzy sets and systems where they are used in chapter 3 and 4. In the second section, type-2 sets and systems are defined. The literature of type-2 fuzzy sets and systems is needed in chapter 4,5 and 6. The third section explains some aspects related to the learning of fuzzy logic systems including the issues linked to the combination of both fuzzy logic systems and learning and optimisation methods. Next, simulated annealing is described with detailed description of its components.

2.2 Type-1 Fuzzy Sets and Systems

2.2.1 Type-1 fuzzy sets

Fuzzy logic is a branch of many-valued logic uses the theory of fuzzy sets proposed by Zadeh in 1965 in his paper "Fuzzy sets" (Zadeh, 1965). In a fuzzy set, any element in the set is given a degree of membership of this set as opposed to the ordinary crisp set where its membership is characterised by two values only. For example, if set A is a crisp set in the universe of discourse X, then the membership function of set A (also called a characteristic function) is defined as :

$$\mu_A(x) = \begin{cases} 1 & \text{if } x \in X \\ 0 & \text{if } x \notin X \end{cases}$$

Fuzzy sets of type-1 are normally known as fuzzy sets. Fuzzy sets are fully described by their membership functions. For example, for a fuzzy set A in the universe of discourse X, the membership function for A is defined as :

$$\mu_A(x) : X \to [0, 1]$$

Where each element in the set is given a number between zero and one as a membership grade. When the set A is discrete, the notation is adopted as follows :

$$A = \mu_A(x_1)/x_1 + \mu_A(x_2)/x_2 + \cdots + \mu_A(x_n)/x_n$$

When the fuzzy set A is continuous, it is defined as follows :

$$A = \int_x \mu_A(x)/x$$

In both cases, $x_1, x_2, .., x_n$ are members of the set A and $\mu_A(x_1), \mu_A(x_2), \ldots \mu_A(x_n)$ are their membership grades respectively. Figure 2.1 shows an example of a type-1 fuzzy

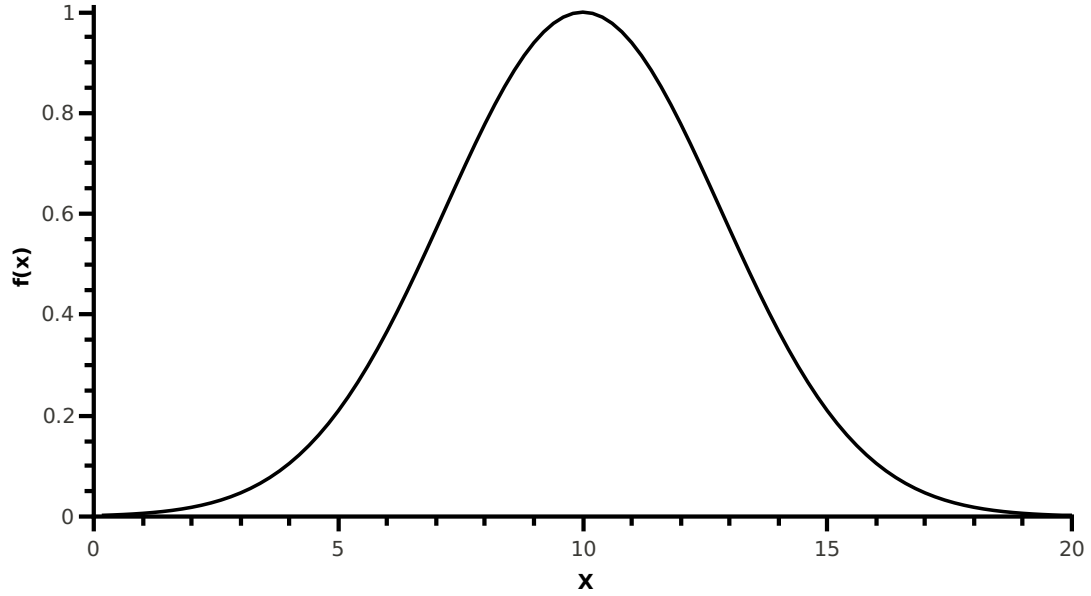

FIGURE 2.1: Type-1 fuzzy set "About 10".

set called "About 10". In this example, crisp values are given memberships based on how close they are from 10. For example, $\mu_A(5) = 0.2$ and $\mu_A(8) = 0.7$.

2.2.2 Type-1 fuzzy sets operations

In fuzzy set theory, there are many ways to define fuzzy operations. Operations of fuzzy sets can be defined using their membership functions by extending the crisp sets operations where the operation result is another fuzzy set. There are many ways proposed in the literature to model these operators. For example, fuzzy union of two fuzzy sets A and B using maximum operator is a fuzzy set defined by the membership function (Dubois and Prade, 1980):

$$\mu_{A \cup B}(x) = max[\mu_A(x), \mu_B(x)] = \mu_A(x) \vee \mu_B(x)$$

while fuzzy intersection operation of two fuzzy sets A and B using minimum operator is defined as :

$$\mu_{A\cap B}(x) = min[\mu_A(x), \mu_B(x)] = \mu_A(x) \wedge \mu_B(x)$$

and using product operator:

$$\mu_{A\star B}(x) = prod[\mu_A(x), \mu_B(x)] = \mu_A(x) * \mu_B(x)$$

The general name for these operations is t-norm and t-conorm when satisfying some mathematical properties. T-norm and t-conorm are defined as follows (Dubois and Prade, 1980):

- T-norm (Triangular Norms) : is the name for the general form of intersection functions in fuzzy sets. The most common t-norm used in fuzzy systems are the minimum operator $min(a, b)$ and the product operator $product(a, b)$.
- T-conorm (Triangular Conorms) : is the name for the general form of union functions in fuzzy sets. The most common t-conorm used in fuzzy systems is the maximum operator $max(a, b)$.

2.2.3 Type-1 fuzzy logic systems

A type-1 fuzzy system is a rule based system which can be viewed as a process that maps crisp inputs to outputs by using the theory of fuzzy sets (Negnevitsky, 2002, p. 106). The well-known Mamdani fuzzy model (Mamdani, 1974) contains four components: fuzzifier, rules, inference engine and output processor (defuzzifier) (Mendel, 2001, p. 6). The Takagi-Sugeno-Kang fuzzy model (TSK) model (Takagi and Sugeno, 1985) is different from the Mamdani model in the inference engine and the output processor. Figure 2.2 shows these components which are (Mendel, 2001, p. 6):

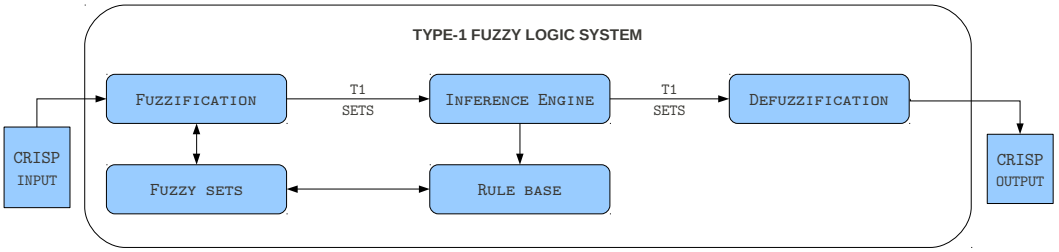

FIGURE 2.2: Type-1 fuzzy logic system

2.2.3.1 Fuzzifier

Fuzzifier maps crisp inputs to fuzzy sets by evaluating the crisp inputs $x = (x_1, x_2, \ldots, x_n)$ based on the antecedents part of the rules and assigns each crisp input a degree of membership $\mu_{A_i}(x_i)$ in its input fuzzy set. There are two types of fuzzifiers, singleton and non-singleton. The singleton fuzzifier maps crisp inputs to fuzzy sets by evaluating the crisp inputs $x = (x_1, x_2, \ldots, x_n)$ based on the antecedents part of the rules and assigns each crisp input to its fuzzy set $A(x)$ in X with its degree of membership in each fuzzy set. The non-singleton fuzzifier maps each given input x_i into a fuzzy set (known as variability set) with a unity membership grade for x_i while their neighbours values are given lesser membership values as they move away from x_i (Mouzouris and Mendel, 1994). In this case the fuzzifier considers x_i as the most likely correct value among its neighbours values and normally is the centre of the fuzzy set (Mendel, 2001, p. 188). Non-singleton fuzzification allows better modelling of input uncertainties (using the variability set) and linguistic uncertainties (using antecedent fuzzy sets) in two stages (Wagner and Hagras, 2010*b*). Figure 2.3 shows how these two fuzzifiers fuzzify crisp inputs.

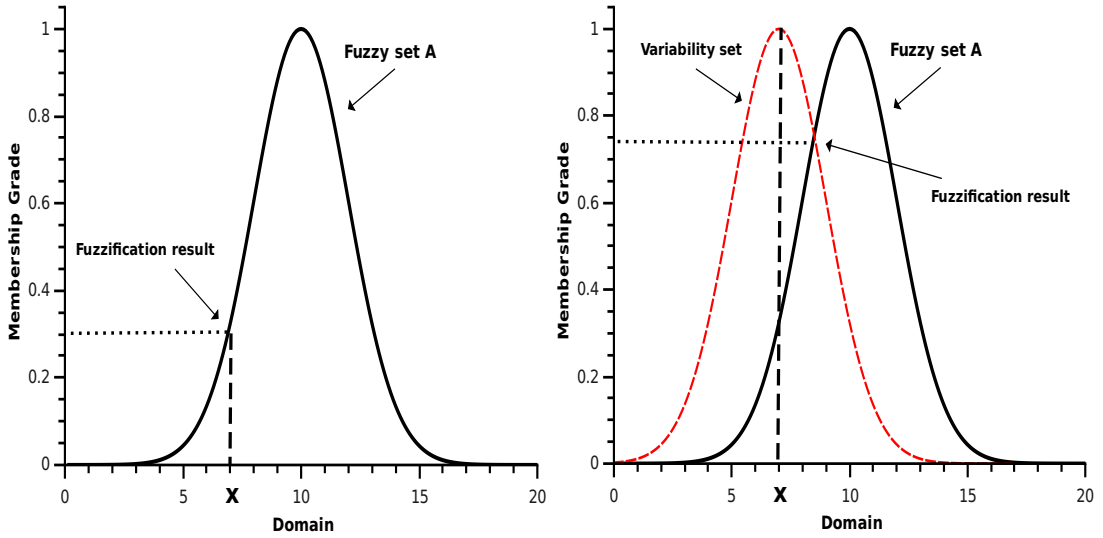

FIGURE 2.3: The fuzzification of a crisp input x using singleton (left) and non-singleton (right) fuzzification

2.2.3.2 Rules

A fuzzy rule is a conditional statements in the form of IF-THEN where it contains two parts, the IF part called the antecedent part and the THEN part called the consequent part. For example :

IF job risk is high THEN salary is high.

These rules are written in fuzzy forms using words such as high, short and slow. To acquire these rules, many methods can be used such as getting them from experts or using data driven methods.

2.2.3.3 Inference engine

The inference engine in the Mamdani model maps the input fuzzy sets into the output fuzzy sets then the defuzzifier converts them to a crisp output. The rules in the Mamdani model have fuzzy sets in both the antecedent part and the consequent part. For

example, the ith rule in the Mamdani model can be described as follows:

$$R^i: \text{ IF } x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i ... \text{ and } x_p \text{ is } A_p^i$$

$$\text{THEN } y \text{ is } B^i$$

Where $x_1, x_2, ..., x_p$ are the input variables to the fuzzy systems and $A_1^i, A_2^i, .., A_p^i$ are input fuzzy sets in the antecedent part. The consequent part consists of output variable y and its output fuzzy set B^i. The operation "and" is normally modelled by a t-norm operator.

2.2.3.4 Defuzzifier

Defuzzifier in Mamdani model converts the output fuzzy sets that have been produced by inference engine into crisp values. Defuzzification can be done by many methods that have been proposed in the literature such as centroid defuzzifier, centre of sets defuzzifier, height defuzzifier and centre of sums defuzzifier. The inference and defuzzification processes in Takagi and Sugeno model (Takagi and Sugeno, 1985) are different from the Mamdani model as the rules in TSK model are based on input fuzzy sets in the antecedent part and a mathematical linear function in the consequent part. The ith rule in the first-order TSK model is described as follows:

$$R^i: \text{ IF } x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i ... \text{ and } x_p \text{ is } A_p^i$$

$$\text{THEN } y^i(x) = c_0^i + c_1^i * x_1 + c_2^i * x_2 ... + c_p^i * x_p$$

Where i represents the rule number and c_0^i, c_1^i, c_2^i and c_p^i are the the coefficients of the consequent part of the fuzzy system rules. The final control value Y is computed as follows:

$$Y = \frac{\sum_{i=1}^{n} \alpha_i * y_i}{\sum_{i=1}^{n} \alpha_i}$$

Where α_i is known as the firing level for the ith rule which is derived by using a t-norm operator such as minimum or product. It is important to mention that while the Mamdani model can handle uncertainty in both antecedent and consequent parts, TSK can handle uncertainty only in the antecedent part where fuzzy sets are used which limits its applicability in situations either where there is no uncertainty or the uncertainty exists only in the antecedent part (Mendel, 2001, p. 188).

2.3 Type-2 Fuzzy Sets and Systems

2.3.1 Fuzzy sets and uncertainty

The existence of uncertainties and the lack of information in many real world problems makes it difficult to model some real-world problems. Uncertainty in fuzzy set theory has been recognised and divided into three types (Klir and Wierman, 1998, p. 43-44) which are:

- Fuzziness (or vagueness): This results from the imprecise boundaries of fuzzy sets.
- Non-specificity (or imprecision): It is linked to relevant sets of alternatives when some alternative belongs to a specific set of alternatives but we do not know which one in the set it is.
- Strife (or discord): which expresses conflicts among the various sets of alternatives.

Type-1 fuzzy logic has been used successfully in a wide range of problems such as control system design, decision making, classification, system modelling and information retrieval (Ross, 2004; Hoffmann, 2001). However, type-1 approach is not able to directly model uncertainties and minimise its effects (Mendel and John, 2002). These

uncertainties exist in a large number of real world applications. It can be a result of uncertainty in inputs, uncertainty in outputs, uncertainty that is related to the linguistic differences, uncertainty caused by the change of conditions in the operation and uncertainty associated with the noisy data when training the fuzzy logic system (Mendel, 2001, p.68). All these uncertainties translate into uncertainties about fuzzy sets membership functions (Mendel and John, 2002). Therefore, existence of uncertainties in the majority of real world applications makes the use of type-1 fuzzy logic inappropriate in many cases especially with problems related to inefficiency of performance in fuzzy logic control (Hagras, 2007). Also, it is arguable weather human brain uses crisp images of membership functions (Zimmermann, 2001, p. 24). Problems related to modelling uncetainity using crisp membership functions of type-1 fuzzy sets have been recognized early and (Zadeh, 1975) introduced higher types of fuzzy sets called type-n fuzzy sets including type-2 fuzzy sets (Mendel, 2003)(Zimmermann, 2001, p. 24). Type-2 fuzzy logic systems might have many advantages compared with type-1 fuzzy logic systems. These advantages include (Hagras, 2007):

- Type-2 fuzzy set can handle numerical and linguistic uncertainties because its membership function is fuzzy and has a footprint of uncertainty (FOU) while type-1 fuzzy sets membership function is precise.
- Using type-2 fuzzy sets to represent inputs and outputs results in using less rules compared with using type-1 fuzzy sets as a result of the wider coverage obtained by footprint of uncertainty (FOU).
- Type-2 fuzzy sets embed a large number of type-1 fuzzy sets to describe variables with a detailed description adding extra levels of smooth control surface and response.
- The extra dimension provided by the FOU enables a type-2 fuzzy logic system to produce outputs that cannot be achieved by type-1 fuzzy logic system using the same number of membership functions (Wu and Tan, 2005a).

Two factors should be considered regarding the widespread perception that general type-2 fuzzy logic system should outperform the interval form which also should outperform type-1 fuzzy logic system (Wagner and Hagras, 2010*b*). These two factors are the dependence of performance on the choice of the model parameters as well as on the variability of uncertainty within the application (Wagner and Hagras, 2010*b*). Therefore, a good choice of the model's parameters using automated methods is desired to get more clearer conclusions regarding this comparison. Type-2 fuzzy logic is a growing research area with much evidence of successful applications (John and Coupland, 2007; Mendel, 2007).

2.3.2 Type-2 fuzzy sets

A type-2 fuzzy set (Mendel and John, 2002), denoted by $\tilde{A}$, is characterized by a type-2 membership function $\mu_{\tilde{A}}(x,u)$ where $x \in X$ and $u \in J_x \subseteq [0,1]$. For example :

$$\tilde{A} = \{((x,u), \mu_{\tilde{A}}(x,u)) \mid \forall x \in X, \forall u \in J_x \subseteq [0,1]\}$$

where $0 \leq \mu_{\tilde{A}}(x,u) \leq 1$. Set $\tilde{A}$ also can be expressed as:

$$\tilde{A} = \int_{x \in X} \int_{u \in J_x} \mu_{\tilde{A}}(x,u)/(x,u), J_x \in [0,1]$$

where $\int$ denotes union. When universe of discourse is discrete, Set $\tilde{A}$ is described as :

$$\tilde{A} = \sum_{x \in X} \sum_{u \in J_x} \mu_{\tilde{A}}(x,u)/(x,u), J_x \in [0,1]$$

When all the secondary grades $\mu_{\tilde{A}}(x,u)$ equal 1, then, $\tilde{A}$ is an interval type-2 fuzzy set. Interval type-2 fuzzy sets are easier to compute than general type-2 fuzzy sets. See figures 2.4 and 2.5 for examples of a general and an interval type-2 fuzzy sets and figure 2.6 for a 2D representation of type-2 set called footprint of uncertainty (FOU)

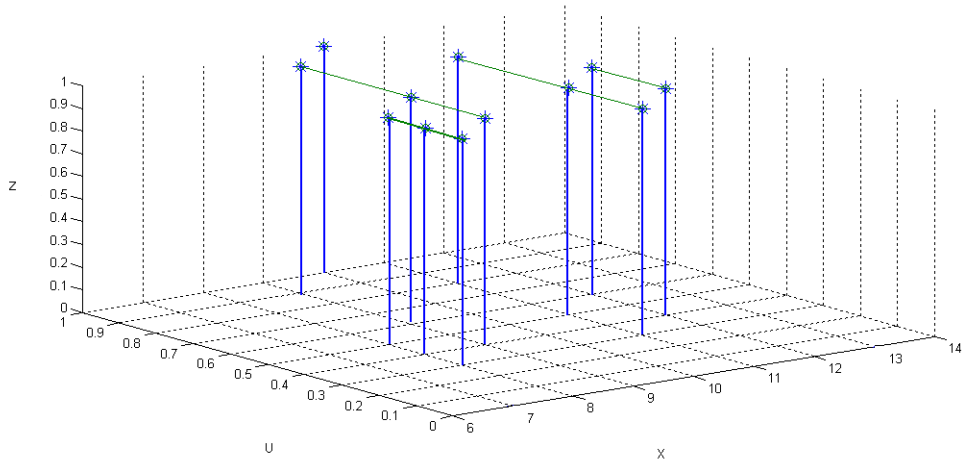

FIGURE 2.4: Interval type-2 fuzzy set "About 10".

which represents the union of all primary membership grades and can be described easily by a lower and upper membership functions. The ease of computation and representation of interval type-2 fuzzy sets is the main driver for their wide usage in real world applications. A well known simple form of type-2 fuzzy sets is based on the principal membership function. When at each secondary membership function of type-2 set there is only one secondary grade equals to 1, Then, the principal membership function is the union of all points with unity membership grades (Mendel, 2001, p.86). Therefore:

$$\mu_{principal}(x) = \int_{x \in X} u/x \text{ where } f_x(u) = 1 \tag{2.1}$$

2.3.3 Type-2 fuzzy sets representation

There are some representations for type-2 fuzzy sets have been proposed in the literature such as vertical-slice representation (Mendel and John, 2002), wavy-slice representation

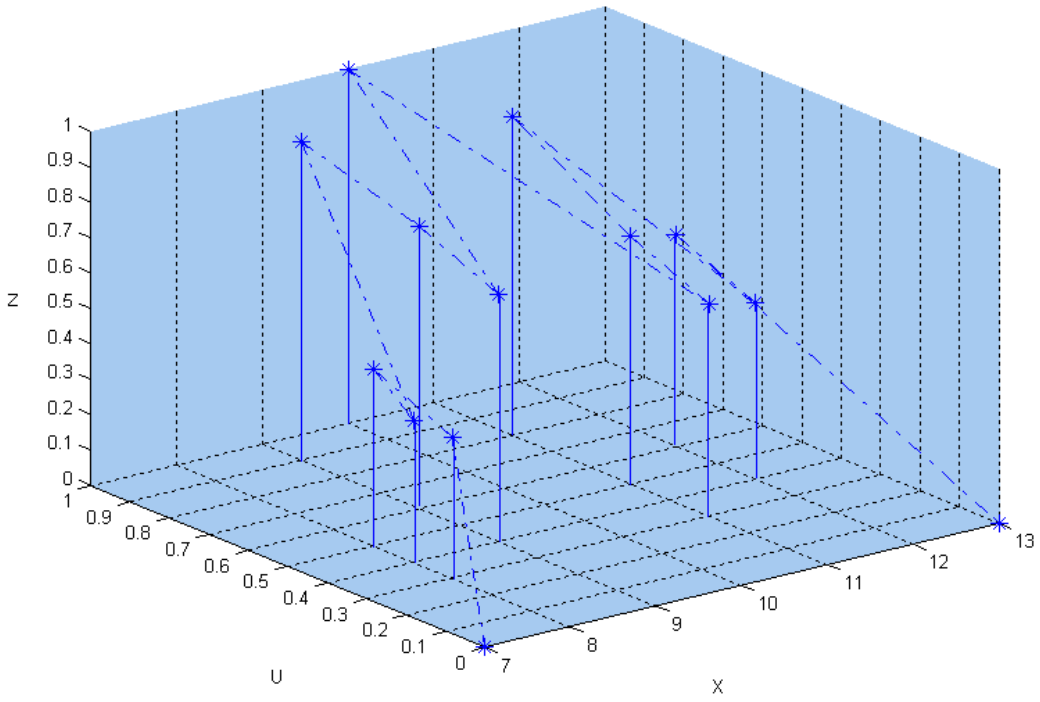

FIGURE 2.5: General type-2 fuzzy set "About 10".

(Mendel and John, 2002), geometric representation (Coupland and John, 2007), alpha-planes (Mendel et al., 2009), alpha cuts (Hamrawi et al., 2010) and Z-slices (Wagner and Hagras, 2010*a*). Here we focus on the first known representations which are:

- Vertical-Slice representations (Mendel and John, 2002): which represents fuzzy sets by using secondary sets in a vertical-slice manner where :

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) \mid \forall x \in X\} \tag{2.2}$$

  This representation is very useful for computation.

- Wavy-Slice representations (Mendel and John, 2002): In wavy-slice representation, Type-2 fuzzy set is represented as a union of embedded type-2 fuzzy sets where each embedded type-2 fuzzy set $\tilde{A}_e$ has the same domain of type-2 fuzzy

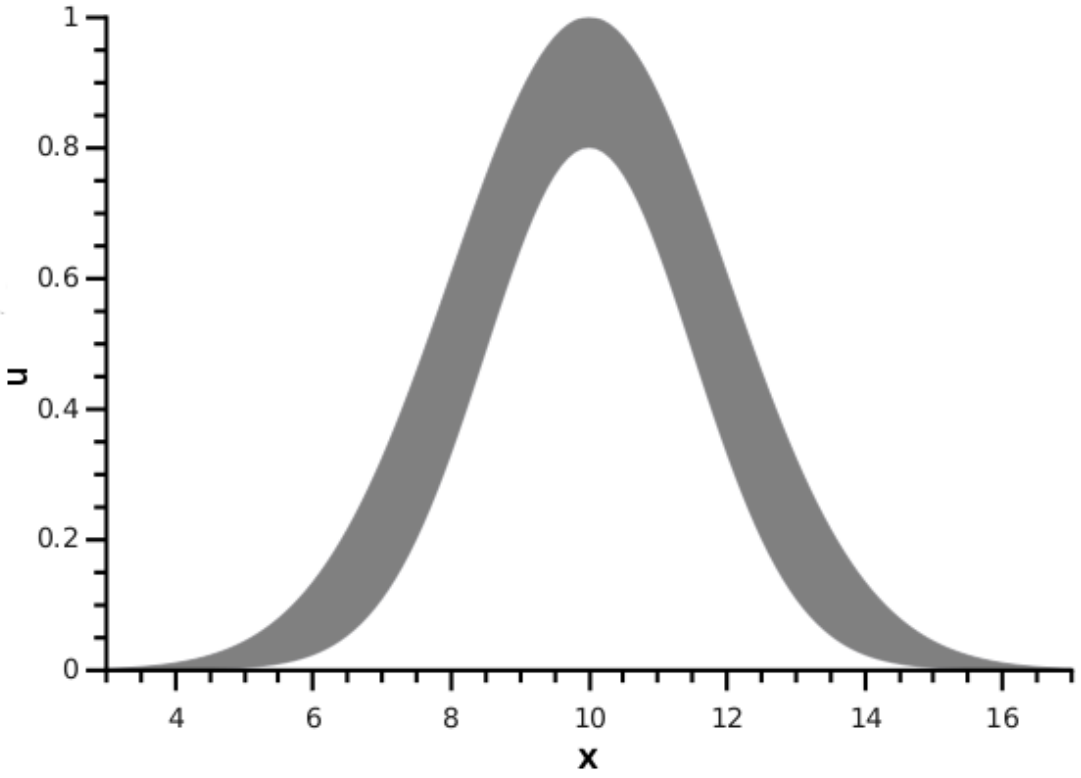

FIGURE 2.6: Footprint of uncertainty (FOU) for type-2 fuzzy set "About 10".

set $\tilde{A}$. Type-2 embedded set $\tilde{A}_e$ has been defined by (Mendel and John, 2002) as follows:

> For discrete universes of discourse X and U, an embedded type-2 set $\tilde{A}_e$ has N elements, where $\tilde{A}_e$ contains exactly one element from J_{x_1}, $J_{x_2}, ..., J_{x_N}$, namely $u_1, u_2, ..., u_N$, each with associated secondary grade, namely $f_{x_1}(u_1)$, $f_{x_2}(u_2), ..., f_{x_N}(u_N)$

For example:

$$\tilde{A}_e = \sum_{i=1}^{N} [f_{x_i}(u_i)/u_i]/x_i, u_i \in J_{xi} \subseteq U = [0, 1]. \tag{2.3}$$

As we see in this definition, the embedded set contains N number of elements represented using the primary memberships $u_i \in J_{x_i}$ that is linked to its secondary

membership grades $f_{x_i}(u_i)$ in ordered pairs. So that type-2 fuzzy set A can be shown as a union of embedded type-2 fuzzy sets as follows:

$$\tilde{A} = \sum_{j=1}^{n} \tilde{A}_e^j \tag{2.4}$$

where the total number of type-2 embedded sets in type-2 fuzzy set A is calculated using the number of discretised points in the primary domain N and the secondary domain M as follows:

$$n = \prod_{i=1}^{N} M_i \tag{2.5}$$

And $\tilde{A}_e^j$ denotes the jth type-2 embedded fuzzy set in type-2 fuzzy set $\tilde{A}$. Wavy-slice representation known as Mendel-John Representation Theorem (RT) has been proposed by (Mendel and John, 2002). It is useful for theoretical derivations but not useful for practical use because of the astronomical number in the union of embedded sets. However, it is very useful when dealing with interval type-2 fuzzy sets due to the ability of using type-1 fuzzy mathematics which is easy to deal with (Mendel et al., 2006).

2.3.4 Type-2 fuzzy sets operations

Based on Zadeh extension principle (Zadeh, 1975), type-2 fuzzy sets operations can be derived as follows (Mizumoto and Tanaka, 1976; Karnik and Mendel, 2001*b*): Let A and B be two type-2 fuzzy sets in universe X while $\mu_{\tilde{A}}(x)$ and $\mu_{\tilde{B}}(x)$ are their membership grades respectively where $\mu_{\tilde{A}}(x) = \int_u f_x(u)/u$ and $\mu_{\tilde{B}}(x) = \int_w g_x(w)/w$ while the primary membership of x are $u, w \in J_x, J_x \subseteq [0,1]$ and secondary membership grades of x are $f_x(u), g_x(w) \in [0,1]$. Using Zadeh extension principle, union, intersection and complement can be defined as (Mizumoto and Tanaka, 1976; Karnik and Mendel, 2001*b*) :

- Union:

$$\tilde{A} \cup \tilde{B} \Leftrightarrow \mu_{\tilde{A} \cup \tilde{B}}(x) = \mu_{\tilde{A}}(x) \sqcup \mu_{\tilde{B}}(x) = \int_u \int_w (f_x(u) \star g_x(w))/(u \oplus w) \qquad (2.6)$$

- Intersection:

$$\tilde{A} \cap \tilde{B} \Leftrightarrow \mu_{\tilde{A} \cap \tilde{B}}(x) = \mu_{\tilde{A}}(x) \sqcap \mu_{\tilde{B}}(x) = \int_u \int_w (f_x(u) \star g_x(w))/(u \star w) \qquad (2.7)$$

where $\star$ represents a t-norm which is called meet operation and $\oplus$ represents the max t-conorm that is called join operation. Karnik (Karnik and Mendel, 2001*b*) has proposed a method to calculate meet and join operations when all secondary membership functions are normal and convex. Coupland (Coupland and John, 2007) has presented an extension to this formula to allow the use of non-normal secondary membership functions. Suppose that there are two type-2 fuzzy sets $\tilde{A}$ and $\tilde{B}$ where all the secondary membership functions in $\tilde{A}$ and $\tilde{B}$ are convex. The apex is the point at maximum value in the secondary membership function. Let v_1 and v_2 be real numbers such that $v_1 \leqslant v_2$ and $\mu_{\tilde{A}}(x, v_1)$ is the apex of $\tilde{A}$ and $\mu_{\tilde{B}}(x, v_1)$ is the apex of $\tilde{B}$. Then, the join and meet operations under minimum t-norm and maximum t-conorm are given by (Coupland and John, 2007):

$$\mu_{\tilde{A} \sqcup \tilde{B}}(x, u) = \begin{cases} \mu_{\tilde{A}}(x, u) \wedge \mu_{\tilde{B}}(x, u), & \text{if } u < v_1 \\ \mu_{\tilde{A}}(x, v_1) \wedge \mu_{\tilde{B}}(x, u), & \text{if } v_1 \leqslant u < v_2 \\ (\mu_{\tilde{A}}(x, u) \vee \mu_{\tilde{B}}(x, u)) \wedge (\mu_{\tilde{A}}(x, v_1) \wedge \mu_{\tilde{B}}(x, v_2)), & \text{if } u \geqslant v_2 \end{cases}$$

$$\mu_{\tilde{A} \sqcap \tilde{B}}(x, u) = \begin{cases} (\mu_{\tilde{A}}(x, u) \vee \mu_{\tilde{B}}(x, u)) \wedge (\mu_{\tilde{A}}(x, v_1) \wedge \mu_{\tilde{B}}(x, v_2)), & \text{if } u < v_1 \\ \mu_{\tilde{A}}(x, u) \wedge \mu_{\tilde{B}}(x, v_2), & \text{if } v_1 \leqslant u < v_2 \\ \mu_{\tilde{A}}(x, u) \wedge \mu_{\tilde{B}}(x, u), & \text{if } u \geqslant v_2 \end{cases} \qquad (2.8)$$

2.3.5 Type-2 fuzzy logic systems

A type-2 fuzzy logic system is a rule based system that is similar to a type-1 fuzzy logic system in terms of its structure and components. The only differences are that a type-2 fuzzy logic system uses at least one type-2 fuzzy set and it has an extra output process component called the type-reducer before defuzzification as shown in figure 2.7. The type-reducer reduces output type-2 fuzzy sets to type-1 fuzzy sets then the defuzzifier reduces it to a crisp output. The components of a type-2 Mamdani fuzzy system are (Karnik et al., 1999):

2.3.5.1 Fuzzifier

Fuzzifier maps crisp inputs into type-2 fuzzy sets by evaluating the crisp inputs $x = (x_1, x_2, \ldots, x_p)$ based on the antecedents part of the rules and assigns each crisp input to its type-2 fuzzy set $\tilde{A}(x)$ with its membership grade in each type-2 fuzzy set. The resulting fuzzified value is a type-1 fuzzy set representing the secondary grades for each input. When using interval type-2 fuzzy logic systems, the resulting value is an interval type-1 fuzzy set.

2.3.5.2 Rules

A fuzzy rule in type-2 fuzzy logic system is also a conditional statement in the form of IF-THEN where it contains two parts, the IF part called the antecedent part and the THEN part called the consequent part. The only difference from type-1 fuzzy logic system rules is that rules in type-2 fuzzy logic system use type-2 fuzzy sets and can use type-1 fuzzy sets. Using one type-2 fuzzy set is enough to consider the system as a type-2 fuzzy logic system.

2.3.5.3 Inference Engine

Inference Engine maps input type-2 fuzzy sets into output type-2 fuzzy sets by applying the consequent part where this process of mapping from the antecedent part into the consequent part is interpreted as a type-2 fuzzy implication which needs computations of union and intersection of type-2 fuzzy sets. The inference engine in a Mamdani type-2 fuzzy logic system maps the input type-2 fuzzy sets into the output type-2 fuzzy sets. The rules in Mamdani model have type-2 fuzzy sets in both the antecedent and the consequent parts. For example, the ith rule in a Mamdani rule base can be described as follows:

$$R^i: \text{ IF } x_1 \text{ is } \tilde{A}_1^i \text{ and } x_2 \text{ is } \tilde{A}_2^i ... \text{ and } x_p \text{ is } \tilde{A}_p^i \tag{2.9}$$

$$\text{THEN } y \text{ is } \tilde{B}^i \tag{2.10}$$

2.3.5.4 Output Processor

There are two stages in the output process:

1. **Type-Reduction**: Type-reducer reduces type-2 fuzzy sets that have been produced by the inference engine to type-1 fuzzy sets by performing a centroid calculation (Mendel, 2001). For example, the centre of sets type-reducer replaces each rule type-2 consequent set by its centroid which is a type-1 set and then calculate a weighted average of these centroids to get a type-1 fuzzy set (Karnik et al., 1999). This is the bottleneck of the type-2 fuzzy logic systems as this process requires expensive computations especially when using non-interval type-2 sets.

   (a) **Type-reduction in interval type-2 fuzzy sets**: For interval type-2 fuzzy sets, (Karnik and Mendel, 2001*a*) developed an algorithm that calculates the centroid of an interval type-2 fuzzy set which is widely known as Karnik-Mendel (KM) iterative algorithm followed by an improved version called

Enhanced Karnik-Mendel Algorithm in (Wu and Mendel, 2009). (Greenfield, Chiclana, Coupland and John, 2009) presented another method for the same purpose known as the collapsing method with some variants of the directions used when collapsing (Greenfield, Chiclana and John, 2009). Other works to type-reduce interval type-2 fuzzy sets have been reported on the literature such as uncertainty bounds (Wu and Mendel, 2002), WuTan (WT) method (Wu and Tan, 2005*b*) and Nie-Tan (NT) method (Nie and Tan, 2008). Such methods encouraged researchers to use interval type-2 fuzzy logic systems in practical applications. Here we explain the collapsing method.

As interval type-2 fuzzy set can be seen as a blurred form of type-1 fuzzy set, the collapsing method carries out the reversal of the blurring process (Greenfield, Chiclana, Coupland and John, 2009). It converts an interval type-2 fuzzy set into a type-1 representative embedded set (RES), whose defuzzified values approximates the defuzzified value of type-2 fuzzy sets. Therefore, it reduces the computational burden of interval type-2 fuzzy set defuzzification. The membership of the type-1 representative embedded set (RES) R that is derived from a discretised interval type-2 fuzzy set $\tilde{F}$ is calculated from the following approximation formula (Greenfield, Chiclana, Coupland and John, 2009):

$$\mu_R(x_i) = \mu_L(x_i) + r_i$$

Where

$$r_i = \frac{(\|L\| + \sum_{j=1}^{i-1} r_j) + b_i}{2 + (\|L| + \sum_{j=1}^{i-1} r_j) + b_i}$$

Where L is the lower membership function, U is the upper membership function, b_i is the blur size for vertical-slice i and r_i is the amount that needed to add to the y-value of the lower membership function L to get the value of R. This approximation iteratively collapses vertical slices and it is reffered as the

"Simple Representative Embedded Set Approximation (RESA)". The order of slice collapsing can be carried out in many variant: forward, backward, inward, outward, the composite variants of forward-backward and outward right-left (Greenfield, Chiclana and John, 2009).

(b) **Type-reduction in general type-2 fuzzy sets**: In general type-2 sets, the type-reduction is more complicated and computationally expensive. A brute-force highly computationally expensive type-reduction strategy was described by (Mendel, 2001, p.248-254) is the first known method. This method computes the union of all the centroids of all the embedded type-2 fuzzy sets involved in the general type-2 fuzzy set. This is impractical in real world applications as the number of embedded sets are normally astronomical. Recently, a recursive algorithm was introduced by (Gafa and Coupland, 2011) includes some interesting ideas to reduce these computations but the complexity is still very high and no practical results have been published yet. The more closer type-reducers to practicality on the open literature are the geometric defuzzifier based on the geometric representation (Coupland and John, 2007), the sampling algorithm which uses a random samples of these embedded sets to find an approximation to the exact value (Greenfield et al., 2005) followed by the importance sampling defuzzifier (Linda and Manic, 2010) and a centroid defuzzifier based on the alpha-plane representation (Liu, 2008). Another way apart from using the embedded sets is to use the centroids of all the vertical slices. The vertical slice centroid type-reducer (VSCTR) which was initially proposed by (John, 2000) then detailed by (Lucas et al., 2007) does not calculate the union for all the embedded sets involved in the general type-2 fuzzy sets. Although, this method does not depend on the concept of embedded sets, it is a good approach for practical usage. (Lucas et al., 2007). Therefore, this is an approximation of the centroid as long as the operation of union of all embedded type-2

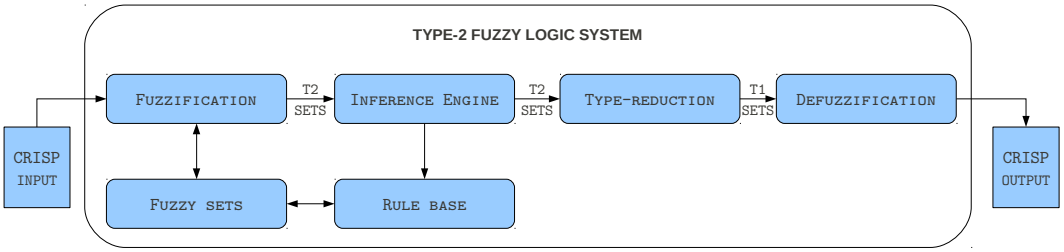

FIGURE 2.7: The components of a type-2 fuzzy logic system (adapted from (Mendel, 2001))

fuzzy sets involved in the general type-2 fuzzy set has not been carried out. When using other representation methods, there are few proposed methods that have been used for this purpose. For example, a type-reducer proposed by (Starczewski, 2009*b*) using triangular type-2 fuzzy sets which uses fuzzy truth numbers where all the secondary membership functions are normal for a unique entity in the secondary domain and convex. This type-reducer uses the iterative KM algorithm and some interpolation operations to get an approximate centroid for triangular type-2 fuzzy sets. Other methods that use other representations include the one proposed in (Liu, 2008) which uses the iterative KM algorithm under the alpha-plane representation and the one based on z-slices in (Wagner and Hagras, 2009).

2. **Defuzzification**: Defuzzifier maps the reduced output type-1 fuzzy sets that have been reduced by the type-reducer into crisp values exactly as the case of defuzzification in type-1 fuzzy logic systems. Any defuzzification methods of type-1 fuzzy sets can be used here.

2.4 Learning of Fuzzy Logic Systems

2.4.1 Fuzzy Logic System Learning

One of the features of fuzzy logic systems is their ability to be built by a large number of choices. For example, fuzzy systems can be designed to give the best output by adapting fuzzy system structure components such as rules, inference operators, fuzzification and defuzzification methods or by adapting the numeric values of the fuzzy system parameters. This feature enforces the need to research for the best configuration of fuzzy systems to best suit specific problems. In fact, this is the area where soft computing is applied with a wide range of methods and applications. Although optimisation search algorithms such as genetic algorithms were not specifically designed for learning, they can offer some advantages for machine learning (Herrera, 2005). When designing a system that operates in stable conditions, there is no need for further learning or tuning as the exact mathematical model can be used while this necessity arises in uncertain condition situations where parameters should be changed depends on the situation (Reznik, 1997, p.153).

Many machine learning methodologies are based on a search for a good model within a space of possible models such as the space of rule sets allowing these types of methodologies to model the learning process as a search problem (Herrera, 2005). These methods try to get the best configuration that gives a high performance by minimising an error function which is defined by the system behaviour or the evaluation of training example sets (Alcala et al., 1999). In general, fuzzy systems can be seen as a system with two components (Alcala et al., 1999):

1. The inference system which is concerned with the fuzzy inference process and inference operators.

2. The knowledge base (KB) which represents the knowledge about the problem such as rules and membership values. It can be divided into two parts (Herrera, 2005):

    - Data base (DB): This part contains the definitions of the fuzzy sets associated to the linguistic terms that are involved in linguistic rules (Alcala et al., 1999).
    - Rule base (RB): This contains a collection of linguistic rules that are fired simultaneously for the same input (Herrera, 2005).

To distinguish between learning and tuning problems, in the learning process, an elaborated search in the space of possible rule base (RB) or the whole knowledge base (KB) to design the fuzzy system automatically starting from scratch and does not depend on a predefined set of rules (Cordon et al., 2004). In the tuning "adaptation" process, the focus on optimising the existing fuzzy system while the predefined rule base (RB) and the preliminary data base (DB) are used within the process of finding the best set of parameters to define the data base (DB) (Alcala et al., 1999). The tuning and learning processes of fuzzy systems are different depending on the kind of fuzzy system used whether it is an approximate or descriptive (linguistic) (Cordón, Herrera, Hoffmann and Magdalena, 2001, p.22-29). In the first approach, local semantics are used rather than referring to common linguistic variables and each rule in the approximate approach defines its fuzzy sets while the descriptive approach "uses linguistic labels to refer to a common set of membership functions defined in the data base" (Cordón, Herrera, Hoffmann and Magdalena, 2001, p.114). As a result of this, the rules in the linguistic model normally are derived from experts and might have some restrictions to preserve the interpretability of the rules. Some authors refers to the approximate model as fuzzy logic systems with independent membership functions as opposed to the conventional model (Dadone, 2001). An example of adapting (learning or tuning) a type-1 fuzzy set defined by a Gaussian membership function is depicted in figure 2.8

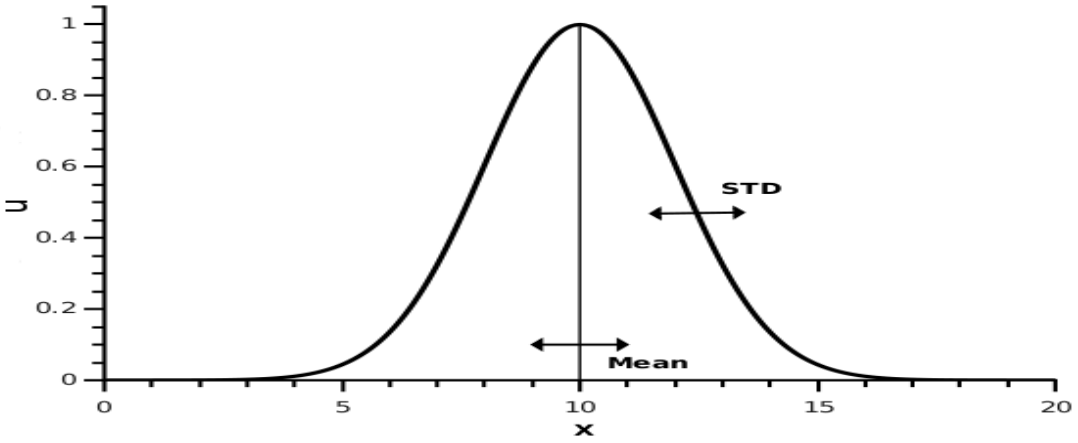

FIGURE 2.8: Tuning type-1 Gaussian membership function.

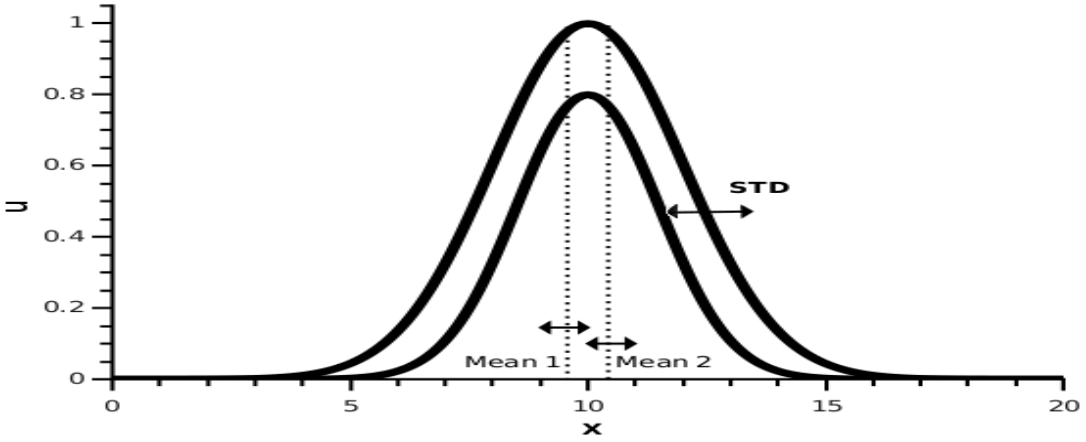

FIGURE 2.9: Tuning interval type-2 Gaussian membership function.

where the parameters of the fuzzy set are the mean and standard deviation while an example adapting an interval type-2 set is shown in figure 2.9 where the membership function is described by two means and one standard deviation.

2.4.2 Methods used for learning of type-1 fuzzy logic systems

Many hybrid approaches of fuzzy systems have been proposed and researched in the framework of soft computing. These approaches have been proposed because of the lack of learning capabilities of the fuzzy systems (Alcala et al., 1999; Cordon et al., 2004). Fuzzy systems are good at explaining how they reached a decision but can not automatically acquire the rules or membership functions to make a decision (Goonatilake and Khebbal, 1995, p.2). On the other hand, learning methods such as neural networks can not explain how a decision was reached but have a good learning capability while hybridisation overcomes the limitations of each method in one approach such as neuro-fuzzy systems or genetic fuzzy systems (Goonatilake and Khebbal, 1995, p.2). Two of the most known approaches for adding learning capability to fuzzy systems are genetic algorithms and neural networks. Genetic algorithms techniques are called genetic fuzzy systems when hybridised with fuzzy systems (Liska and Melsheimer, 1994; Hoffmann, 2001; Herrera, 2005; Kim and Kim, 2002; Cordon et al., 2004) while neural networks techniques are known as neuro-fuzzy systems (Jang and Sun, 1995; Horikawa et al., 1992; Jang et al., 1997; Shann and Fu, 1995). In fact, genetic algorithms are one of the most common search algorithms used with fuzzy systems that has been applied to a wide range of problems such as control system design, decision making, optimisation classification, modelling and information retrieval (Hoffmann, 2001). On the other hand, fewer researchers have studied the use of simulated annealing to learn fuzzy systems (Drack and Zadeh, 2006). Amongst those who studied this combination (Huyghe and Hamam, 1995) who applied Simulated Annealing to optimise fuzzy logic controllers, (Garibaldi and Ifeachor, 1999) applied simulated annealing to tune a medical fuzzy expert system, (Liu and Yang, 2000) presented a study of using simulated annealing for learning and tuning the membership functions, (Cordón et al., 2000) presented a method to obtain a uniform fuzzy partition granularity that improves the fuzzy system performance by using simulated annealing, (Drack and Zadeh, 2006) applied simulated annealing to two complex problems in aerospace design problems for aircraft propellers and manoeuvre

control of a satellite. (Guely et al., 1999) studied the use of simulated annealing to optimize the membership functions of Takagi-Sugeno membership functions and (Yanar and Akyrek, 2011) has used Wang and Mendel algorithm and fuzzy c-means clustering algorithm with simulated annealing to tune a linguistic Mamdani model for predicting Mackey-Glass time series. There are a wide range of other methods used to design fuzzy systems from other optimisation techniques to ad-hoc methods where the learning of fuzzy rules are guided by covering criteria of the data in the training set (Alcala et al., 1999). In 1992 and 1993, three articles appeared in the literature about designing fuzzy logic systems parameters by using the numerical training data rather than using fixed parameters chosen by the designer arbitrarily (Mendel, 2001, p.157). These works are Wang-Mendel algorithm (Wang and Mendel, 1992), a neuro-fuzzy system called ANFIS (Jang, 1993) and fuzzy neural networks with the back-propagation algorithm (Horikawa et al., 1992). All these works reported above were based on using type-1 fuzzy sets to build type-1 fuzzy logic systems. Many other methods used to learn fuzzy logic systems including local search algorithms such as gradient decent (Musikasuwan et al., 2004) and classical learning methods such as least-squares method (Mendel, 2001, p.162). The next section will present some attempts to design type-2 fuzzy logic systems.

2.4.3 Methods used for learning of type-2 fuzzy logic systems

The advancement on research in type-2 fuzzy sets and systems encouraged many researchers to apply some learning methods to type-2 fuzzy logic systems. For example:

- Type-2 fuzzy logic controllers for autonomous robots were evolved using genetic algorithms (Wagner and Hagras, 2007).
- Type-2 fuzzy logic controllers used for a coupled-tank liquid-level control system and evolved using genetic algorithms (Wu and Wan Tan, 2006).

- A type-2 fuzzy system was trained by particle swarm algorithm to estimate the blood pressure mean (Al-Jaafreh and Al-Jumaily, 2007).
- Type-2 fuzzy logic systems were designed using orthogonal least-squares and back-propagation (Méndez and de los Angeles Hernandez, 2009).
- Type-2 fuzzy logic systems were tuned using gradient descent to predict the Mackey-Glass time series with various levels of added noise (Musikasuwan et al., 2004).
- Type-2 fuzzy logic systems were designed using back-propagation method to predict the Mackey-Glass time series with added noise (Mendel, 2001, p.336).
- Type-2 fuzzy logic systems were tuned using genetic algorithms to design a trajectory tracking controller (Martinez et al., 2009).

There are many other research using some kind of learning or tuning to type-2 fuzzy logic systems reported in the literature (*e.g.* see (Mendel, 2007)). To the best of author knowledge, no work has been reported in the literature using simulated annealing to design type-2 fuzzy logic systems, despite that many other search algorithms have been used. (Miller et al., 2011) applied simulated annealing to model inventory management using supply chain model and interval type-2 fuzzy sets but no learning or optimisation has been carried out to these sets. In some other experiments, some manual tuning and settings of interval or general type-2 fuzzy logic systems were used as reported by (Coupland et al., 2006; Wagner and Hagras, 2009; Sepúlveda et al., 2007). It is indicated from above works and many others that interval type-2 fuzzy logic system can add more abilities to handle the uncertainties than type-1 fuzzy logic system. Although, the manual tuning of interval type-2 fuzzy logic systems can bring some success over type-1 fuzzy logic system where the third dimension is fixed, the automatic design of interval type-2 fuzzy logic systems can add extra fine tuning to model problems. Although, automated methods can add more fine tuning to expert designs, there is no rational

basis for choosing secondary membership grades (the third dimension) in general type-2 fuzzy sets (Mendel, 2001, p.302). This issue reinforces the need for automated methods in such designs which will be discussed in more details in Chapters 5 and 6.

Although, type-2 fuzzy logic is a growing research topic with much evidence of successful applications (John and Coupland, 2007), up to now, almost all developments type-2 fuzzy logic systems were based on interval type-2 fuzzy logic systems with some exceptions related to some works using different representations of type-2 sets and systems such as geometric type-2 fuzzy logic systems (Coupland and John, 2007), alpha-planes (Mendel et al., 2009), alpha cuts (Hamrawi et al., 2010) and Z-slices (Wagner and Hagras, 2010*a*; Christian Wagner, 2009). The ease of computation associated with the interval form of type-2 sets is the main driver for the wide usage of interval type-2 set and systems compared to the generalised form. One attempt to design general type-2 sets based on zSlices representation was proposed in (Christian Wagner, 2009) where survey data and device characteristics were used to build zSlices sets automatically. Another attempt to learn general type-2 fuzzy logic systems using alpha-plane representation has been introduced in (Mendel et al., 2009). Some other works using some neural networks concepts or classification algorithms such as: type 2 Adaptive Network Based Fuzzy Inference System (ANFIS) (John and Czarnecki, 1998), general type-2 fuzzy neural network (GT2FNN) (Jeng et al., 2009) and fuzzy C-means algorithm with a model known as "efficient triangular type-2 fuzzy logic system" (Starczewski, 2009*b*). To the best of the author's knowledge, no attempt to employ a learning method on general type-2 fuzzy logic systems using the vertical-slice representation has been made. The next section will introduce the simulated annealing algorithm.

2.5 Simulated Annealing Algorithm

The concept of annealing in the optimisation field was introduced by Kirkpatrick *et al* in (Kirkpatrick et al., 1983). The simulated annealing algorithm is a simple and

general algorithm for finding global minima by simulating behaviours of some physical processes (Salamon et al., 2002, p.6). It uses a randomised search method based on the Metropolis algorithm. With some restraints, many comparative studies for solving problems such as job shop scheduling and travelling salesman suggest that simulated annealing can outperform most other local search algorithms in terms of effectiveness and can find good solutions for a wide range of problems but normally with the cost of higher running costs (Aarts and Eikelder, 2002).

We now define the simulated annealing algorithm. Let s be the current state and $N(s)$ be a neighbourhood of s that includes alternative states. By selecting one state $s' \in N(s)$ and computing the difference between the cost of the current state and the cost of the selected state as $d = f(s') - f(s)$, s' is chosen as the current state based on *Metropolis criterion* in two cases:

- $d \leqslant 0$ means the new state has a smaller or equal cost, then s' is chosen as the current state as down-hill and equal moves are always accepted.
- $d > 0$ and the probability of accepting s' is larger than a random value Rnd such that $e^{-d/T} > Rnd$ then s' is chosen as the current state. T is a control parameter known as *Temperature* which is gradually decreased during the search process making the algorithm more greedy as the probability of accepting uphill moves decreases over time. Rnd is a randomly generated number where $0 < Rnd < 1$. Accepting uphill moves is important for the algorithm to avoid being stuck in a local minima.

In the third case where $d > 0$ and the probability is lower than the random value $e^{-d/T} \leqslant Rnd$, no moves are accepted and the current state s continues to be the current solution. When starting with a large cooling parameter, large deteriorations can be accepted. Then, as the temperature decreases, smaller deteriorations are accepted until the temperature approaches zero when no deteriorations are accepted. Therefore,

adequate temperature scheduling is important to optimise the search. Simulated annealing can be implemented to find the optimal solution by allowing infinite number of transitions or can be implemented to find a nearly optimal value within a finite time where the cooling schedule is specified by four components (Aarts and Eikelder, 2002):

1. Initial value of temperature.
2. A function to decrease temperature value gradually.
3. A final temperature value.
4. The length of each homogeneous Markov chains. A Markov chain is a sequence of trials where the probability of the trial outcome depends on the previous trial outcome only and called homogeneous when the transition probabilities do not depend on the trial number (Aarts and Lenstra, 2003, p.98). Homogeneous Markov chains are used to model the cooling schedule where the temperature is updated for each Markov chain. Therefore, at each Markov chain, a specific number of iterations is carried out.

The choice of good simulated annealing parameters is important for the success of simulated annealing. For example, small initial temperatures could cause the algorithm to get stuck in local minimas as the first stages of the search are supposed to aim for exploration of solution space while large ones could result in random search and excessive running times. In addition, an appropriate cooling schedule is important for the same reason as fast cooling causes getting stuck in local minimas and slow cooling make the algorithm very slow. Although, simulated annealing has been used in combinatorial optimisation where the search space is discrete, it can be used with continuous search spaces which require some form of discretisation of the search space (Ingber, 1993). In fuzzy systems, membership function parameters are continuous and the search space is discretised when optimising these parameters. Choosing the form of discretisation of these parameters constitutes the neighbourhood representation which

is another important issue when using simulated annealing. One of the criticisms of the simulated annealing approach is the difficulty of fine tuning the simulated annealing parameters (Ingber, 1993). For example, large step sizes allow for more exploration capabilities and help finding the optimal region but then the algorithm behaves badly and never reaches the peak of the optimum (Nolle et al., 2001). On the other hand, small step sizes are used to avoid this oscillatory behaviour in the final stages of the optimisation but this affects the convergence speed of the algorithm (Dadone, 2001). In addition, if the initial state is too far from the global optima, the algorithm might not reach the global optima before the temperature freezes the algorithm to the nearest local optima (Nolle et al., 2001). While there are some solutions to this problem by increasing the length of the Markov chains, this is impractical in some real-world applications with time constraints (Nolle et al., 2001). In the fuzzy system optimisation literature, few researchers have used adaptive step sizes such as (Jang, 1993) for type-1 fuzzy systems. The most of the approaches reported in the literature use small fixed step sizes (Dadone, 2001). In continuous optimisation problems, the adjustment of the neighbourhood range for simulated annealing might be important (Miki et al., 2002). The step sizes should not be all equal for all inputs rather they should be chosen based on its effects to the objective function (Locatelli, 2002). One of the methods used to determine the step size during the annealing process was proposed by (Nolle et al., 2001) which starts by using large step sizes and decrease them gradually. One of the methods to determine the initial temperature value proposed by (White, 1984) is to choose the initial temperature value within the standard deviation of the mean cost of a number of moves. When using finite Markov chains to model simulated annealing mathematically, the temperature is reduced once for each Markov chain while the length of each chain should be related to the size of the neighbourhood in the problem (Aarts and Eikelder, 2002).

2.6 Summary

This chapter introduced the existing literature on the methods and concepts discussed within the thesis. Four main methods and concepts are described. We first presented an introduction to the theory of fuzzy sets and fuzzy logic systems of type-1. Then, type-2 fuzzy logic systems were introduced as an extension to type-1 fuzzy logic systems. Both models of type-2 fuzzy sets, interval and general type-2 fuzzy sets and systems have been described with an overview of the last advances of research that tackled some of their issues. These issues include some attempts to reduce the computations needed for representation, operations and type-reduction of type-2 fuzzy sets. The issue of uncertainty handling between type-1 and type-2 fuzzy logic systems has been discussed. The third part highlighted some of the learning issues of fuzzy logic systems and how learning and optimisation methods can contribute to the design of type-1 and type-2 fuzzy logic systems. An overview of some of the well known learning methods of type-1 and type-2 fuzzy logic systems has been drawn. The final part explained the simulated annealing algorithm and some of its configuration issues. The next chapter will show the work carried out using simulated annealing and type-1 fuzzy logic systems.

Chapter 3

Learning Type-1 Fuzzy Logic Systems using Simulated Annealing

3.1 Introduction

This chapter presents the work for learning type-1 fuzzy logic systems using simulated annealing to forecast time-series. It will describe some issues related to this combination and shows how the combination is working. Many of these issues are needed for the next chapters as type-1 fuzzy logic system is very similar to type-2 fuzzy logic system in its structure and components. In addition, the experimentation settings and results for the use of simulated annealing with two fuzzy logic systems models (Mamdani and Takagi-Sugeno-Kang (TSK)) and under two fuzzification models (singleton and non-singleton fuzzifications) are detailed. These models are used to predict the well known Mackey-Glass time series with six different measurement noise levels representing different levels of uncertainties. The results of the proposed methods are compared by their ability to

handle uncertainty as well as comparing the noise-free time series with other works in the literature. Extra experiments and analysis on type-1 fuzzy logic systems will be presented in the third chapter for comparison and consistency.

3.2 Issues Related to The combination of Fuzzy Logic Systems and Simulated Annealing

This thesis compares across type-1, interval and general type-2 fuzzy logic systems with one optimisation algorithm which is simulated annealing. One of the motivations for using simulated annealing with fuzzy systems is that they do not require the existence of mathematical properties such as differentiability in the problem which allows the possibility of using all fuzzy structure components choices including non-differentiable t-norms and non-differentiable membership functions. Also, simulated annealing has the ability to avoid getting trapped in local optima and find global optima due to its mechanism of accepting higher-cost states with some probability. Although, the combination might have more complexity and longer search time than local search algorithms, it is likely to find the global or near global optima of the configuration of fuzzy logic systems using simulated annealing more than local search approaches. This is due to the ability of simulated annealing to avoid local optima by accepting some bad moves in order to explore the problem space. In addition, simulated annealing can suit high dimensionality problems as it scales well with the increase of variable numbers which allows simulated annealing to be a good candidate for fuzzy logic systems optimisation (Drack and Zadeh, 2006). Also, it is able to handle cost functions with different degrees of non-linearities, discontinuities, and stochasticity (Ingber, 1993). The problem of optimising membership functions of the fuzzy logic system in order to minimise the objective function is a complex problem due to the following (Guely et al., 1999):

- The object function is not derivable everywhere. For example due to the use of minimum t-norm or triangular membership functions.
- The object function is not continuous everywhere. For example when membership functions do not overlap.
- The number of parameters are large, increasing the dimensionality of the solution space. For example a fuzzy logic system with 8 rules and 4 inputs and one output might have $80 - 160$ parameters when two to four parameters are used for each fuzzy set.

As a result of these issues of complexity, a global search method such as simulated annealing is desired for this problem. Among global optimisation algorithms, two main classes of algorithms are known (Oliveira et al., 2012, p.22). The first is the single-point optimisation algorithms where the algorithm maintains one solution at a time. The other class is population-based optimisation algorithms where a population of individuals are maintained in an instantaneous time. The basic and well known form of simulated annealing is an example of the single-point optimisation algorithms while genetic algorithms and particle swarm optimisation are examples of the population-based optimisation algorithms. The first class has the property of describing a trajectory in the search space during the whole optimisation process (Oliveira et al., 2012, p.22). Although, the second class can exploit parallel computing, some forms of simulated annealing provide this ability using parallel simulated annealing techniques (Ram et al., 1996)(Onbaşoğlu and Özdamar, 2001). Unfortunately, due to the higher complexity in type-2 fuzzy logic systems especially when using general type-2 fuzzy sets, the use of parallel computing algorithms require more computations than single-point algorithms at each instantaneous time due to the need to compute the whole population rather than one individual. Therefore, single-point optimisation algorithms can suit some cases with on-line learning better than parallel computing. In addition, the use of single-point optimisation algorithms is considered as a wise choice in some cases when

tuning an existing linguistic expert fuzzy logic system. This is due to the need to preserve interpretability and the global structure of the experts fuzzy logic system which is difficult to maintain by population-based algorithms compared to the first class. To exploit parallel computing in other cases to reduce the whole experimentation time, parallel simulated annealing techniques can be used.

On the other hand, the simulated annealing convergence normally requires an exponential time which causes the algorithm to be impractical in some cases (Aarts and Lenstra, 2003, p.14). Also, the difficulty of determining suitable simulated annealing parameters such as a temperature scheduling and a good representation of the problem neighbourhood could cause the algorithm to yield undesired performance. One of the criticisms of simulated annealing is the difficulty to fine tune simulated annealing parameters and therefore quite time-consuming for developers to find an optimal fit (Ingber, 1993). The formalisations and configurations for simulated annealing to design fuzzy logic systems can be chosen from a large number of choices proposed on the literature. Here, we show some of these configurations when designing fuzzy logic systems parameters as stated in these points:

3.2.1 Neighbourhood representation

The parameters of the optimised fuzzy logic system when optimising its data-base (DB) are the parameters of the fuzzy sets included within rules. An example of these parameters are the mean and standard deviation of a Gaussian fuzzy set. Therefore, the neighbourhood representation is defined by moving one or more of these parameters to one or more directions by a defined move class. For example, by adding a defined step size to one or more of the current parameter's values to a specific direction. When doing so, the outputs of the fuzzy logic system might be changed which then affects the objective function value. However, the change depends on many factors including input variables, fuzzification methods, the number of rules, other rules firing levels, inference

engine operators and the defuzzification method. These factors can affect outputs in which the output might not change and both states have the same objective function value. The use of move constraints in neighbourhood representation including left and right bounds of membership functions can be helpful in cases where some linguistic properties such as interpretation have to be preserved. (De Oliveira, 1999) presented a methodology to achieve this objective and proposed some of these constraints. However, in cases where no predefined linguistic rules exist, the so called approximate fuzzy logic system, the use of these constraints might be in less need.

As stated in the previous chapter, the move class defined using a step size has an important role in the simulated annealing search. In fuzzy logic system optimisation literature, few researchers have used adaptive step sizes such as (Jang, 1993) for type-1 fuzzy systems. The most of the approaches reported in the literature were using small fixed step sizes (Dadone, 2001). The step sizes should not be all equal for all inputs rather it should be chosen based on its effects on the objective function (Locatelli, 2002). One way to apply this in data-driven fuzzy logic systems is to have a step size proportional to each input space in which all its fuzzy sets having a step size which might be different from the other inputs of the fuzzy logic system. The same issue is applied to the output fuzzy sets. Some of the issues related to step sizes of simulated annealing in continuous space have been discussed in (Locatelli, 2000) and (Miki et al., 2002).

3.2.2 Initial solution

Simulated annealing requires an initial solution to start with. The initial solution should be different each time simulated annealing is applied to the same fuzzy logic system to allow more explorations to take place especially when the first configurations of simulated annealing are dependent on the initial solution such as the initial temperature.

However, in general, the settings of simulated annealing should allow exploring the solution space in the first stages of the search. This objective can be achieved by allowing enough search before getting to small temperature values where no wide exploration is allowed. Some researchers used some quick algorithms to build the initial fuzzy logic systems before applying simulated annealing search. For example, (Yanar and Akyrek, 2011) has used the Wang and Mendel algorithm and fuzzy c-means clustering algorithm to get initial fuzzy logic systems before applying simulated annealing to tune a linguistic Mamdani model for predicting Mackey-Glass time series. However, these techniques normally can bring local minima solutions requiring simulated annealing initial configurations to account for this issue. Another way of getting the initial input membership functions for an approximate fuzzy logic system is to divide the input space for each input variable into partitions and allow enough overlapping between them (Guely et al., 1999). The initial output membership functions can be chosen randomly (Guely et al., 1999) (Mendel, 2001, p.171).

3.2.3 Objective function

In general, any objective function represents how the fuzzy logic systems' outputs are close to the optimal outputs can be used. In some cases, an error function is chosen as an objective function. Error functions are calculated from the outputs of the fuzzy logic systems compared to the optimal desired outputs (targets) using statistical metrics such as; MSE (mean square errors), $RMSE$ (root mean square error) and $SMAPLE$ (symmetric mean absolute percentage error). $RMSE$ is desired in some cases because it is measured in the same scale as the data (Hyndman and Koehler, 2006). The $RMSE$ as the objective function is defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i}^{n}(f(x) - \widehat{f}(x))^2} \tag{3.1}$$

Where n is the number of data samples in the observed data set, $f(x)$ is the output of the trained fuzzy logic system and $\widehat{f}(x)$ is the target output that the trained system aims to approach.

3.2.4 Initial Temperature

Small initial temperatures could cause the algorithm to get stuck in a local minima as the first stages of the search is supposed to aim for exploration of solution space while large ones could result in random search and excessive running times when accompanied with a slow cooling process. One of the methods to determine the initial temperature value proposed in (White, 1984) is to choose the initial temperature value within the standard deviation of the mean cost of a number of moves. This is achieved by moving to another state and evaluating its cost and returning to the initial state for a number of times. Then, the standard deviation of these states costs is calculated and assigned to the initial temperature.

3.2.5 Cooling schedule and cooling rates

An appropriate cooling schedule is important for the the success of simulated annealing as fast cooling causes getting stuck in a local minima and slow cooling make the convergence very slow. Having a suitable static or dynamic cooling rate will help simulated annealing converge to a global minima and avoid getting stuck in a local minima. Normally, practitioners use a static cooling rate ω normally chosen between $0.8 - 0.99$ (Aarts and Eikelder, 2002) such that :

$$T_{i+1} = T_i * \omega. \tag{3.2}$$

Static cooling values close to 1 allow more exploration for the search space while smaller values allow more greedy search and might not explore the whole search space.

```
InitialState = InitialiseParameters()
BestState = InitialState
for 1 to MarkovChainsNumber do
  for 1 to MarkovChainLength do
    s' = MoveToNighbour(FuzzySysParameters)
    if cost(s') ≤ cost(BestState) then
      BestState = s'
    else
      if exp((cost(s') − cost(s))/Temperature) ≥ Rnd) then
        BestState = s'
      end if
    end if
  end for
  Temperature = Temperature ∗ CoolingRate
end for
```

FIGURE 3.1: pseudocode of simulated annealing algorithm to design fuzzy systems

3.2.6 Markov chains configurations

When using Markov chains to model simulated annealing iterations mathematically, the temperature is reduced once for each Markov chain. The length of each chain should be related to the size of the neighbourhood in the problem (Aarts and Eikelder, 2002). The neighbourhood size in our case is the number of all fuzzy logic systems parameters involved in the optimisation process multiplied by two directions.

3.2.7 Stopping criterion

Typical stopping criterion of simulated annealing includes, stopping at small objective function values, stopping at lower temperature values, stopping after enough number of iterations or Markov chains and stopping when the changes of energy in the objective function are sufficiently small. The choice of stopping criterion is difficult as the optimal objective function is unknown in many practices (Garibaldi and Ifeachor, 1999).

The pseudocode of simulated annealing algorithm configurations to design fuzzy logic systems is shown in Figure 3.1.

3.3 Experimental Data

3.3.1 Mackey-Glass time series

The Mackey-Glass Time Series is a chaotic time series proposed in (Mackey and Glass, 1977). It is obtained from this non-linear equation :

$$\frac{dx(t)}{dt} = \frac{a * x(t-\tau)}{1 + x^n(t-\tau)} - b * x(t)$$

Where a, b and n are constant real numbers, t is the current time and τ is the difference between the current time and the previous time $t - \tau$. To obtain the simulated data, the equation can be discretised using the Fourth-Order Runge-Kutta method. In the case where $\tau > 17$, it is known to exhibit chaos and has become one of the benchmark problems in soft computing (Mendel, 2001, p.116).

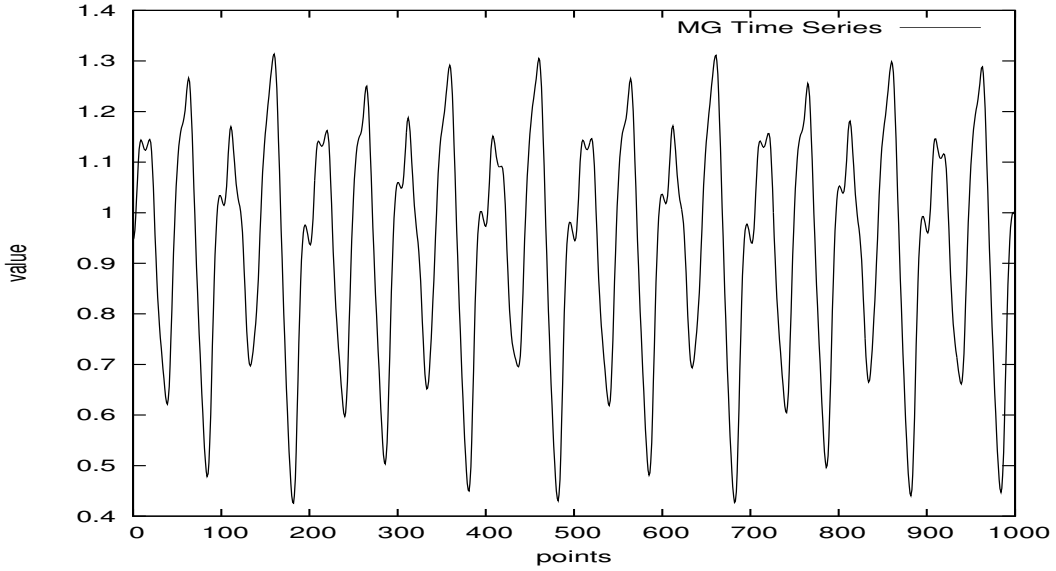

FIGURE 3.2: Mackey-Glass time series points values

3.3.2 Mackey-Glass time series with added noise

Adding some noise to the time series produces more challenges to the prediction job. In this experiment, noisy time series will be used to test our models. The noisy Mackey-Glass time series will be generated by adding noise to Mackey-Glass time series that are generated as described above. The amount of noise will be in different levels added to all inputs. The noise is measured by signal-to-noise ratio (SNR).

3.4 Methodology

The experiment can be divided into four steps; generating time series, adding noise to the time series, constructing the initial fuzzy logic system and learning the fuzzy logic system parameters. The experiment is more illustrated by the flowchart in figure 3.3.

3.4.1 Generating data sets

- The noise-free time series is generated with the following parameters : $a = 0.2$, $b = 0.1$, $\tau = 17$ and $n = 10$. The Runge-Kutta method is used to obtain the values of $x(t)$ at each time point with a time step of 0.1 and the initial condition $x(0) = 1.2$ where $x(t) = 0$ for $t < 0$. For comparative purposes, these settings for generating the time series are the same as (Jang, 1993). The input-output samples are extracted in the form $x(t-18), x(t-12), x(t-6)$ and $x(t)$ where $t = 118$ to $t = 1117$ using a step size of 6. A sample of the generated noise-free data are depicted in figure 3.2. Using a step size of 6, the input values to the fuzzy system are the previous data points $x(t-18), x(t-12), x(t-6)$ and $x(t)$ while the output from the fuzzy system is the predicted value $x(t+6)$. Four initial input values $x(114)$ and $x(115)$ and $x(116)$ and $x(117)$ are used to predict the first four training outputs.

- The noisy time series is generated using the same procedure above followed by adding different levels of measurement noise. This noise is added to each of the four inputs and no noise added to the outputs. The noise is measured by signal-to-noise ratio (SNR) which is used to measure the amount of noise compared to the original data. Therefore, the smaller SNR means larger noise. The different levels of noise are :

  level 1 : SNR=0 db (decibel).

  level 2 : SNR=10 db.

  level 3 : SNR=20 db.

  level 4 : SNR=30 db.

  level 5 : SNR=40 db.

  level 6 : noise-free.

Samples of the generated noisy data are depicted in figures 3.4,3.5,3.6,3.7 and 3.8 where a sample of the free-noise data is depicted in figure 3.2. Then the generated data are divided into 500 data points for training and the remaining 500 data points for testing.

3.4.2 Designing and learning of fuzzy logic systems

The fuzzy logic system has four-inputs and one-output. Both Mamdani and first-order Takagi-Sugeno-Kang (TSK) models consist of four dependent input fuzzy sets A_1, A_2, A_3 and A_4. Unlike TSK model, Mamdani model has one independent output fuzzy set B_l for each rule. Gaussian membership functions were chosen to define the fuzzy sets. Any other types of membership functions can be chosen but we are interested in reducing the rule-base complexity and computations time as Gaussian type has only two parameters instead of three in triangular type or four as the case in trapezoidal type. The parameters of the Gaussian membership functions are the mean m and the

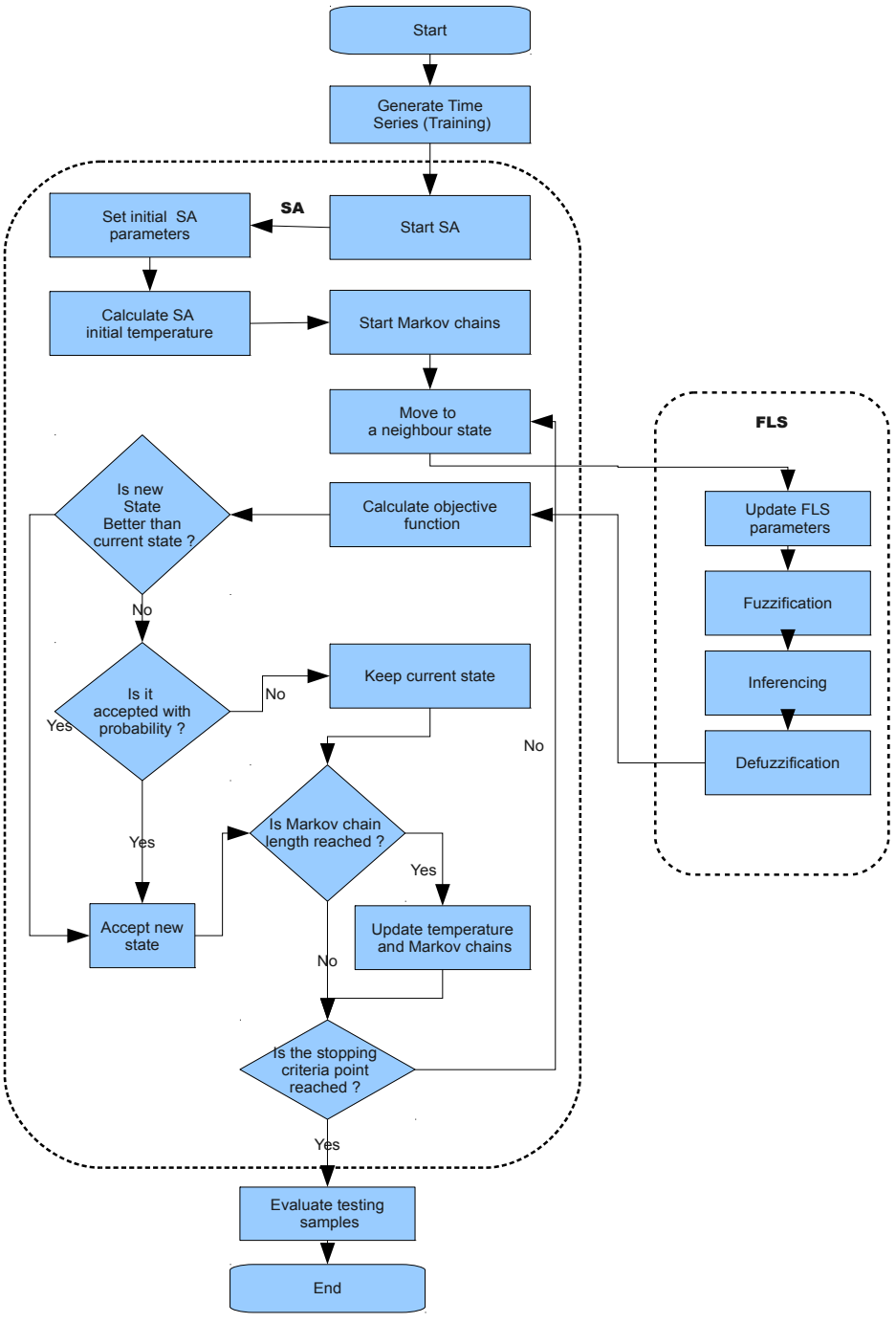

FIGURE 3.3: A flowchart of the method of using simulated annealing with fuzzy logic system in modelling applications

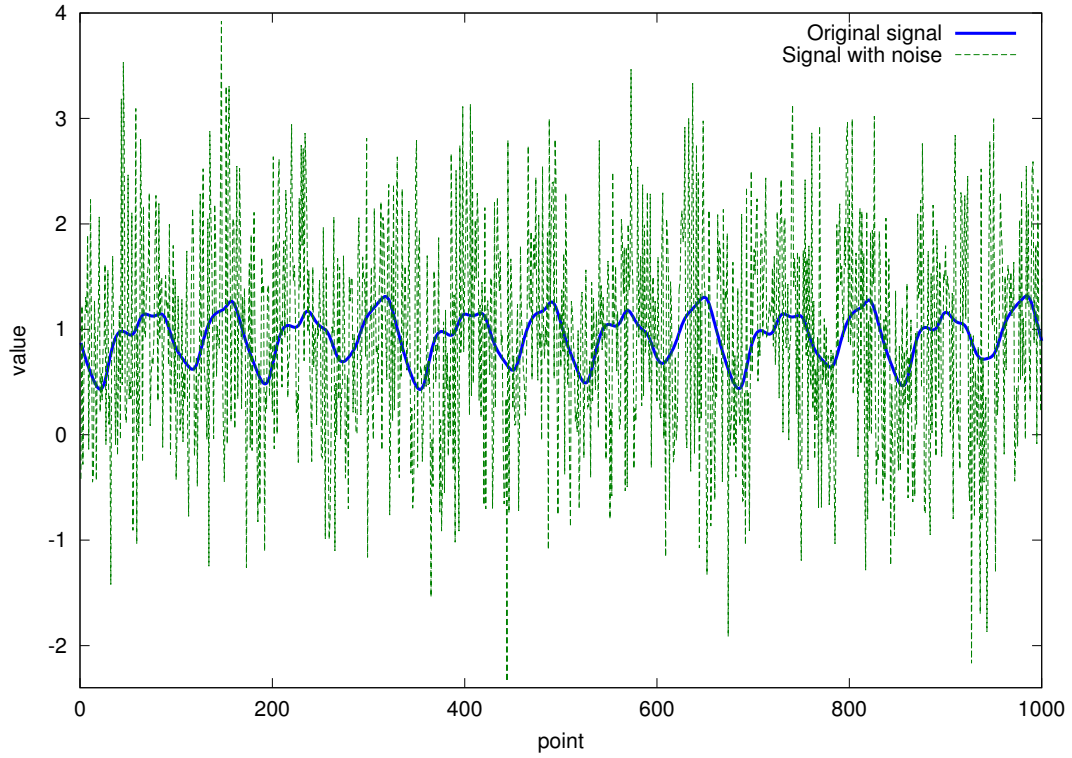

FIGURE 3.4: The first input of the time series when SNR=0 (Level 1)

standard deviation σ which is defined as follows:

$$\tilde{f}(x) = \exp^{-(\frac{x-m}{2\sigma})^2} \tag{3.3}$$

All the means and standard deviations are initially set for all the input fuzzy sets by dividing the input space into four fuzzy sets and enabling enough overlapping between them. The fuzzification process is based on the minimum t-norm. In the singleton fuzzification, the first operation is the fuzzification followed by the implication where both using minimum t-norm. The non-singleton fuzzification needs a pre-filtering process using a variability fuzzy set to model the input measurements for each input (Mendel, 2001, p. 188). The variability set is defined using a Gaussian membership function with a standard deviation (spread) proportional to the noise level as suggested by (Mendel, 2001, p. 188). The standard deviation for the variability set for each input is equal to

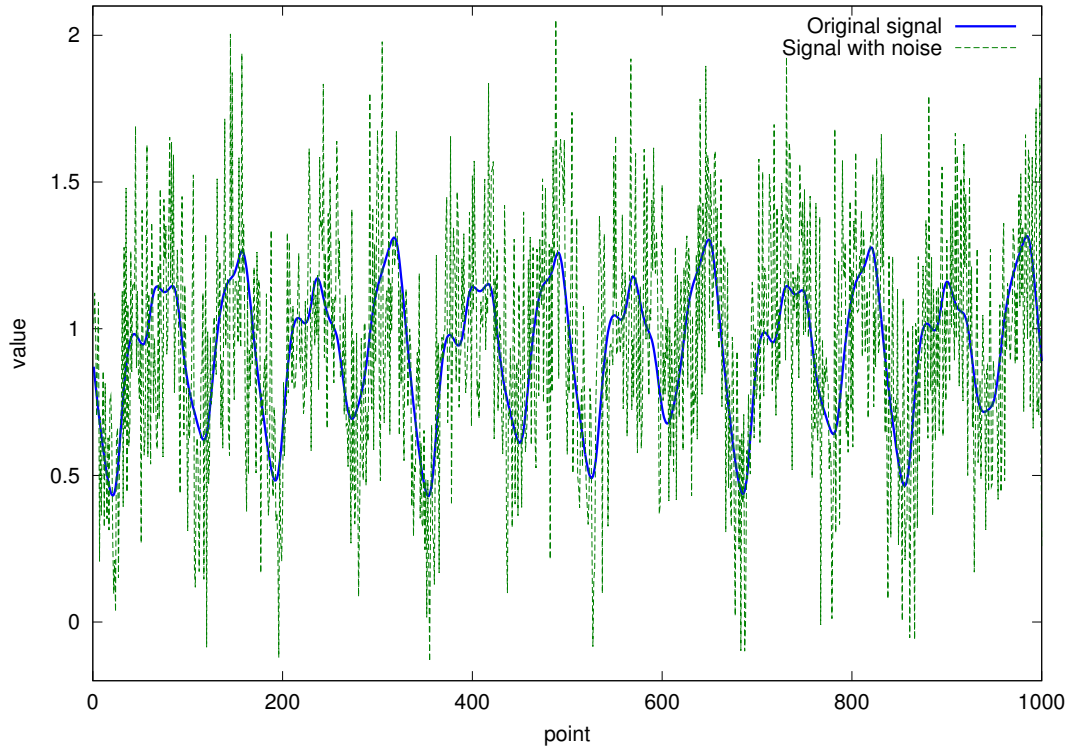

FIGURE 3.5: The first input of the time series when SNR=10 (Level 2)

the standard deviation for all noisy input data related to that input and stays fixed during the run. The minimum t-norm is chosen for pre-filtering, non-singleton fuzzification and implication. The training procedure aims to learn the parameters of the antecedent parts and the consequent parts of the fuzzy system rules. Then, the found parameters are used to predict the next 500 testing data points. By using four inputs and two fuzzy sets for each input, we end up with 16 rules and 8 input fuzzy sets representing all possible combinations of input values with input fuzzy sets. In Mamdani, each rule is linked with 1 output set while TSK model has 5 coefficients c_0, c_1, c_2, c_3 and c_4 in each rule. Therefore, there are 8 means and 8 standard deviations in the antecedent part linked with all these rules as well as one mean and one standard deviation for Mamdani model. The total number of optimised parameters in Mamdani model is $8 + 8 + (16 * 2) = 48$ while in TSK is $8 + 8 + (5 * 16) = 96$. The objective is to find the best set of parameters for all the rules.

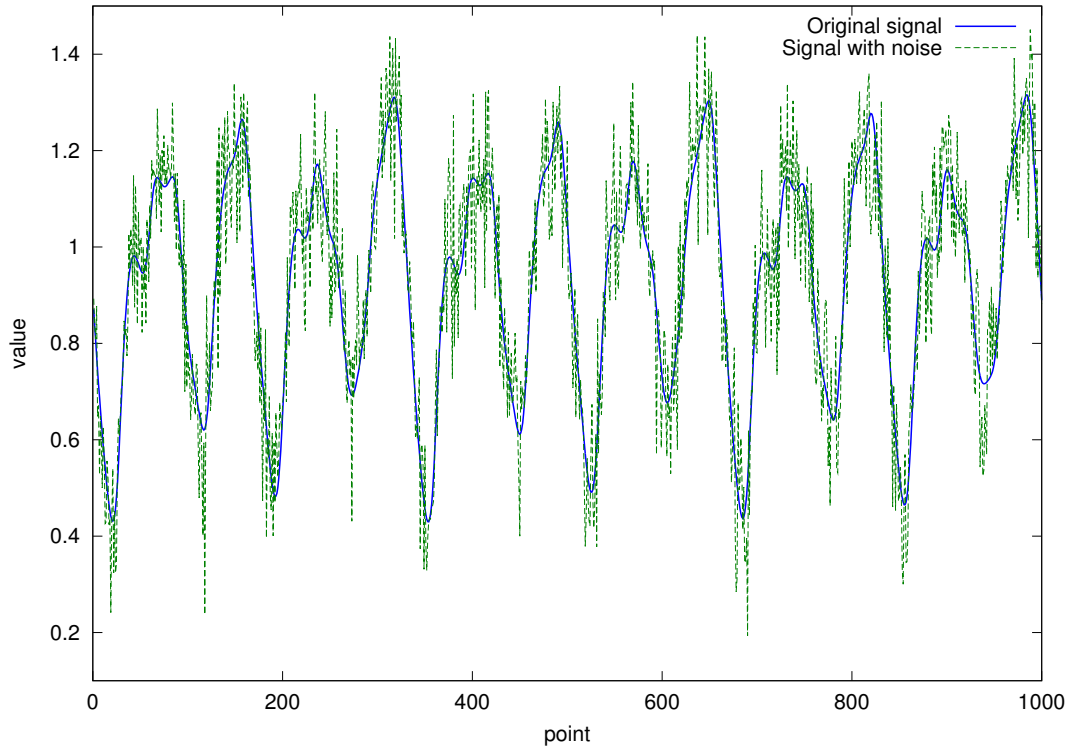

FIGURE 3.6: The first input of the time series when SNR=20 (Level 3)

The optimisation process is done using simulated annealing that searches for the best configuration of the parameters by trying to modify one parameter each time and evaluate the cost of the new state which is measured by Root Mean Square Error (RMSE). The simulated algorithm is initialised with a temperature that equals to the standard deviation of mean of RMSE's for 100 runs for the 500 training points. The cooling schedule is based on a cooling rate of 0.999 updated for each Markov chain which allows a slow cooling process. Each Markov chain has a length related to the number of variables in the search space. The search ends after 3 hours of time or if the changes of the energy in the objective function are sufficiently small meaning that no improvements appeared for a large number of iterations. The neighbouring states for a current state are chosen randomly by :

- TSK : Adding a small number 0.002 to one of the 80 coefficients (consequent

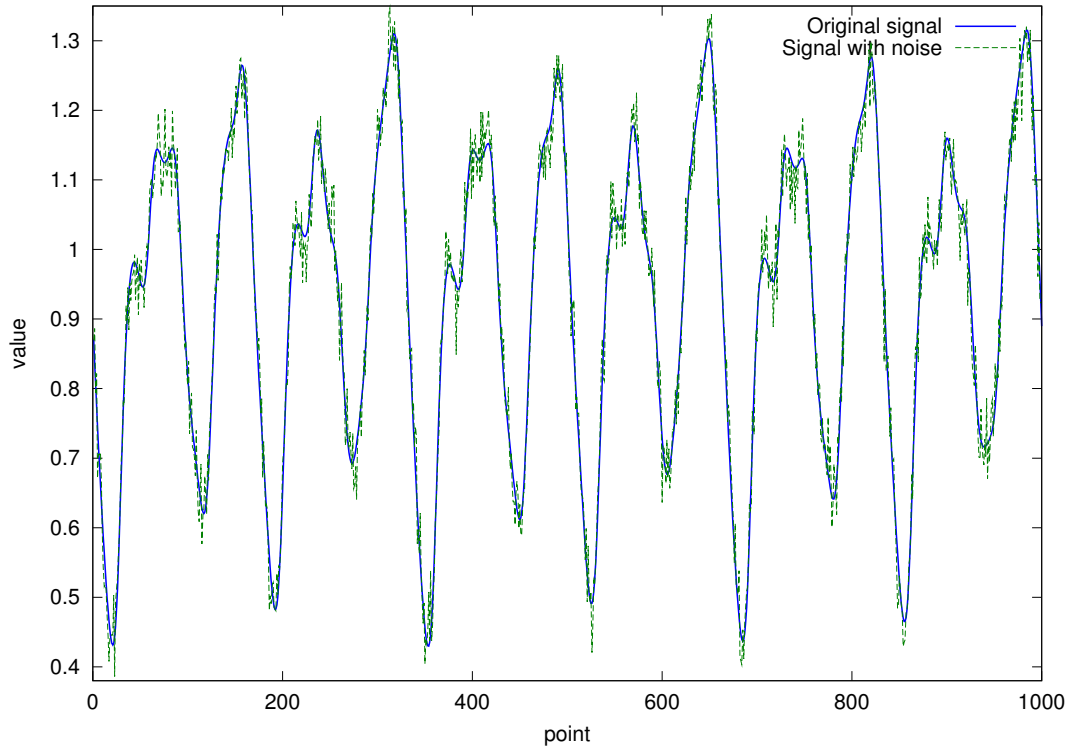

FIGURE 3.7: The first input of the time series when SNR=30 (Level 4)

part parameters) in the current state or adding a small number to one of the 16 antecedent parameters. This value is related to the difference between maximum and minimum values in the input data and = max-min/300.

- Mamdani : Adding a small number to one of the 16 antecedent parameters or the 32 consequent parameters. This value is related to the maximum and minimum value in the data and = max-min/300.

Then, the new state is evaluated by examining the 500 data points outputs. The experiment has been carried out 24 times for Mamdani model and 24 times for TSK model. Among them 24 times for each fuzzification method (singleton and non-singleton). The average RMSE's for the testing samples have been calculated.

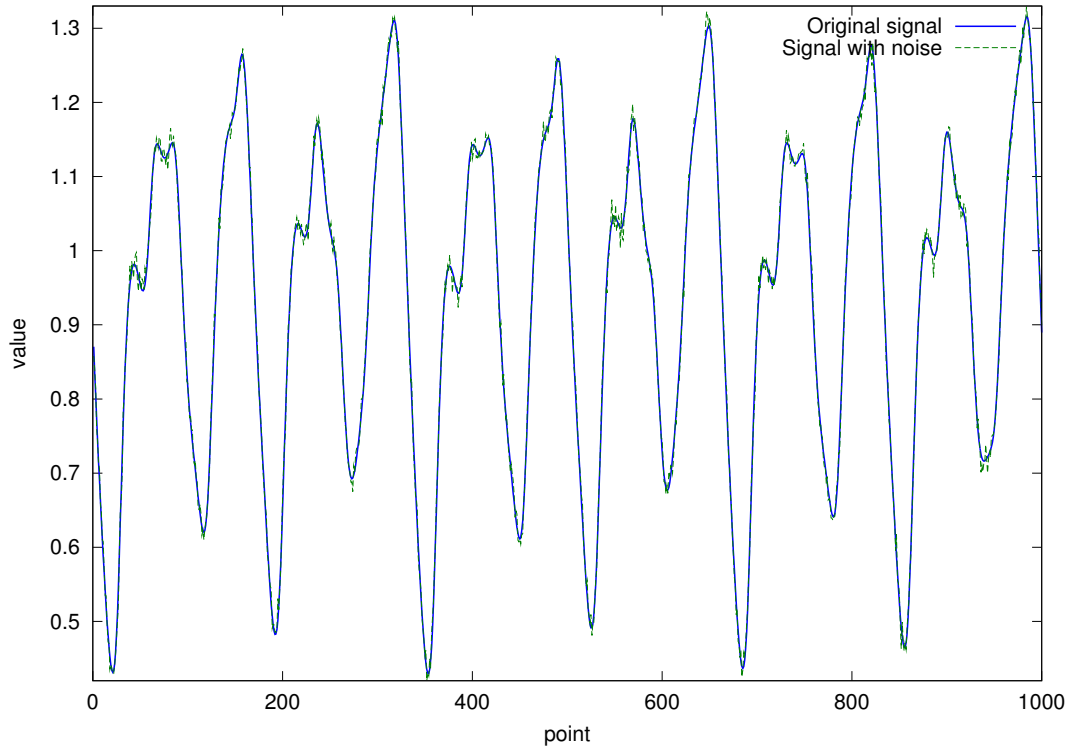

FIGURE 3.8: The first input of the time series when SNR=40 (Level 5)

3.5 Results and Discussion

It is shown from the results in Table 3.1 and Figure 3.9 that non-singleton systems handled the noisy data better than the singleton systems in the majority of cases except one case in the third level when singleton Mamdani was better than TSK non-singleton while the worse results for noisy cases were obtained by singleton systems in the majority of cases. On the other hand, in the absence of noise, the singleton system performs better than the non-singleton system. This might be explained by the fact that the non-singleton models still account for the input measurement uncertainty which does not exist in this case. Therefore, the best way to handle such situations by the non-singleton models is to change the standard deviation of the variability fuzzy set to zero which exactly converts it to a singleton system. These findings agree with some previous findings about the ability of non-singleton type-1 systems to handle uncertainty better

than the singleton systems in uncertain inputs situations (Mouzouris and Mendel, 1994) (Mendel, 2001, p. 186) and (Musikasuwan et al., 2004). Also, it is observed that the Mamdani non-singleton performance outperforms TSK in the higher noise levels from 1 to 4 while TSK performs well as the noise reduced especially for singleton systems. However, the non-singleton models require more computations than singleton models due to the extra operation in the fuzzification stage. It is important to mention that these experiments are based on one random generation of the added noise as another generation will produce different data which makes it difficult to compare the results of the prediction by simulated annealing and fuzzy systems with other methods for noisy Mackey-Glass time series (levels 1-5). In level 6 where no noise has been added, we can compare our results with some of the literature results. To show how good the combination, the comparison between our models results with others in Table 3.2 shows that our result of (RMSE)$= 0.009 - 0.0138$ have achieved good results compared to the best results which were obtained by ANFIS,GEFREX and Kukolj despite the general structure that our method has using a simple combination of a general search algorithm and a fuzzy system compared to the more complicated structures for ANFIS, Kukolj and GEFREX methods. Note that the fuzzy system model and structure has a notable impact on the performance of the fuzzy system and normally it is chosen heuristically. The comparison provided here is to show how good simulated annealing can be with fuzzy logic systems but not valid for an accurate comparison due to the differences between the the fuzzy system models and structures.

3.6 Summary

In this chapter, the work of learning type-1 fuzzy logic systems using simulated annealing has been presented and applied to a forecasting problem. The chapter discussed some issues related to this combination in more details. These issues include; the rationale for choosing this combination, the problem of learning fuzzy logic systems from an

TABLE 3.1: The forecasting results for Mackey-Glass time series with different levels of added noise

Fuzzy Model	Fuzzification	Level	SNR	RMSE average
Mamdani	non-singleton	1	0	0.2039
TSK	non-singleton	1	0	0.2081
Mamdani	singleton	1	0	0.2104
TSK	singleton	1	0	0.3273
Mamdani	non-singleton	2	10	0.1334
TSK	non-singleton	2	10	0.1387
Mamdani	singleton	2	10	0.1394
TSK	singleton	2	10	0.1549
Mamdani	non-singleton	3	20	0.0665
Mamdani	singleton	3	20	0.0670
TSK	non-singleton	3	20	0.0694
TSK	singleton	3	20	0.0777
Mamdani	non-singleton	4	30	0.0342
TSK	non-singleton	4	30	0.0349
TSK	singleton	4	30	0.0352
Mamdani	singleton	4	30	0.0401
TSK	non-singleton	5	40	0.0157
Mamdani	non-singleton	5	40	0.0197
Mamdani	singleton	5	40	0.0326
TSK	singleton	5	40	0.0435
TSK	singleton	6	free	0.0090
Mamdani	singleton	6	free	0.0093
TSK	non-singleton	6	free	0.0104
Mamdani	non-singleton	6	free	0.0138

optimisation perspective, how the two methods are combined in practise and some of the configurations of simulated annealing to suit fuzzy logic systems optimisation. In addition, the experiments for the use of simulated annealing with two fuzzy logic systems models (Mamdani and Takagi-Sugeno-Kang (TSK)) and under two fuzzification models (singleton and non-singleton fuzzifications) are carried out followed by their results and analysis. Theses models are used to predict the well known Mackey-Glass time series with six different noise levels representing different levels of uncertainties. The results of the proposed methods are compared by their ability to handle uncertainty as well as comparing the noisy free time series with other works in the literature. The

TABLE 3.2: Results comparison for predicting Mackey-Glass time series

Method	RMSE
Wang and Mendel (Lin and Lin, 1997)	0.08
Lin and Lin/FALCON-ART (Lin and Lin, 1997)	0.04
Kim and Kim/ GA Ensemble (Kim and Kim, 2002)	0.026
Juang and Lin/SONFIN (Juang and Lin, 1998)	0.018
Lo and Yang/TSK model (Lo and Yang, 1999)	0.0161
Russo / GEFREX (GA + NN) (Russo, 2000)	0.0061
Kukolj / Fuzzy cluster + LS + WRLS (Kukolj, 2002)	0.0061
Jang / ANFIS (Jang, 1993)	0.0015
This Model (TSK-singleton)	0.0090
This Model (Mamdani-singleton)	0.0093
This Model (TSK-non-singleton)	0.0104
This Model (Mamdani-non-singleton)	0.0138

combination exhibited good performance compared to other methods on the literature. In addition, both Mamdani and TSK models have been compared in their ability to handle uncertainty in the form of uncertain inputs using singleton and non-singleton fuzzification. The results show the ability for non-singleton systems to handle noisy data better than the singleton systems in the majority of cases while the worse results for noisy data were obtained by singleton systems in the majority of cases. In this chapter, a description of the Mackey-Glass time series problems with and without noise that were applied in this chapter is presented which will be needed in next chapters. Extra experiments and analysis on type-1 fuzzy logic systems will be presented in the Chapter 4 with interval type-2 fuzzy logic systems for comparison and consistency purposes.

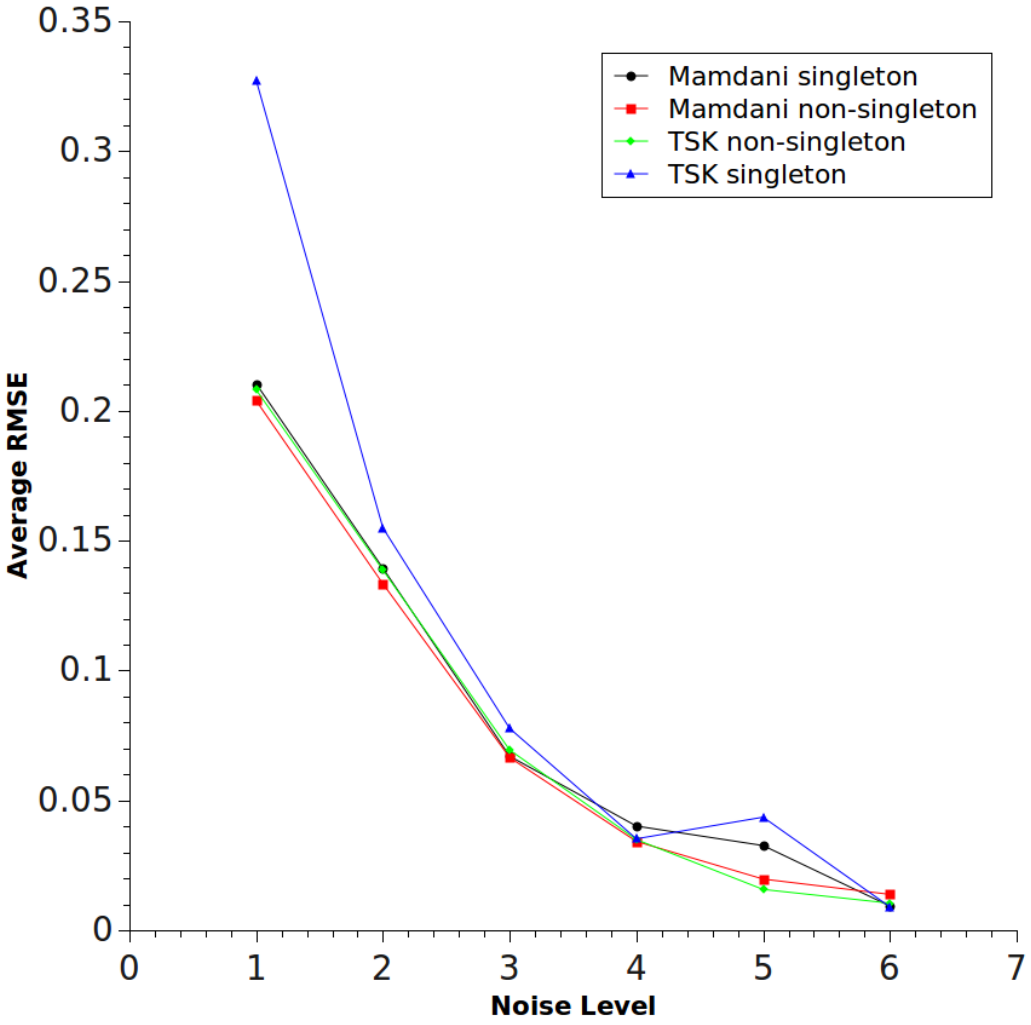

FIGURE 3.9: The forecasting results of simulated annealing with fuzzy logic systems for all noise levels.

Chapter 4

Learning Interval Type-2 Fuzzy Logic Systems Using Simulated Annealing

4.1 Introduction

This chapter will report the work of using simulated annealing and interval type-2 fuzzy logic system to provide more efficient systems to model uncertainty. In addition, both interval type-2 fuzzy logic system and type-1 fuzzy logic system are used to model four problems. These are noise-free and noisy Mackey-Glass time series forecasting (Mackey and Glass, 1977) and two real world problems which are the estimation of the low voltage electrical line length in rural towns and the estimation of the medium voltage electrical line maintenance cost (Cordón et al., 1999). Simulated annealing searches for the best configuration of the type-1 and interval type-2 fuzzy logic system parameters of the antecedent and the consequent parts of the rules for a Mamdani model. Detailed analysis and conclusions for the accuracy, speed and convergence of both models are

drawn. The results of using interval type-2 fuzzy logic systems are compared to the results of using type-1 fuzzy logic systems where it shows encouraging results for interval type-2 fuzzy logic systems. The chapter starts by explaining the differences when using simulated annealing with both fuzzy models followed by the experimental data and methodology. Then, the results are analysed and conclusions are drawn.

4.2 Issues Related to The Combination of Type-2 Fuzzy Logic Systems and Simulated Annealing

Interval type-2 fuzzy logic system is a rule based system that is similar to a type-1 fuzzy logic system in terms of its structure and components as described in Section 2.2.3. The only differences are that an interval type-2 fuzzy logic system uses type-2 fuzzy sets with its operations and has an extra output process component called the type-reducer before defuzzification. Therefore, the optimisation (learning) process is mainly the same as the one used with type-1 fuzzy logic systems. However, due to the extra parameters and computations needed when using interval type-2 fuzzy sets and systems, the time needed for the search process is normally longer. The problem of optimising the membership functions of interval type-2 fuzzy logic systems is more complex than type-1 fuzzy logic systems due to the extra parameters associated normally with its fuzzy sets. For instance, a triangular type-1 fuzzy set is defined using 3 parameters where it is defined using 5 or 6 parameters in interval type-2 fuzzy set. The extra parameters add extra dimensions to the solution space and require more searching process to find a good solution. The differences in the operations and the defuzzification stage do not affect the way that simulated annealing is used but they add more computations inside the evaluation process for the objective function every time the interval type-2 fuzzy logic system is evaluated. The other details for using simulated annealing with interval type-2 fuzzy logic systems are the same as those reported in Chapter 3 including how the whole combination is used to model problems. Therefore, this information is not

repeated here. The rest of the chapter will report the experimentation data, settings and results of using both systems to model four problems.

4.3 Methodology

In these experiments, simulated annealing will be used to design type-1 and interval type-2 fuzzy logic systems. The proposed method will be applied to four bench-mark problems. The experiment can be divided into three steps : preparing data, constructing the initial fuzzy system and optimising the fuzzy system parameters.

4.3.1 Experimental data

4.3.1.1 Mackey-Glass time series

The Mackey-Glass time series is a chaotic time series proposed in (Mackey and Glass, 1977). It has been described in section 3.3.1. To create the time series, firstly, the noise-free time series is generated with the following parameters : $a = 0.2$, $b = 0.1$, $\tau = 17$ and $n = 10$. The Runge-Kutta method is used to obtain the values of $x(t)$ at each time point with a time step of 0.1 and the initial condition $x(0) = 1.2$ where $x(t) = 0$ for $t < 0$. For comparative purposes, these settings for generating the time series are the same as other authors such as (Jang, 1993; Russo, 2000; Kim and Kim, 2002; Kukolj, 2002). The input-output samples are extracted in the form $x(t-18), x(t-12), x(t-6)$ and $x(t)$ where $t = 118$ to $t = 1117$ using a step size of 6. A sample of the generated noise-free data are depicted in figure 3.2. Then the generated data are divided into 500 data points for training and the remaining 500 data points for testing. Using a step size of 6, the input values to the fuzzy system are the previous data points $x(t-18), x(t-12), x(t-6)$ and $x(t)$ while the output from the fuzzy system is the predicted value $x(t+6)$. Four initial input values

$x(114)$ and $x(115)$ and $x(116)$ and $x(117)$ are used to predict the first four training outputs.

4.3.1.2 Mackey-Glass time series with added noise

Adding some noise to the time series produces more difficulties to model time-series. In this experiment, a noisy time series will be used to test our models. The noisy Mackey-Glass time series will be generated by adding noise to Mackey-Glass time series that are generated as described above. The amount of noise will be 20 *db* added to all inputs and outputs. The noise is measured by signal-to-noise ratio (SNR). A sample of the noisy data is depicted in figure 4.1.

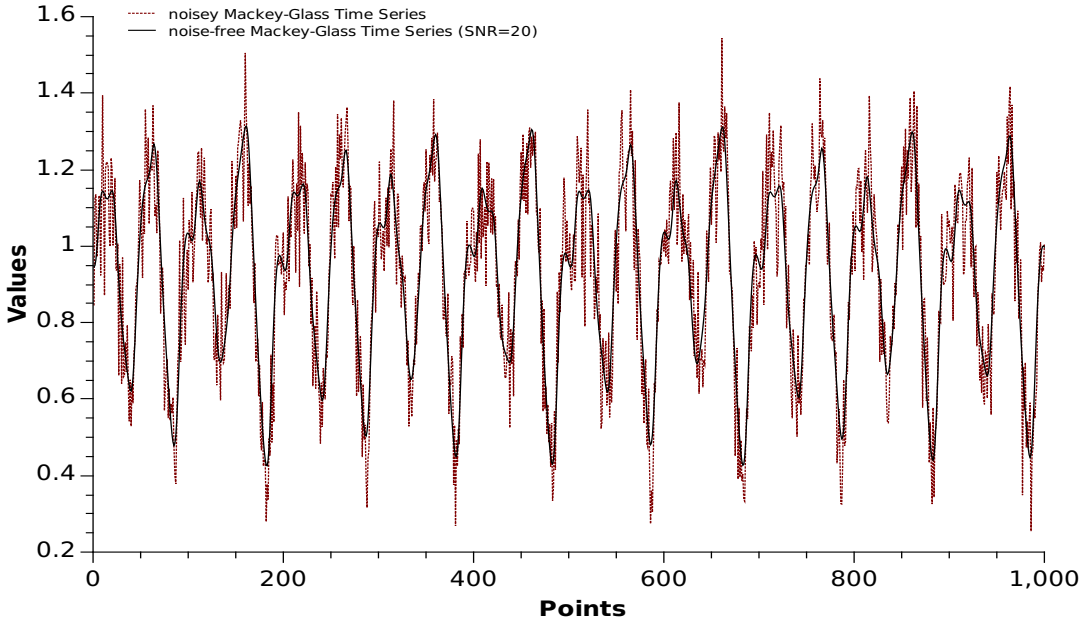

FIGURE 4.1: A sample Mackey-Glass time series when SNR=20 with the original signal.

4.3.1.3 Estimation of the low voltage electrical line length in rural towns

This problem and the next one are bench-mark real world problems in fuzzy logic community proposed in (Cordón et al., 1999). The first is concerned with finding a model that estimates the total length of low voltage line installed in a rural town using some available information. The data consists of 495 samples in which the real data was measured by a Spanish company. Each sample has two inputs which are the number of inhabitants in the town and the mean of the distances from the centre of the town to the three furthest clients in it while the output is the estimated length of low-voltage line. The data set has been taken from (Casillas, 2011). The data samples were randomly divided into two sets labelled training and testing sets which are randomly selected from the whole sample as reported in (Cordón, Herrera and Villar, 2001) and (Cordón et al., 2002). As with other authors, 396 samples are used for training while the other 99 samples are used for testing.

4.3.1.4 Estimation of the medium voltage electrical line maintenance cost

The aim of this application is to estimate the minimum maintenance costs of the medium voltage electrical line based on a model of the optimal electrical network for some Spanish towns (Cordón et al., 1999). The problem has four input variables: sum of the lengths of all streets in the town, total area of the town, area that is occupied by buildings, and energy supply to the town while the output is the minimum maintenance cost. The data set consists of 1056 samples and has been taken from (Casillas, 2011). The data samples were randomly divided into two sets labelled training and testing sets which are randomly selected from the whole sample as reported in (Cordón, Herrera and Villar, 2001) and (Cordón et al., 2002). As with other authors, 845 samples are used for training while the other 211 samples are used for testing.

4.3.2 The initial fuzzy logic systems

Two fuzzy systems have been chosen: a type-1 fuzzy logic system and an interval type-2 fuzzy logic system. The fuzzy model consists of a number of independent input fuzzy sets and one independent output fuzzy set for each rule. For the maintenance cost and Mackey-Glass problems, there are 16 rules while each rule is characterised by 5 fuzzy sets (4 antecedent fuzzy sets and one consequent fuzzy set) where it is 16 rules with 3 fuzzy sets for the low voltage problem. However, the number of rules was chosen heuristically and any number of rules can be chosen. The fuzzy sets of type-1 are described by Gaussian membership functions which is defined as :

$$\tilde{f}(x) = \exp^{-(\frac{x-m}{2\sigma})^2}$$

where the parameters of the Gaussian membership functions are the mean m and the standard deviation σ. Any other types of membership functions can be chosen but we are interested in reducing the number of parameters as Gaussian type has only two parameters instead of three in triangular type or four as the case in trapezoidal type. All the means and standard deviations for all the input fuzzy sets and the output fuzzy sets are initialised randomly. For a type-2 system, the system is built from scratch rather than using the optimised type-1 system to initialise the fuzzy sets. Using optimised type-1 fuzzy logic system might restrict the design to some local minimas when designing interval type-2 fuzzy logic system. In addition, each interval type-2 fuzzy set is described by Gaussian primary membership functions with uncertain means represented by two means and one standard deviation as follow (Mendel, 2001, p.91):

$$\tilde{f}(x) = \exp^{-(\frac{x-m}{2\sigma})^2} \; m \in [m_1, m_2]$$

Therefore the upper $\overline{\mu}_{\tilde{A}}(x)$ and lower $\underline{\mu}_{\tilde{A}}(x)$ membership functions are defined by following mathematical functions:

$$\overline{\mu}_{\tilde{A}}(x) = \begin{cases} \exp^{-(\frac{x-m_1}{2\sigma})^2} & \text{if } x < m_1 \\ 1 & \text{if } m_1 \leq x \leq m_2 \\ \exp^{-(\frac{x-m_2}{2\sigma})^2} & \text{if } x > m_2 \end{cases}$$

$$\underline{\mu}_{\tilde{A}}(x) = \begin{cases} \exp^{-(\frac{x-m_2}{2\sigma})^2} & \text{if } x \leq \frac{m_1+m_2}{2} \\ \exp^{-(\frac{x-m_1}{2\sigma})^2} & \text{if } x > \frac{m_1+m_2}{2} \end{cases}$$

All the means and standard deviations are initialised for all the input fuzzy sets by partitioning each input space into 16 dependent fuzzy sets and enabling enough overlapping between them while the output fuzzy sets are initialised randomly. The conjection of fuzzified values process is based on the product t-norm while the centre-of-sets has been chosen for defuzzification. Hence, for type-1 sets, this is the same as height defuzzification method because all sets are convex, symmetric and normal (Mendel, 2001, p.148). In type-2 defuzzification, the collapsing method proposed in (Greenfield, Chiclana, Coupland and John, 2009) has been used to calculate the centroids of the interval type-2 sets that needed to compute centre-of-sets. This is done by using the composite outward right-left variant of the collapsing method (Greenfield, Chiclana and John, 2009) as it is described in Chapter 2. The training procedure aims to optimise the parameters of the antecedent parts and the consequent parts of the fuzzy system rules. Then, the found parameters are used to predict the next testing data points. The standard deviations of the consequent sets are not included in the optimisation process as they have no effects on the output when using type-1 centre-of-sets defuzzification. The total number of optimised parameters for the low voltage problem when using type-1 FLS is $16 * 2 * 2 + 16 * 1 = 80$ and it is $16 * 2 * 3 + 16 * 2 = 128$ for interval type-2 fuzzy logic system. For the maintenance cost and Mackey-Glass problems, the number is $16 * 4 * 2 + 16 * 1 = 144$ and $16 * 4 * 3 + 16 * 2 = 224$ respectively. It is calculated as (number of rules * number of antecedent fuzzy sets in each rule * number

of parameteres in each antecedent fuzzy set) + (number of rules * number of consequent fuzzy sets in each rule(=1) * number of parameteres in each consequent fuzzy set).

4.3.3 The learning of the fuzzy logic systems

The optimisation process is done using simulated annealing that searches for the best configuration of the parameters by trying to modify one parameter each time and evaluate the cost of the new state which is measured by a cost function which is Root Mean Square Error (RMSE) defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{k=1}^{n}[f(k) - f(k^*)]^2}$$

The only constraint to the variables of the optimisation problem is that all standard deviations in Gaussian functions of all fuzzy sets must be ≥ 0. The simulated annealing algorithm is initialised with a temperature that equals to the standard deviation of mean of RMSE's for 1000 runs for the 500 training points as proposed in (White, 1984). The cooling schedule is based on a static cooling rate of 0.9 updated for each Markov chain. Each Markov chain has a long length related to the number of variables in the search space which equals to the power of the number of parameters. The search ends after a number of Markov chains namely 100 Markov chains. The new states are chosen from neighbouring states randomly by adding a small number (step size) to one of the antecedent parameters or the consequent parameters. The step size value is calculated using the maximum and minimum value for each input space and = max-min/200 while the direction of the search is chosen randomly. After that, the new state is evaluated by examining the RMSE of the 500 data points outputs. The experiment has been carried out 20 times and the average and the minimum of the cost function of the training and testing results have been calculated. Also, the way that SA is applied in these models tries to get a global or nearly global solution to unveil the potential for type-1 and

type-2 systems to capture the uncertainties without time constraints. This is done by using long Markov Chains with a slow cooling process where the acceptance ratios in the first Markov Chains are over 90%.

4.4 Results

4.4.1 Mackey-Glass time series results

The results of predicting noise-free Mackey-Glass time series by simulated annealing with type-1 fuzzy logic system (SA-T1FLS) and interval type-2 fuzzy logic system (SA-IT2FLS) are shown in Table 4.1 while the prediction accuracies and errors for the best runs are depicted in Fig 4.2 and 4.3. The results show good accuracies for both SA-T1FLS and SA-IT2FLS. However, Table 4.1 shows that SA-IT2FLS outperforms SA-T1FLS in both the average RMSE and the best minimum RMSE in the testing phase indicating that SA-IT2FLS was able to capture more information and uncertainties than SA-T1FLS with an improvement on the average RMSE about 43% from the SA-T1FLS average RMSE. In addition, the standard deviation for the 20 RMSE's in SA-IT2FLS was small compared to the one achieved by SA-T1FLS. Figure 4.4 shows the average error curves for both models during the training phase taken at the end of each iteration sequence (Markov Chain) where SA-IT2FLS shows stable smaller errors where the increase of iterations in SA-T1FLS does not help SA-T1FLS to outperform SA-IT2FLS. The smoothness in theses curves is a result of computing the average RMSE for each Markov Chain for all the 20 runs instead of drawing the best runs performances. On the other hand, the time taken to learn SA-IT2FLS was long compared to SA-T1FLS. This is expected due to the extra computations associated with T2FLS especially in the defuzzification phase.

TABLE 4.1: Forecasting results for noise-free Mackey-Glass time series by simulated annealing with T1FLS and T2FLS

Type-1 FLS			
Statistics	$Train_{RMSE}$	$Test_{RMSE}$	$Time_{Seconds}$
Average	0.007848	0.007040	1,218.9
Standard Deviation	0.004501	0.003431	9.09294
Minimum	0.003996	0.00396	1,198
Type-2 FLS			
Average	0.003978	0.0039889	18,274.3
Standard Deviation	0.001221	0.001429	261.62
Minimum	0.002864	0.002763	17,774

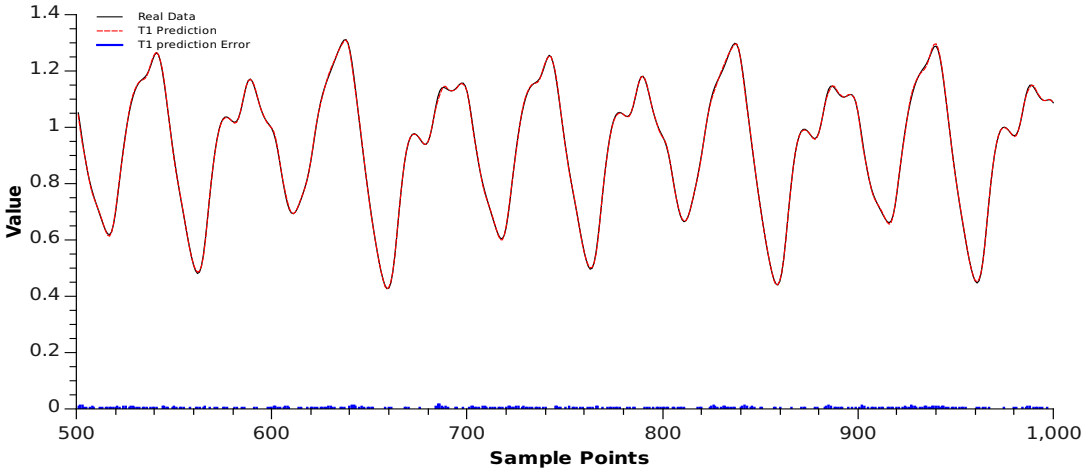

FIGURE 4.2: The prediction results and prediction error for noise-free Mackey-Glass time series problem by SA-T1FLS. The predicted data are difficult to distinguish from the target data which shows the accuracy of the method

4.4.2 Mackey-Glass time series with added noise results

The results of predicting noisy Mackey-Glass time series by SA-T1FLS and SA-IT2FLS are shown in Table 4.2 while the prediction accuracies and errors for the best runs are depicted in Fig 4.5 and 4.6. Table 4.2 shows that SA-IT2FLS yields smaller errors in both the average RMSE and the best (minimum) RMSE in the testing phase indicating that SA-IT2FLS was able to capture more information and uncertainties than SA-T1FLS with an improvement on the average RMSE about 2% from the SA-T1FLS average RMSE. In addition, the standard deviation for the 20 RMSE's in SA-IT2FLS

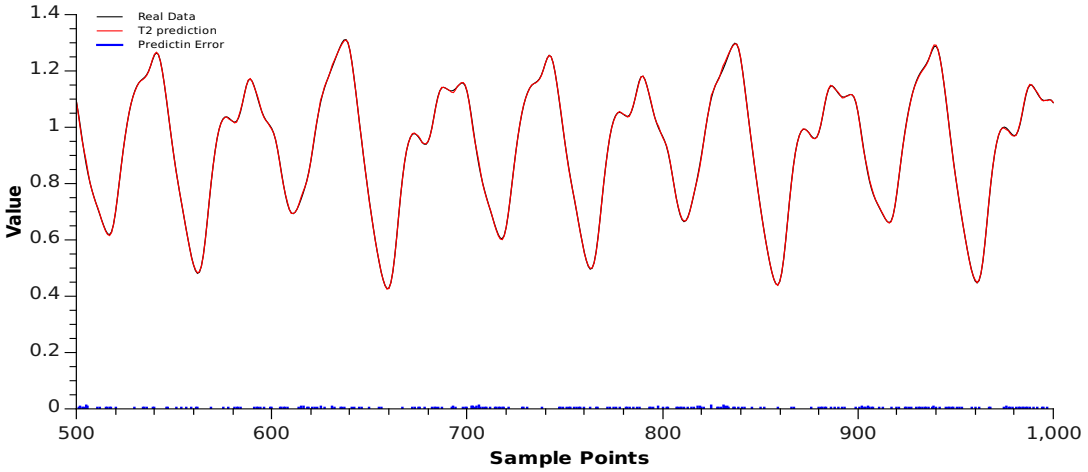

FIGURE 4.3: The prediction results and prediction error for noise-free Mackey-Glass time series problem by SA-IT2FLS. The predicted data are difficult to distinguish from the target data which shows the accuracy of the method

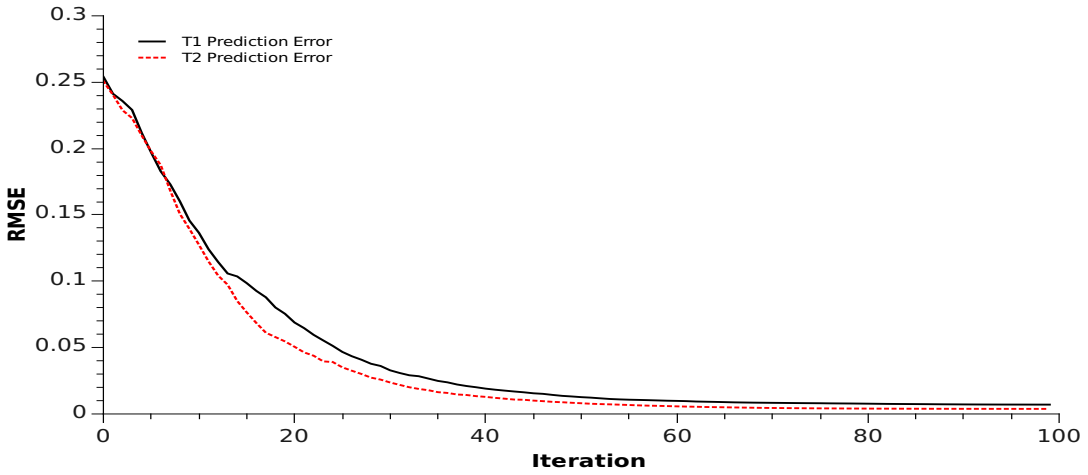

FIGURE 4.4: The average convergence of SA-T1FLS and T2FLS for noise-free Mackey-Glass time series problem.

was small compared to the one achieved by SA-T1FLS. Figure 4.7 shows the average error curves for both models during the training phase where SA-IT2FLS shows smaller errors in the majority of iterations. Again, the time taken to tune SA-IT2FLS was long compared to SA-T1FLS where this is expected due to the extra computations associated with T2FLS.

TABLE 4.2: Forecasting results for Mackey-Glass time series with added noise by simulated annealing with T1FLS and T2FLS

Type-1 FLS			
Statistics	$Train_{RMSE}$	$Test_{RMSE}$	$Time_{Seconds}$
Average	0.1082247	0.1590716	1,226.65
Standard Deviation	0.003445	0.014258	19.25
Minimum	0.098845	0.144805	1,191
Type-2 FLS			
Average	0.098651	0.155815	13,896
Standard Deviation	0.003392	0.009098	206.88
Minimum	0.0918268	0.142193	13,552

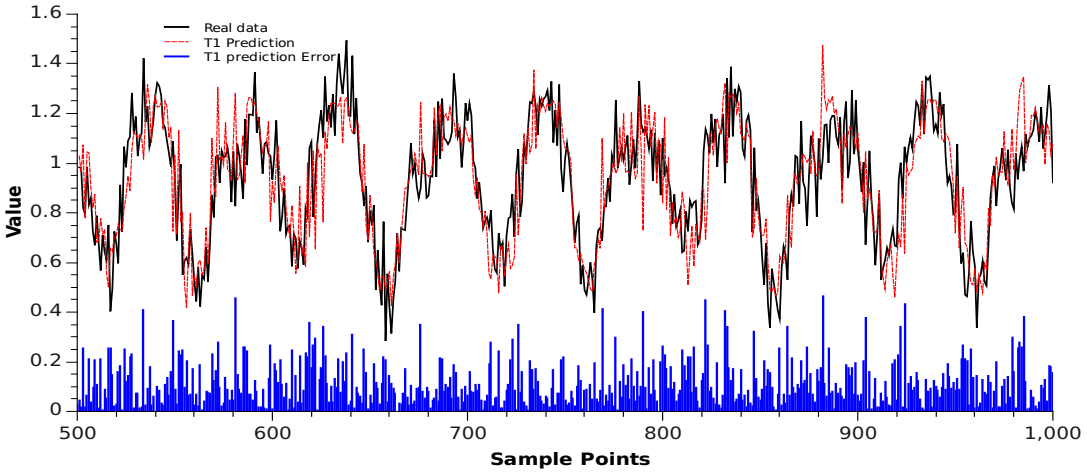

FIGURE 4.5: The prediction results and prediction errors for noisy Mackey-Glass time series problem by SA-T1FLS

4.4.3 Estimation of the low voltage electrical line length in rural towns results

The results of the estimation of lines length by SA-T1FLS and SA-IT2FLS are shown in Table 4.3 While the estimation accuracies and errors for the best runs are depicted in Fig 4.8 and 4.9. Table 4.3 shows that SA-IT2FLS obtained smaller errors in both the average RMSE and the best (minimum) RMSE in the testing phase indicating that SA-IT2FLS was able to capture more information and uncertainties than SA-T1FLS with an improvement on the average RMSE about 5% from the SA-T1FLS average

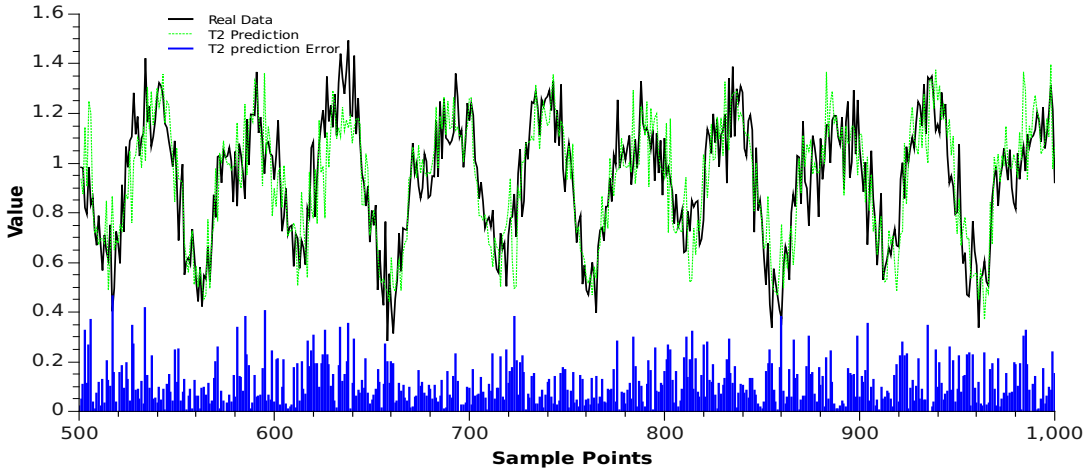

FIGURE 4.6: Prediction results and prediction errors for noisy Mackey-Glass time series problem by SA-IT2FLS.

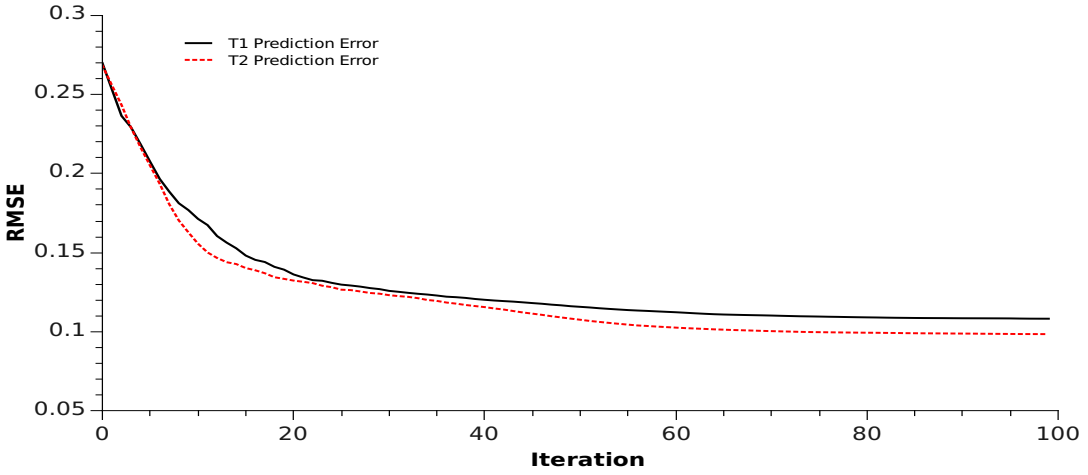

FIGURE 4.7: The average convergence of SA-T1FLS and T2FLS for noisy Mackey-Glass time series problem.

RMSE. In addition, the standard deviation for the 20 RMSE's in SA-IT2FLS was small compared to the one achieved by SA-T1FLS. Figure 4.10 shows the average error curves for both models during the training phase where SA-IT2FLS shows a stable smaller errors during training iterations. On the other hand, the time taken to tune SA-IT2FLS was long compared to SA-T1FLS due to the extra computations associated with T2FLS.

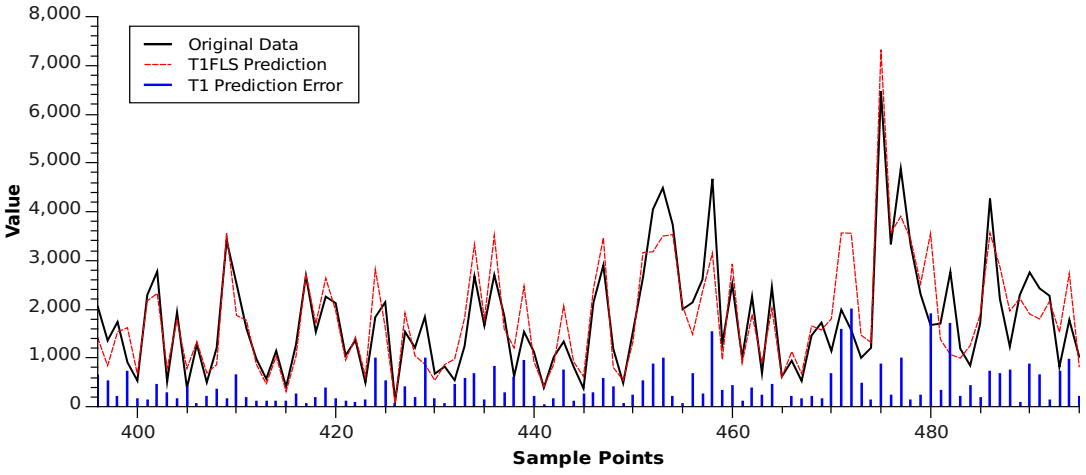

FIGURE 4.8: Estimation results and estimation errors for low voltage line problem by SA-T1FLS

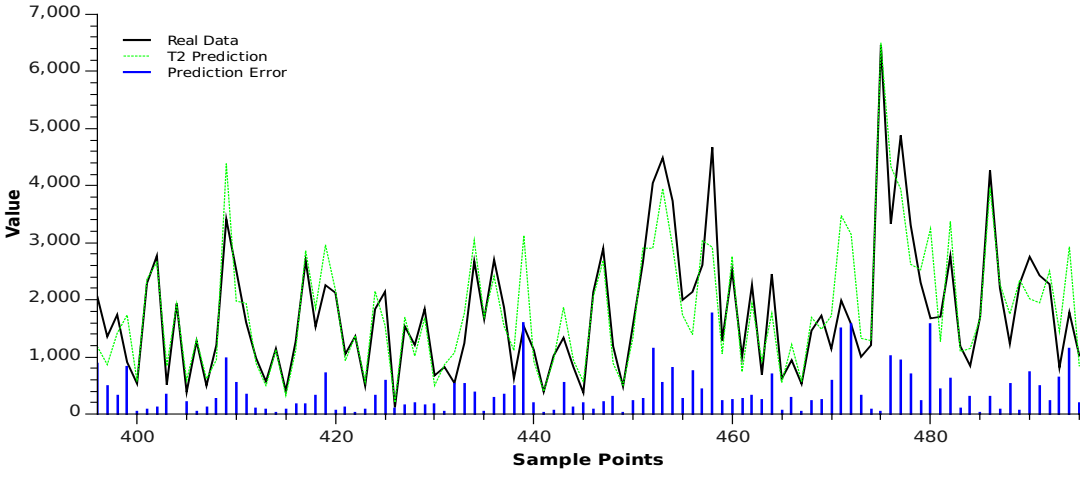

FIGURE 4.9: Estimation results and estimation errors for low voltage line problem by SA-IT2FLS

TABLE 4.3: Etimation results for low voltage electrical line length by simulated annealing with T1FLS and T2FLS

Type-1 FLS			
Statistics	$Train_{RMSE}$	$Test_{RMSE}$	$Time_{Seconds}$
Average	535.62	699.16	258.1
Standard Deviation	40.12	77.95	6.69
Minimum	480.5	587.37	247
Type-2 FLS			
Average	490.17	663.23	1,679.15
Standard Deviation	8.95	12.12	12.12
Minimum	445.79	540.89	1,584

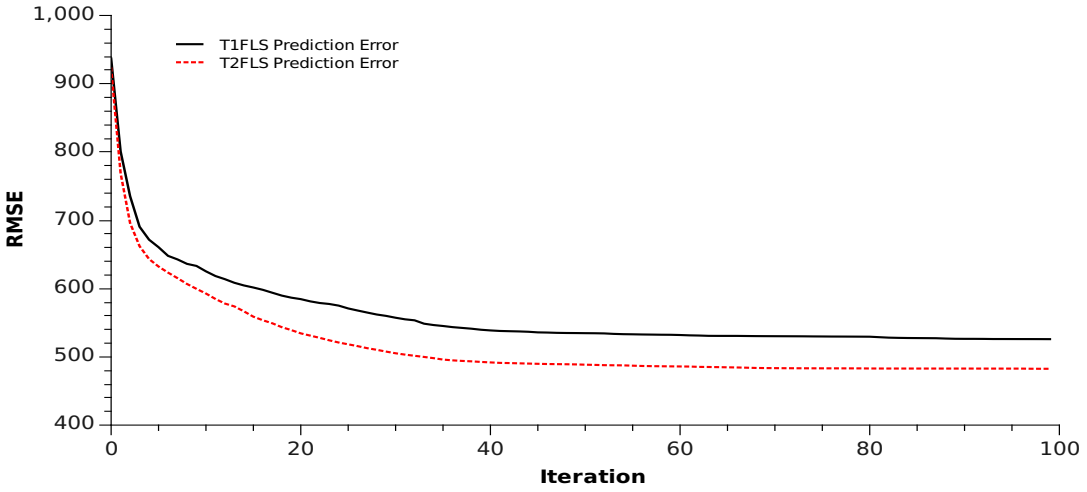

FIGURE 4.10: The average convergence of SA-T1FLS and T2FLS for low voltage line problem

4.4.4 Estimation of the medium voltage electrical line maintenance cost results

The results of the estimation of the medium voltage electrical line maintenance cost by SA-T1FLS and SA-IT2FLS are shown in Table 4.4 While the estimation accuracies and errors for the best runs are depicted in Fig 4.11 and 4.12. Although, this is a real data problem, the results show how accurate both SA-T1FLS and SA-IT2FLS where the predicted data are difficult to distinguish from the target data. Table 4.4

TABLE 4.4: Estimation results for the maintenance cost problem by simulated annealing with T1FLS and T2FLS

Type-1 FLS			
Statistics	$Train_{RMSE}$	$Test_{RMSE}$	$Time_{Seconds}$
Average	129.39	141.51	2,192.1
Standard Deviation	51.19	35.12	72.6
Minimum	81.78	103.39	2,092
Type-2 FLS			
Average	87.45	133.87	13,333
Standard Deviation	36.49	69.03	325.67
Minimum	60.04	75.24	12,806

shows that SA-IT2FLS obtained smaller errors in both the average RMSE and the best (minimum) RMSE in the testing phase indicating that SA-IT2FLS was able to capture more information and uncertainties than SA-T1FLS with an improvement on the average RMSE about 5% from the SA-T1FLS average RMSE. Although, SA-IT2FLS has the best average and minimum RMSE in all the four problems and smaller standard deviations in the previous three problem, the standard deviation for the 20 RMSE's in SA-IT2FLS was bigger in the testing phase compared to the one achieved by SA-T1FLS. On the training phase, Figure 4.13 shows the average error curves for both models where SA-IT2FLS shows smaller errors in the majority of iterations. On the other hand, the time taken to tune SA-IT2FLS was long compared to SA-T1FLS due to the extra computations associated with T2FLS.

4.4.5 Results summary

The result of using the two models to solve the four problems show that interval type-2 fuzzy logic system was able to handle more of the uncertainties and information of theses problems by providing the best accuracy results for all the four problems with different amount of improvements over type-1 fuzzy logic system. The only drawback reported for using interval type-2 fuzzy logic system is the amount of time needed to employ it due to the extra computations associated with interval type-2 fuzzy logic

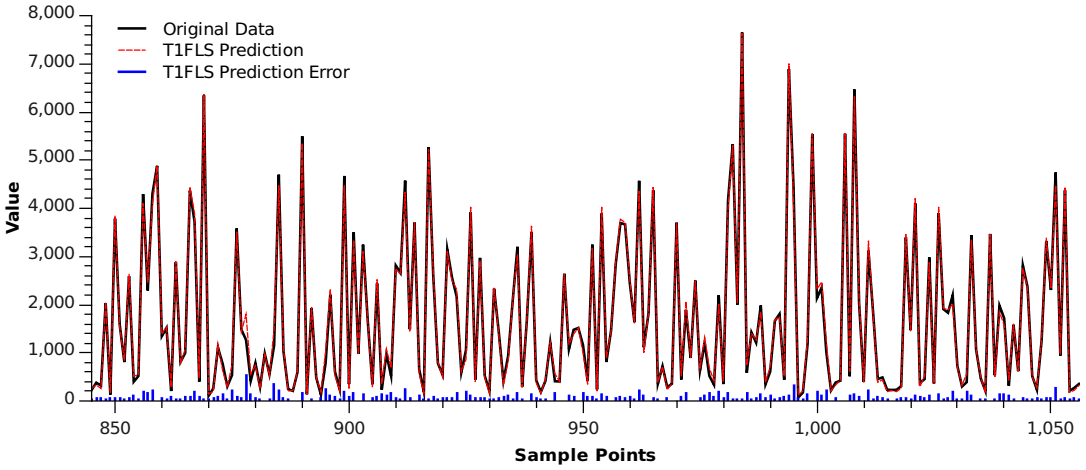

FIGURE 4.11: Estimation results and estimation errors for maintenance cost problem by SA-T1FLS

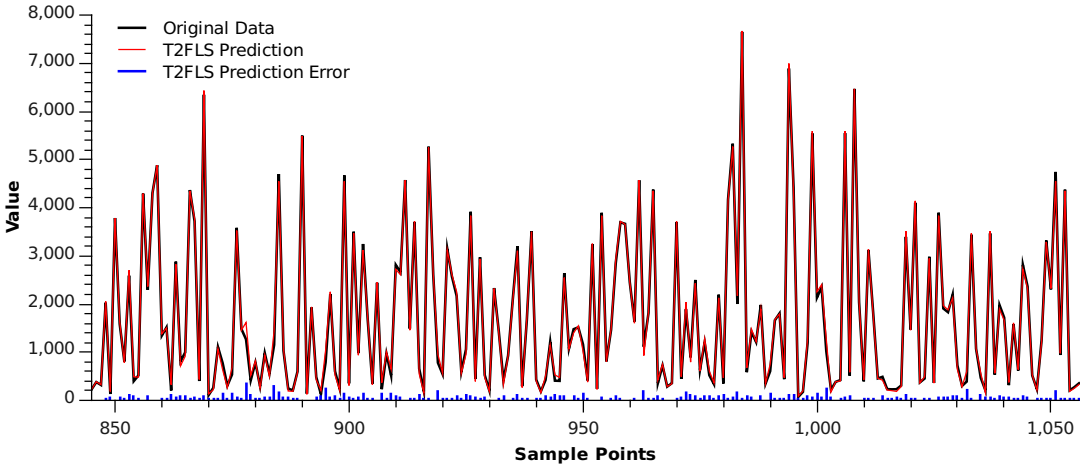

FIGURE 4.12: Estimation results and estimation errors for maintenance cost problem by SA-IT2FLS

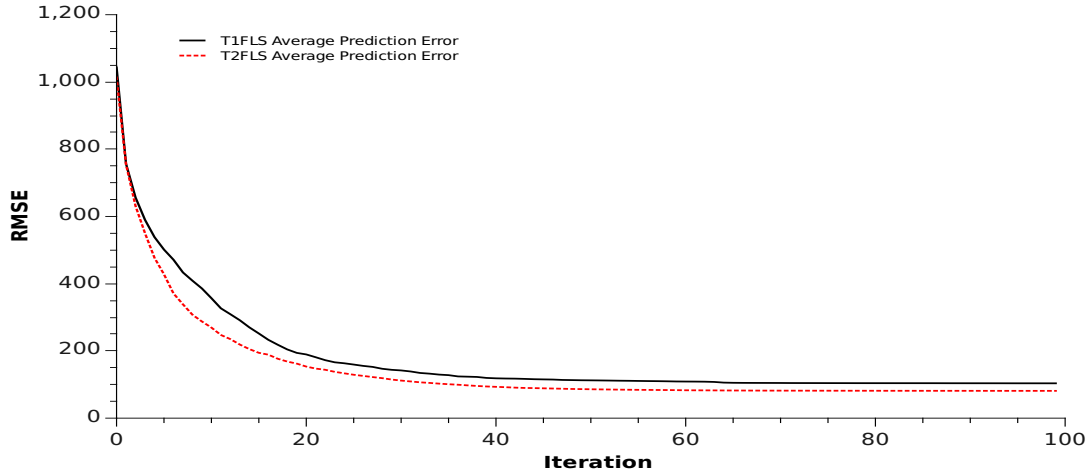

FIGURE 4.13: The average convergence of SA-T1FLS and T2FLS for maintenance cost problem

system compared with type-1 fuzzy logic system. As the aim of this work is to unveil the potentials for type-1 and type-2 systems without time constraints, a relatively long simulated annealing search was applied. Future work might look to reduce the time taken by SA-T2FLS by using some other variants of simulated annealing to allow them to be used in problems with on-line learning or with real-time problems.

4.5 Summary

In this chapter, simulated annealing is used with type-1 and interval type-2 fuzzy logic systems to provide models to solve two bench-mark time series and two real world electrical problems. Simulated annealing searches for the best configuration of the type-1 and interval type-2 fuzzy logic system parameters of the antecedent and the consequent parts of the rules for a Mamdani model. In addition, the issues related to the combination of simulated annealing and type-2 fuzzy logic systems and how they are combined have been discussed. The results of using the interval type-2 fuzzy logic system are compared to the results of using type-1 fuzzy logic system where it shows that interval type-2 fuzzy logic system was able to handle more of the uncertainties and

the information of theses problems by providing better accuracy results for all the four problems with different amount of improvements over type-1 fuzzy logic system. The only drawback reported for using interval type-2 fuzzy logic system is the amount of time needed to employ it due to the extra computations associated with interval type-2 fuzzy logic system compared with type-1 fuzzy logic system. As the aim of this work is to unveil the potentials for type-1 and interval type-2 fuzzy logic system without time constraints, a relatively longer simulated annealing search was applied. Future work might look to reduce the time taken by this combination by using some other variants of simulated annealing to allow them to be used in problems with on-line learning or with real-time problems. The next chapter will report the work of using simulated annealing with general type-2 fuzzy logic system.

Chapter 5

Designing General Type-2 Fuzzy Logic Systems using Interval Type-2 Fuzzy Logic Systems and Simulated Annealing

5.1 Introduction

In this chapter, simulated annealing algorithm is used to design general type-2 fuzzy logic systems (GT2FLS) with the aid of interval type-2 fuzzy logic systems (IT2FLS). The proposed practical design methodology aims to reduce computations needed to get the best footprint of uncertainty (FOU) using interval type-2 fuzzy logic systems. Simulated annealing is used to learn interval type-2 fuzzy logic systems followed by learning the secondary membership functions (SMFs) in general type-2 fuzzy logic systems using a novel parametrisation method. In addition, the proposed method can help to answer the question of whether the third dimension (SMF) can add more capabilities to handle

uncertainties and model more information over interval type-2 fuzzy logic systems parameters. The justification needed for the choice of the defuzzification method used in this thesis will be discussed including its effects on the learning process. The proposed methodology has been applied to four bench-mark problems. some practical issues have been discussed and some analysis of the results obtained by interval and general type-2 fuzzy logic systems have been drawn.

5.2 Learning the third dimension in general type-2 fuzzy logic systems

Type-2 fuzzy logic is a growing research topic with much evidence of successful applications (John and Coupland, 2007). However, up to now, almost all developments of type-2 fuzzy logic systems have been based on interval type-2 fuzzy logic systems with some exceptions related to using different representations of type-2 sets and systems such as geometric T2FLS (Coupland and John, 2007), alpha-planes (Mendel et al., 2009), alpha cuts (Hamrawi et al., 2010) and Z-slices (Wagner and Hagras, 2010*a*; Christian Wagner, 2009). The ease of computation associated with the interval form of type-2 sets is the main driver for the wide usage of interval type-2 set and systems compared to the generalised form. However, the third dimension in a general type-2 set offers extra degrees of freedom over interval type-2 fuzzy sets (Mendel and John, 2002). A main drawback of an interval type-2 set is that the uncertainty is spread equally across the FOU which prevents modelling of variations of the uncertainty (Christian Wagner, 2009). Hence, interval type-2 is a simplified and restricted form of general type-2 set. In 2001, Mendel stated the three areas where the complication of GT2FLS come from compared to IT2FLS (Mendel, 2001, p.302):

- The prohibitive computations to calculate the meet operation.
- The prohibitive computations to do type-reduction.

- There is no rational basis for choosing secondary membership functions.

Ten years after this statement, some research has been carried out in the first two areas. Examples of the first area including geometric meet and join (Coupland and John, 2007), optimised meet and join (Greenfield and John, 2007) and extended t-norms (Starczewski, 2009*a*). For type-reduction area, the geometric defuzzifier (Coupland and John, 2007), the sampling defuzzifier (Greenfield et al., 2005) followed by importance sampling defuzzifier (Linda and Manic, 2010) and a centroid defuzzifier based on the alpha representation (Liu, 2008) have been proposed. In addition, modern processors allow far more computational capability than ten years ago. The third area can be resolved by using some learning approaches or experts opinions to design a good IT2FLS to initialize GT2FLS. Then, choosing the best secondary membership functions by learning approaches. One attempt to design general type-2 sets based on zSlices representation was proposed in (Christian Wagner, 2009) where survey data and device characteristics were used to build zSlices sets automatically. Another work using alpha-planes representation has applied a learning method to the SMF in the GT2FLS to forecast Mackey-Glass time-series (Mendel et al., 2009). The latter showed a better performance of general type-2 fuzzy logic systems using a simpler model known as "triangle quasi-type-2 fuzzy logic system" first presented in (Mendel and Liu, 2008). Some other researchers used some neural networks concepts or classification algorithms such as: type 2 Adaptive Network Based Fuzzy Inference System (ANFIS) (John and Czarnecki, 1998), general type-2 fuzzy neural network (GT2FNN) (Jeng et al., 2009) and fuzzy C-means algorithm with a model known as "efficient triangular type-2 fuzzy logic system" (Starczewski, 2009*b*). To the best of the author knowledge, no attempt to employ a learning method to general type-2 fuzzy logic systems using the vertical-slice representation was reported. To achieve this objective, apart from using a practical type-reducer, some kinds of parametrisation are needed for general type-2 sets to allow learning or optimisation techniques to deal with these parameters easily rather than having all the secondary grades or membership functions to be chosen manually.

The parametrisation method should preserve the most of the freedom associated with GT2FLS. For example, interval type-2 fuzzy set is a parametrised form of general type-2 set.

5.3 A practical choice for general type-2 fuzzy set

To have a good practical form of general type-2 set, the chosen form should :

- Have a low computational burden.
- Preserve the most of the freedom associated with general type-2 sets.

These two objectives normally are in conflict as more freedom requires more computations. Therefore, some trade-offs are needed using some parameterisation mechanisms. One way to do this is to have parameterised secondary membership functions that are asymmetric and convex. For example, a triangular secondary membership function with an apex in the area between the lowest and the highest FOU points (FOU_{lower} and FOU_{upper}) primary memberships for each x in the domain. We mean by the apex the point in the secondary domain that represents the highest secondary membership grade for x. The asymmetry is preferred to allow optimising the apex location of the SMF when their primary memberships are fixed. The other preferred property is to have a convex SMF to allow quick meet and join operations when using these sets in GT2FLS. To allow learning the best location for the apex for each SMF, a function to determine the SMF's apexes locations in FOU for each x in the primary domain is needed. The apexes locations values must be bounded by the highest and the lowest FOU points (FOU_{upper} and FOU_{lower}) for each x in the domain. A possible approach is to have a piecewise linear function or a smooth piecewise-polynomial function and to use some interpolation methods. However, it could be possible to ensure this condition of boundaries when designing the first model of the general type-2 set but this

is very difficult to trace and ensure for each x in the continuous domain when learning $SMF_{apex}(x)$ as the interpolation might define some apexes domains out of the FOU boundaries. Therefore, a new parametric formula is proposed here that normalises the FOU apexes locations to be within $FOU(x)$ for each x in the primary domain. This is done by defining the $SMF_{apex}(x)$ as following:

$$SMF_{apex}(x) = h(x)/(FOU_{low}(x) + g(x) \times (FOU_{up}(x) - FOU_{low}(x))). \quad 0 \leqslant g(x) \leqslant 1. \tag{5.1}$$

Where $g(x)$ is a parameter called "apex factor" that is used as an apex location indicator for each x and $h(x)$ is the height (the secondary grade) of the apex for each x. This parameter can be used to change the apex location without the need to check for the boundaries condition. For example when $g(x) = 0.5$, the location of apex is in the middle between $FOU_{upper}(x)$ and $FOU_{lower}(x)$ and the resulting SMF is symmetrical. Therefore, this parameter is acting as a variable representing the apexes locations when doing some optimisation or learning for the general type-2 set. An example of using this parameter is to use a piecewise linear function to determine this parameter for all x in the primary domain such as: suppose that $k_1, k_2, ..., k_n$ are ordered points in the x domain and $g(k1), g(k_2), ..., g(k_n)$ are their apexes factors which both defining the piecewise linear function, then:

$$g(x) = \begin{cases} 0.5, & x < k_1 \\ g(k_i) + \frac{x-k_i}{k_{i+1}-x} \times (g(k_{i+1}) - g(k_i)), & k_i \leqslant x \leqslant k_{i+1} \\ 0.5, & x > k_n \end{cases} \tag{5.2}$$

Another similar function to determine the height of the apexes when non-normal SMF are used can be designed by the same way. For example:

$$h(x) = \begin{cases} 1, & x < k_1 \\ h(k_i) + \frac{x-k_i}{k_{i+1}-x} \times (h(k_{i+1}) - h(k_i)), & k_i \leqslant x \leqslant k_{i+1} \\ 1, & x > k_n \end{cases} \tag{5.3}$$

This form is not identical to the principal function described in equation 2.1 or the fuzzy truth numbers proposed in (Starczewski, 2009*b*) because for each x value, the SMF can be non normal. The lowest and highest FOU points (FOU_{lower} and FOU_{upper}) for each x can be defined by another functions such as trapezoidal, Gaussian or triangular functions or any other functions used to define interval type-2 sets. An example of the proposed method is shown in figure 5.1 and an example of learning the secondary membership functions is shown in figure 5.2. The chosen form is based on the last general type-2 literatures using the vertical-slice representation and the novel method we proposed here to determine the apexes locations and heights of SMFs. Although, this is not a new representation and can not be generalised for all forms of general type-2 sets, the aim of this method is to have general type-2 fuzzy sets simplified for practical usage.

5.4 A proposed methodology to design general type-2 fuzzy logic systems

The GT2FLS can be designed using the method described above to build general type-2 sets used in GT2FLS. The first choice is to build GT2FLS directly without the need to initialise it from a good IT2FLS design. This choice is more computationally expensive due to the higher computations needed for GT2FLS. The next chapter will investigate this choice. The second choice is to design GT2FLS using IT2FLS in three steps:

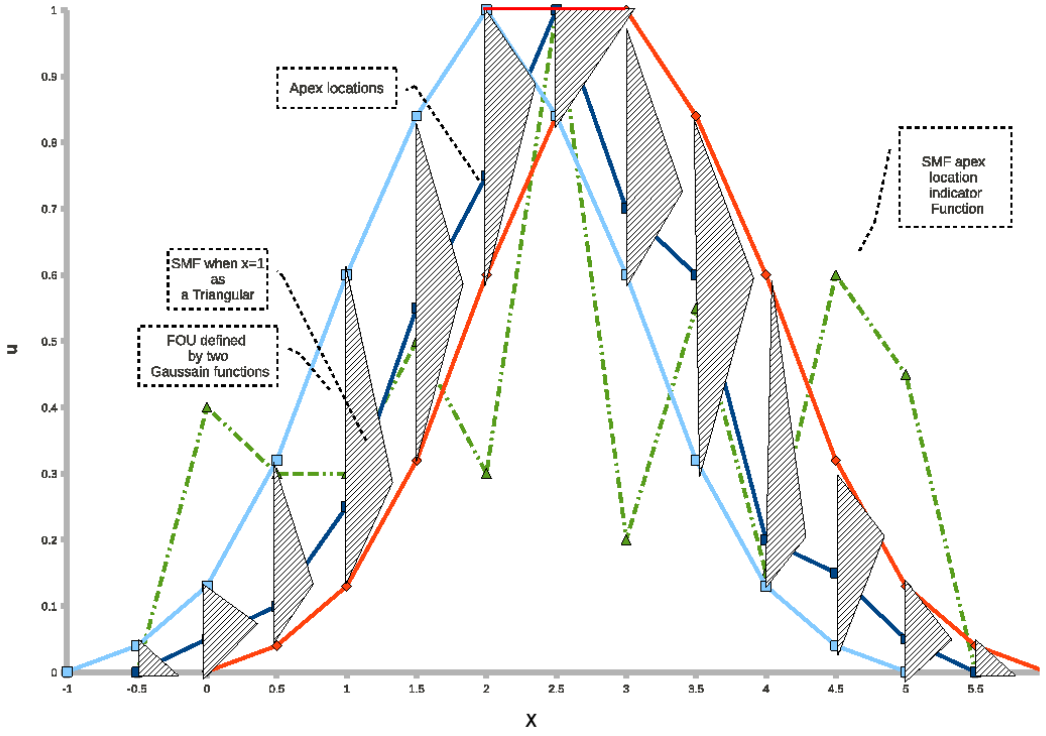

FIGURE 5.1: General type-2 fuzzy Set defined by its FOU (using two piecewise linear functions) and a triangular SMF (using linear interpolation for its apexes indicators).

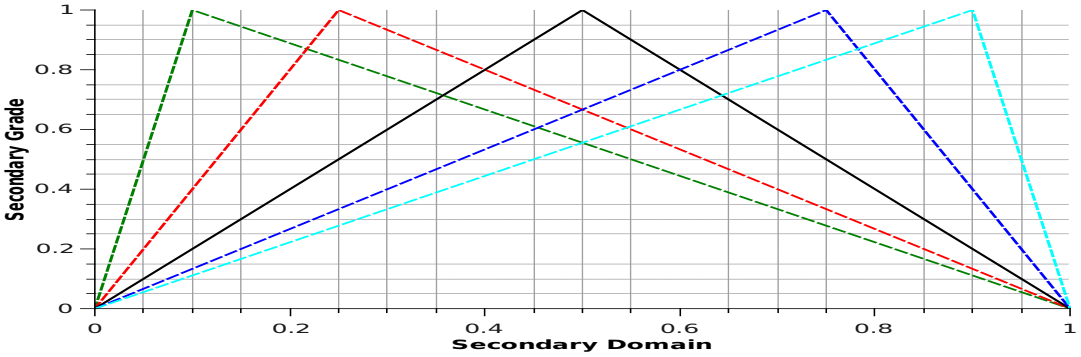

FIGURE 5.2: An example of learning a triangular SMF by adapting the apex location.

- The first step is to get a good or optimal FOU. This is done using IT2FLS as each interval type-2 set is bounded by two functions for the lower and upper membership functions that bound the FOU in the secondary memberships domain. To get a good IT2FLS, experts opinions or automated learning can be applied.
- Using the optimal interval type-2 sets to design general type-2 sets by converting them into general type-2 sets. This stage might be called the conversion stage where each general type-2 set is initialised using the same FOU brought by interval type-2 sets and a chosen initial SMF.
- Learning the best location for the apexes of each SMF by fixing the optimal FOU to get the best secondary membership functions.

This three stages method seems to be logical as the definition of the uncertainty boundaries (primary memberships) should precede the definition for how much secondary membership grades (uncertainty distribution) will be given to each primary membership. In addition, the conversion stage proposed reduces the computations needed to get the best FOU using IT2FLS mathematics. The third choice is to start by applying the first step followed by the conversion stage, learning the FOU using GT2FLS mathematics and then, learning the SMF. This chapter's work is based on the second choice.

5.5 The choice for the defuzzification method

As mentioned in Chapter 2, the bottleneck part of the general type-2 fuzzy logic system is the defuzzification phase. This is due to the high computational burden associated with the type-reduction process. Therefore, special attention should be given to the choice of such methods. The aim of this section is to highlight this issue and its effects on the learning process. Based on our choice for the representation of general type-2 fuzzy sets using vertical-slices, the available defuzzification options are :

1. The exhaustive brute-force highly expensive type-reduction method which computes the union of all the centroids of all the embedded type-2 fuzzy sets involved in the general type-2 fuzzy set. Now, we explain how this method is impractical to use for our purpose.

   The exhaustive method calculates the type-reduced set by the following procedure (Mendel, 2001, p.250):

   (a) Discretise the x-domain into a suitable number of points N i.e. $x_1, x_2, \ldots, x_N$.

   (b) Discretise each secondary membership domain into a suitable number of points $M_j (j = 1, \ldots, N)$.

   (c) Enumerate all embedded sets involved in the general type-2 fuzzy set. The number of embedded sets will be $p = \prod_{j=1}^{N} M_j$ as described in (2.3).

   (d) For each embedded set $\tilde{A}_e$, calculate its centroid (a type-1 fuzzy set $C_{\tilde{A}_e}$) using the following steps:

   - Assign the minimum secondary membership grade of the embedded set $min(z_1, z_2, \ldots, z_n)$ to its centroid membership grade.
   - Assign the centroid of the primary memberships of the embedded set to its centroid domain value x. The calculation of this part is based on the centre of area (centroid) of type-1 fuzzy sets. Therefore, the centroid of each embedded set can be calculated as follows:

   $$C_{\tilde{A}_e} = min(z_1, z_2, \ldots, z_n)/(\frac{\sum_{i=1}^{n} x_i \cdot z_i}{\sum_{i=1}^{n} z_i}$$

   (e) For each x-domain value for the resultant type-1 fuzzy sets, the maximum membership grade found for each x-domain value is paired with its x-domain value. The resultant set of pairs is a type-1 fuzzy set represents the exact type-reduced set.

TABLE 5.1: The number of centroid operations needed to type-reduce general type-2 fuzzy sets that are needed to get outputs in case of iterative evaluations for optimisation

X domain points	Y domain points	Number of embedded sets	FLS samples number	SA iterations	Number of centroid operations for embedded sets
25	5	3e+17	200	10,000	6e+23
25	9	7.2e+23	200	10,000	1.44e+30
50	9	5.2e+47	200	10,000	1.04e+54
101	9	2.4e+96	200	10,000	4.8e+102

In practice, the number of embedded sets is normally astronomical and above the current data structure. For instance, for a general type-2 fuzzy sets discretised into our choice of 101 x-domain points and each vertical slice into 9 points, the number of embedded sets are 2.39×10^{96} which is -by far -above current data structure. In our chosen language (C++), the longest data structure size can be allocated is unsigned integer $= 2,14 \times 10^{9}$. In addition, this astronomical computations are not the end of the story in our case. This number of embedded sets are unioned to get *ONE* sample output in *ONE* fuzzy logic system evaluation in *ONE* iteration of the optimisation process. Table 5.1 shows how type-reduction complexity evolves in our problem with some reasonable choices of fuzzy logic system input samples and reasonable number of simulated annealing iterations. Note that table 5.1 is for the type-reduction operations only (i.e. does not include fuzzification and other fuzzy logic system operations). Therefore, this choice is impractical to choose for our models in this thesis.

2. The recursive algorithm introduced by (Gafa and Coupland, 2011). It includes some interesting ideas to reduce these computations but the complexity is still very high and no practical results were published yet.

3. The sampling defuzzifier (Greenfield et al., 2005). As stated in Chapter 2, this is a practical approach that improves the speed of type-reduction with a little loss in accuracy (Greenfield et al., 2012). It approximates the type-reduced sets

using a sample of the embedded sets rather than using all embedded sets. The approximation of some centroids of general type-2 fuzzy sets reported in (Greenfield et al., 2012) shows no significant loss of accuracy. However, this might affect the learning process by a some degrees as will be explained later. The sampling method is working the same as the exhaustive method described above apart from the third step as follows:

(a) Discretise the x-domain into a suitable number of points N i.e. $x_1, x_2, \ldots, x_N$.

(b) Discretise each secondary membership domain into a suitable number of points $M_j (j = 1, \ldots, N)$.

(c) **Enumerate a suitable number of embedded sets involved in the general type-2 fuzzy set randomly.**

(d) For each embedded set $\tilde{A}_e$, calculate its centroid (a type-1 fuzzy set $C_{\tilde{A}_e}$) using the following steps:

- Assign the minimum secondary membership grade of the embedded set to its centroid membership grade z.
- Assign the centroid of the primary memberships of the embedded set to its centroid domain value x. The calculation of this part is based on the centre of area (centroid) of type-1 fuzzy sets:

$$C_{\tilde{A}_e} = \frac{\sum_{i=1}^{n} x_i \cdot z_i}{\sum_{i=1}^{n} z_i}$$

(e) For each x-domain value in the type-1 fuzzy set, the maximum membership grade found for each x-domain value is paired with its x-domain value. The resultant set of pairs is a type-1 fuzzy set **represent an approximate type-reduced set**.

4. The vertical slice centroid type-reducer (VSCTR) which initially proposed by (John, 2000) then detailed by (Lucas et al., 2007). It does not calculate the union

for all the embedded sets involved in the general type-2 fuzzy sets. Although, this method does not depend on the concept of embedded sets, it is a good approach for practical usage. This method works as follows:

- For each vertical slice, the centroid of each vertical slice is calculated exactly as type-1 set centroid calculation.
- The type-reduced set domain is the same as the vertical slices values. The membership grades of the type-reduced set are the centroids of these vertical slices in the type-reduced set.

When optimising the FOU's and SMF's parameters using the non-deterministic sampling defuzzifier, the learning process is affected -by some degree- by the random errors and the fluctuations of the evaluation of the objective function. The effects come from the fact that the evaluation of one state will differ each time the sampling method approximate the type-reduced sets. Consequently, the outputs of the fuzzy logic system will be changed causing the objective function (RMSE) to get different energy values for the same state each time the evaluation is carried out. Whatever the objective function is, the outputs from the fuzzy logic system will affect that objective function. In fact, these random errors are small compared to the scale of the fuzzy logic system outputs and the scale of the objective function but affects the learning performance as will be shown later in this chapter. These effects can be ignored in the first exploration stages of the search when moves from a state to a state can bring relatively large differences but this noise can deteriorate the search at the last stages when small effects of the objective function can be affected by this noise. In optimisation, noise associated with the objective function has some effects on the quality of the solution. This is illustrated in these points :

1. In all optimisation problems, the criteria used by the search algorithm to decide the quality of a state is very important for the success of the search algorithm.

This criteria should be precise and should avoid stochastic measurements or approximation as possible. However, in real-world, this condition is difficult in many cases. Reasons for using noisy objective functions include cases where no precise and numeric measurements available as well as cases where the evaluation of the precise objective functions is computationally expensive (Branke et al., 2008). In fact, our problem here falls under the second case where the computation for one state using the precise algorithm is impractical. Therefore, the use of sampling method is highly acknowledged in practice.

2. In the simulated annealing literature, many papers have tackled the problem of noisy objective functions. All solutions proposed in simulated annealing literature fall under these three categories (Branke et al., 2008):

   (a) Solutions adapt the convergence properties to allow better handling of noisy objective functions. These methods rely on storing all visited states and their evaluations or increasing the number of iterations according to a known schedule. This type of solutions adds extra computations and needs larger memories to be executed.

   (b) Solutions rely on revisiting each state a number of times to improve the approximation to the true objective function values. Then, using some statistical approaches to calculate approximated objective functions. Again, this is computationally expensive depends on the number of evaluations n needed for each state. Hence, the computations will be multiplied by n.

   (c) Solutions adapt the acceptance function to maintain an adequate thermodynamic equilibrium. Unfortunately, the three solutions require more computations and do not guarantee the exact objective functions.

In general, when dealing with noisy objective functions, we are not interested in exact best solutions, rather, we are interested in alternatives to the best energy value that are nearly equally good (Salamon et al., 2002, p.64). Therefore, the

use of these methods adds another computational burden and does not lead to more accurate solutions. In order to get a fair comparison with interval type-2 fuzzy logic system, the use of such methods is not the best choice for our purpose. Therefore, a solution can be sought from the general type-2 fuzzy logic system side. This is by using a deterministic approach to get the type-reduced set such as the vertical slice centroid type-reducer (VSCTR). As our aim in this chapter is to apply the learning process only on the third dimension of general type-2 fuzzy sets, the choice of sampling method is applicable for this purpose as no comparison will be applied between interval type-2 fuzzy logic system and general type-2 fuzzy logic system where the third dimension learning will be applied in a separate stage as described in previous subsection. The comparison of the two models will be reported in the next chapter where a full learning of all parameters of both systems will be tested.

5.6 Methodology

The experiment can be divided into five steps : preparing data, constructing the initial interval type-2 fuzzy system, learning the interval type-2 fuzzy system parameters, constructing the initial general type-2 fuzzy system using the proposed conversion and learning the secondary membership functions.

5.6.1 Data

5.6.1.1 Mackey-Glass time series

The Mackey-Glass time series has been described in section 3.3.1 but it is used here with different number of samples. To get the time series, firstly, the noise-free time series is generated with the following parameters : $a = 0.2$, $b = 0.1$, $\tau = 17$ and $n = 10$.

TABLE 5.2: The time needed for learning GT2FLS to predict Mackey-Glass time series by simulated annealing

The number of training samples	200	500	1,000
One sample evaluation time (Seconds)	0.000373	0.000373	0.000373
Time needed for one evaluation of all samples	0.0746	0.1865	0.373
Time needed for a Markov chain (3000)	223.8	559.5	1,119
Time needed for 40 Markov chains	8,952	22,380	44,760
Time needed for 20 runs	179,040	447,600	895,200
Time needed for all problems (80 runs)	716,160	1,790,400	3,580,800
Time needed in *days*	8.288889	20.722222	41.44

The Runge-Kutta method is used to obtain the values of $x(t)$ at each time point with a time step of 0.1 and the initial condition $x(0) = 1.2$ where $x(t) = 0$ for $t < 0$. The input-output samples are extracted in the form $x(t-18)$,$x(t-12)$,$x(t-6)$ and $x(t)$ from $t = 118$ to $t = 417$ using a step size of 6. A sample of the generated noise-free sample is depicted in figure 5.3. Then the generated data are divided into 200 data points for training and the remaining 200 data points for testing. Using a step size of 6, the input values to the fuzzy system are the previous points $x(t-18)$,$x(t-12)$,$x(t-6)$ and $x(t)$ while the output from the fuzzy system is the predicted value $x(t+6)$. Four initial input values $x(114)$,$x(115)$,$x(116)$ and $x(117)$ are used to predict the first four training outputs. This number of training samples used here (200) is different from the previous chapters (500). The reason behind this reduction on the number of samples is the impractical computational time that needed for the 500 samples as discussed in previous subsections. Therefore, we choose to reduce the number of samples in order to get the learning process in an acceptable time. To have a look at this issue, table 5.2 can highlight this problem. This table was built based on the experiments times of learning GT2FLS for Mackey-Glass time series that will be presented later in this chapter. It is clear from the table how difficult to choose more training samples.

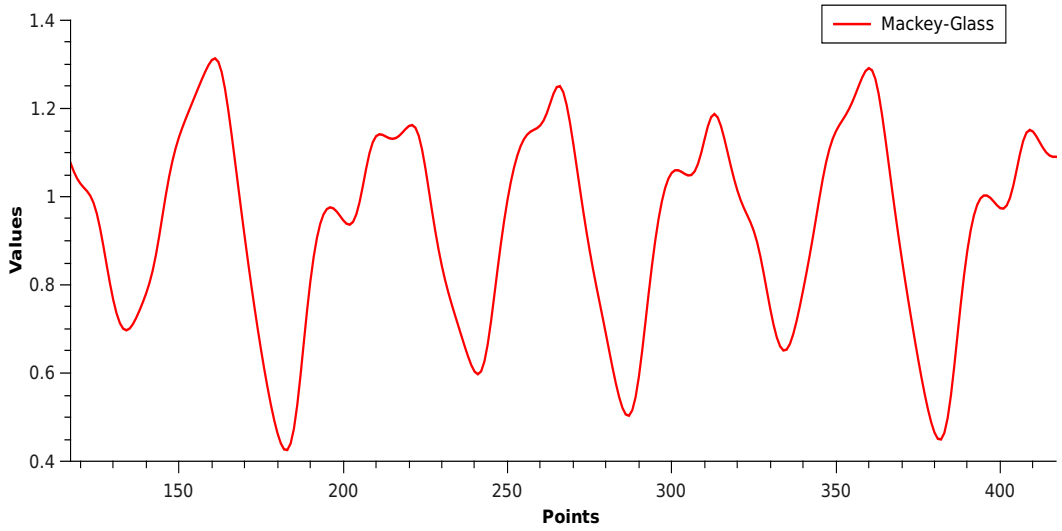

FIGURE 5.3: A sample of Mackey-Glass time series points values used in this experiment

5.6.1.2 Mackey-Glass time series with added noise

In this experiment, a noisy time series will be used to test our models. The noisy Mackey-Glass time series will be generated by adding noise to Mackey-Glass time series that are generated as described above. The amount of noise will be $20db$ added to all inputs and outputs. The noise is measured by signal-to-noise ratio (SNR). Again, the number of training samples used here is 200 which is different from the previous chapters (500) as explained above. A sample of the generated noise-free and noisy data that used in this experiment is depicted in figure 5.4.

5.6.1.3 Estimation of the low voltage electrical line length in rural towns

This problem has been introduced in section 4.3.1. The data samples were randomly divided into two sets labelled training and testing sets which are randomly selected from the whole sample as reported in (Cordón, Herrera and Villar, 2001) and (Cordón et al., 2002). As with other authors, 396 samples are used for training while the other

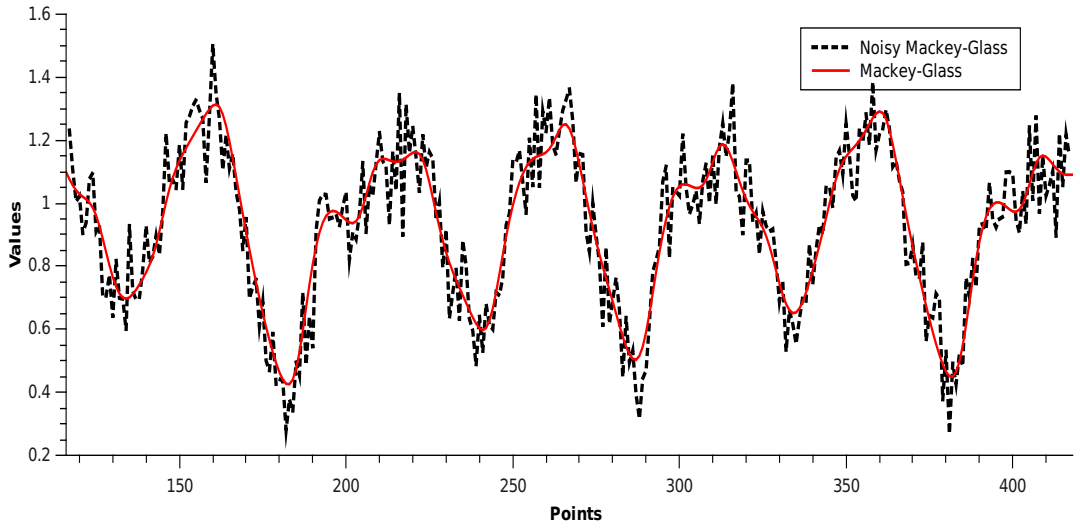

FIGURE 5.4: A sample of the noisy Mackey-Glass time series points values used in this experiment

99 samples are used for testing. The number of samples was chosen the same as others without a reduction due to the smaller number of inputs involved in this problem (2 instead of 4) which reduces the computational burden.

5.6.1.4 Estimation of the medium voltage electrical line maintenance cost

This problem has been introduced in section 4.3.1. In order to reduce the training computations and time, the number of samples has been reduced from the one used in Chapter 4. 400 data samples from the whole set were divided into two sets labelled training and testing sets with 200 samples for each set. Therefore, 400 samples have been used instead of 1056 samples.

5.6.2 The initial interval type-2 fuzzy logic systems

The fuzzy model consists of a number of independent input fuzzy sets and one independent output fuzzy set for each rule. There are 8 rules while each rule is characterised

by a number of fuzzy sets equals to the inputs number (i.e. 4 antecedent fuzzy sets and one consequent fuzzy set). However, the number of rules was chosen heuristically where we are interested in reducing complexity. However, any number of rules can be chosen as the model is not a linguistic fuzzy system. The system is built from scratch rather than using the optimised type-1 system to initialise the fuzzy sets. Each type-2 fuzzy set is described by Gaussian primary membership functions with uncertain means represented by two means and one standard deviation as follow (Mendel, 2001, p.91):

$$\tilde{f}(x) = \exp^{-(\frac{x-m}{2\sigma})^2} \; m \in [m_1, m_2] \tag{5.4}$$

Therefore the upper $\overline{\mu}_{\tilde{A}}(x)$ and lower $\underline{\mu}_{\tilde{A}}(x)$ membership functions are defined by following mathematical functions (Mendel, 2001, p.91):

$$\overline{\mu}_{\tilde{A}}(x) = \begin{cases} \exp^{-(\frac{x-m_1}{2\sigma})^2} & \text{if } x < m_1 \\ 1 & \text{if } m_1 \leq x \leq m_2 \\ \exp^{-(\frac{x-m_2}{2\sigma})^2} & \text{if } x > m_2 \end{cases} \tag{5.5}$$

$$\underline{\mu}_{\tilde{A}}(x) = \begin{cases} \exp^{-(\frac{x-m_2}{2\sigma})^2} & \text{if } x \leq \frac{m_1+m_2}{2} \\ \exp^{-(\frac{x-m_1}{2\sigma})^2} & \text{if } x > \frac{m_1+m_2}{2} \end{cases} \tag{5.6}$$

Where the upper $\overline{\mu}_{\tilde{A}}(x)$ and lower $\underline{\mu}_{\tilde{A}}(x)$ membership functions in this equation are used to define FOU_{lower} and FOU_{upper}. All the means and standard deviations are initialised for all the input fuzzy sets by partitioning each input space into 4 fuzzy sets and enabling enough overlapping between them while the output fuzzy sets are initialised randomly around the average value of training outputs. The fuzzification process is based on the minimum t-norm while the centre-of-area has been chosen for type-reduction. The collapsing method proposed by (Greenfield, Chiclana, Coupland and John, 2009) has been used to calculate the centroids of the type-2 sets that needed to compute centre-of-area. This is done by using the composite outward right-left variant of the collapsing method as it is described in (Greenfield, Chiclana and John,

2009). The training procedure aims to learn the parameters of the antecedent parts and the consequent parts of the fuzzy system rules. Then, the found parameters are used to predict the next testing data points. The total number of the parameters is $8 * 4 * 3 + 8 * 3 = 120$ in all problems except the line length problem where it is $8 * 2 * 3 + 8 * 3 = 72$ parameters.

5.6.3 The learning of the interval type-2 fuzzy systems

The learning process of the interval fuzzy systems is done using simulated annealing that searches for the best configuration of the parameters by trying to modify one parameter each time and evaluate the cost of the new state which is measured by a cost function which is Root Mean Square Error (RMSE) that is defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{k=1}^{n}[f(k) - f(k^*)]^2} \tag{5.7}$$

The only constraint on the variables of the optimisation problem is that all new values of standard deviations of all fuzzy sets must be ≥ 0. The simulated annealing algorithm is initialised with a temperature that equals to the standard deviation of mean of RMSE's for 1000 runs for the training samples as proposed by (White, 1984). The cooling schedule is based on a static cooling rate of 0.9 updated for each Markov chain. Each Markov chain has a length related to the number of variables in the search space which equals to 30 times the number of variables. The search ends after a finite number of Markov chains namely 100 Markov chains. The new states for a current state are chosen from neighbouring states randomly by adding a small number (step size) to one of the antecedent parameters or the consequent parameters. The step size value is related to the maximum and minimum values for each variable space and = max-min/200 while the direction of the search is chosen randomly. After that, the new state is evaluated

by examining the training samples outputs. Then, the average and the minimum of the cost function of the training and testing data results have been calculated.

5.6.4 The initial general type-2 fuzzy logic systems

The initial general type-2 fuzzy logic system is built by using the structure described in sections 5.3 and 5.4 using the optimal configuration for IT2FLS as follows:

1. **The general type-2 set definition** The FOU is defined by the two Gaussian membership functions that define the interval type-2 sets. Our choice for the SMFs in this work is to use a triangular SMF with a normal apex initialised in the middle between (FOU_{lower} and FOU_{upper}) for $k_1, k_2, ...k_n$ points ($n = 7$) by choosing their apexes locations indicators $g(k_1) = g(k_2) = ... = g(k_n) = 0.5$ and then calculating the apex locations for other x points using linear interpolation as described by Equation 5.2. The other two SMF triangular points are defined by $FOU_{lower}(x)$ and $FOU_{upper}(x)$ points which are taken from the two Gaussian functions that define the interval type-2 sets. Therefore, apex location values are constrained by $FOU_{lower}(x)$ and $FOU_{upper}(x)$.

2. **Fuzzification** The fuzzification process will fuzzify each x value into a type-1 fuzzy set (SMF) which is a triangular function as described above. The fuzzified SMF is described by FOU_{upper} and FOU_{lower} which are derived from the two Gaussian functions for x and its apex location indicators. The output from each fuzzification process is a triangular SMF.

3. **Combination of antecedents** The combination between all antecedent fuzzified values is done using the meet operation proposed by (Coupland and John, 2007) and explained in equation 2.8. This operation is chosen because all SMF's are convex. Therefore, reducing unneeded computations. The output from this phase is a convex SMF that might be non-normal.

4. **Implication** To do the implication phase, firstly, the consequent sets space is discretised into $n = 101$ points $y_1, y_2, ..., y_n$ in Y domain. Then the implication is done using the same meet operation proposed by (Coupland and John, 2007). The third step is to do a join between all secondary membership grades for each $y \in Y$ using the join operation proposed in (Coupland and John, 2007) and explained in equation 2.3.4. Again this join operation is chosen to reduce the computations as all the SMFs in this phase are convex.

5. **Type-Reduction** The sampling method (Greenfield et al., 2005) has been used to calculate an approximate centre of area (centroid). Although, this method showed a good approximation to the centroid (Greenfield et al., 2005), we prefer to increase the number of samples of the embedded sets to get extra precision. We use 100 samples of the embedded sets. The output from this phase is a type-1 fuzzy set.

6. **Defuzzification** Any defuzzification method for type-1 fuzzy sets can be chosen in this phase. Our choice in this experiment is to use the centre of area (centroid) defuzzification.

7. **Approximation and descretisation issues** To get a good approximation to the interval type-2 fuzzy logic systems outputs, the discretisation used to get the centroid of type-2 sets should be the same in both systems. The more finer discretisations can help get more accurate outputs but increase the computations required to calculate the centroid. Hence, it is a trade-off between accuracy and speed. The configurations of IT2FLS and GT2FLS used in this experiment are detailed in Table 5.3.

5.6.5 The learning of the secondary membership functions

The learning process in this stage aims to get the optimal location for all the SMF's parameters where two points for each triangular SMF are fixed. The parameters in this

TABLE 5.3: The configurations of IT2FLS and GT2FLS

Stage	**IT2FLS**	**GT2FLS**
Membership Function	Gaussian	Gaussian + triangular SMF
Number of parameters (four inputs)	120	360
fuzzification	singleton	singleton
Antecedent combination t-norm	minimum	minimum using Coupland's meet
Implication t-norm	minimum	minimum using Coupland's meet
Join t-conorm	maximum	maximum using Coupland's join
SMF discretised points	not needed	9
Type-reduction method	collapsing method	sampling method
Defuzzification method	centroid	centroid
Y Descritisation points	101	101

case are the apexes location factors $g(k_1), g(k_2), ..., g(k_n)$ for each general type-2 set involved in the system. The learning is done using simulated annealing algorithm with the same configuration used when learning IT2FLS apart from the following :

- The constraints for optimised variables (apexes factors) for each k_i are their ($FOU_{lower}(k_i)$ and $FOU_{upper}(k_i)$) points (the secondary domain boundaries) for each k_i.
- The step size is the value that changes the apex location indicator. The new value must be between $[0, 1]$ and the step size should be large enough to make a difference in the cost function as small values might not change the outputs when it does not overcome the next descretisation step in SMF. The chosen step size is 0.15.
- The length of each Markov chain is equal to 10 times the number of variables in the search space. The search ends after 40 Markov chains.

The number of all parameters being adapted in this stage for each fuzzy set is $n = 9$. Therefore, The total number of all parameters being adapted in this stage is (the number of fuzzy sets $* n$ which is $40 * 9 = 360$ in all problems except the line length problem where it is $24 * 9 = 216$ parameters.

5.7 Results and discussion

The experiment has been developed using the C++ language and has been carried out on a number of PCs with an equal CPU speed of 3 GH'z and a memory of 4GB. After repeating the experiment for 20 times, the average and the minimum of the training and testing data results have been calculated. The results are shown for the three stages, SA-IT2FLS learning , the conversion stage and SA-GT2FLS learning for the four problems. The results of the predictions and the estimations for the four problems by simulated annealing with interval type-2 fuzzy logic systems (SA-IT2FLS) and general type-2 fuzzy logic systems (SA-GT2FLS) are shown in Tables 5.4, 5.5, 5.6 and 5.7. The average RMSE's during the search are depicted in figures 5.6, 5.8, 5.10 and 5.12. The main observations from these tables are :

1. The conversion process has caused some small losses in accuracy when converting the optimal IT2FLS to GT2FLS in both training and testing samples. This loss over the IT2FLS accuracies are between -0.36% and 3.78% of the average RMSE's in the training samples and between $+0.14\%$ and -2.47% of the average RMSE's in the testing samples. These losses were negative in the majority of times while it improves the accuracy in one case only in the testing phase in the noisy Mackey-Glass problem. Figure 5.5 shows one example for the results obtained by IT2FLS and the initial GT2FLS for the testing samples in Mackey-Glass time series problem. These losses caused by the conversion process might be due to two factors :

    - The difference in the fuzzy logic systems components in both systems. This is mainly in defuzzification stage where the collapsing method is used in IT2FLS and the sampling method is used in GT2FLS.
    - The difference between the unity secondary membership functions in IT2FLS and their approximated triangular shaped secondary membership functions

in GT2FLS. Therefore, the resulted outputs will differ because of the different numbers yielded by the t-norms and t-conorms operations in the secondary membership functions. However, the approximation still preserves the 97-98% of the IT2FLS outputs which might be a good approximation in some applications. Although, the proposed method of apex factors can be extended easily to trapezoidal membership functions by using two apex factors for the two unity endpoints to get a better approximation, the optimised operations used for join and meet are not applicable to use. As this work is mainly interested in designing GT2FLS in a practical manner and not investigating the best approximation methods from IT2FLS to GT2FLS, this issue is worth investigating in the future.

2. The learning of the secondary membership functions has improved the results over the initial general type-2 sets. The improvements are shown in all the training samples and in three out of four testing samples. However, the effects of the accuracy losses in the conversion stage were larger than these improvements brought by SMF's learning in some cases resulting in close results to the IT2FLS in some cases. However, these improvements are small compared to the scale of the values and the impacts of the FOU's learning. Also, the effects of using a stochastic defuzzification method can be seen from the unpleasant smaller acceptance ratios during the learning of SMFs as shown in the right sides of figures 5.7, 5.9, 5.11 and 5.13. Acceptance ratio is defined as the number of accepted moves divided by the number of all moves (accepted and rejected moves). In typical implementations of simulated annealing , acceptance ratios start at values close to 1 (the majority of moves) and gradually decrease to values close to 0 at the last iterations. The smaller acceptance ratios during the learning of SMFs were close to 0 in less than 10 Markov chains which means no improvements or equal moves were obtained in the rest of iterations as a result of noisy objective function values. However, the learning of SMFs brought these small improvements despite these effects. In

the next chapter, another defuzzification method which is not stochastic will be used to compare the full learning process of general type-2 fuzzy logic systems with interval type-2 fuzzy logic systems.

3. The average time for one evaluation of one sample of inputs is 0.00009 seconds in IT2FLS and 0.000323 seconds in GT2FLS in which the GT2FLS time is only 3.6 times the IT2FLS time. This reduction in time for GT2FLS is a good indicator of the practicality of the work.

We conclude from the above that the the proposed method reduced computation time. Although, the comparison between IT2FLS and GT2FLS is not accurate due to the accuracy losses in the conversion stage, the learning of the SMFs brought extra improvements over the initial converted sets which enforces the idea that the third dimension in GT2FLS adds extra ability to model uncertainty over IT2FLS (Mendel and John, 2002) (Christian Wagner, 2009). Although, the learning of the FOUs parameters has not been tried after the conversion stage in this work, we can expect a good improvement of the results when applying the learning to all general type-2 set parameters (FOU and SMF).

5.8 Summary

This work has proposed a methodology to design GT2FLS using a three stages methodology to reduce computations needed to get the best footprint of uncertainty (FOU). The first stage is to design IT2FLS using simulated annealing algorithm until getting good results then a conversion process will convert interval type-2 sets into symmetrical general type-2 sets followed by the learning of the secondary membership functions using simulated annealing. A novel parametrisation method has been proposed to allow learning the SMFs in GT2FLS easily. The results showed that the conversion process brought a good approximation to the IT2FLS outputs with small losses in accuracies

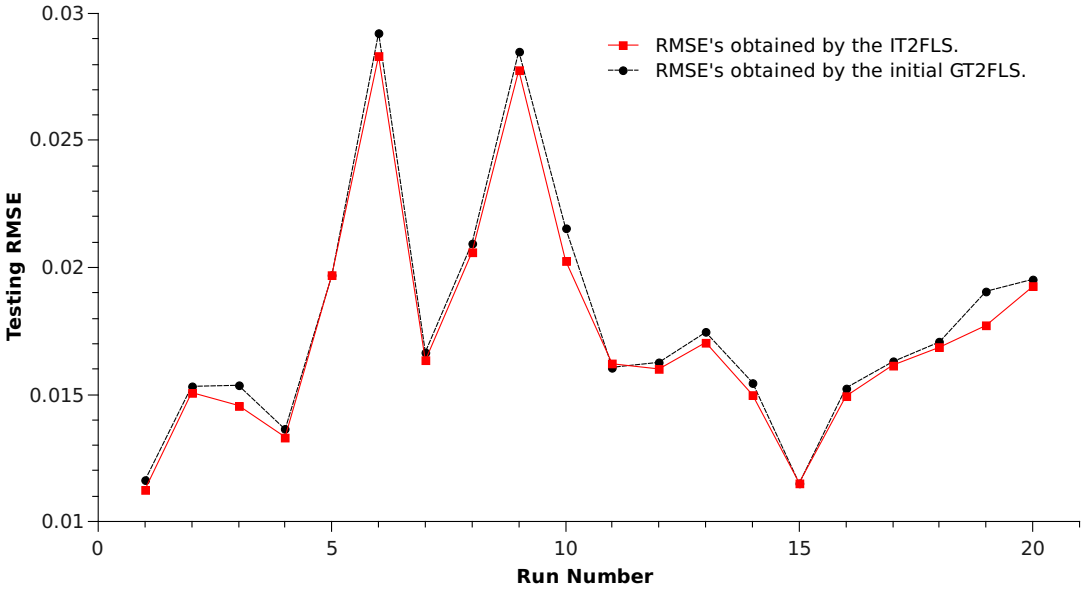

FIGURE 5.5: An example of the results obtained by IT2FLS and the initial GT2FLS for the testing samples in Mackey-Glass time series problem

TABLE 5.4: The forecasting results for noise-free Mackey-Glass time Series by Simulated Annealing with IT2FLS and GT2FLS

Stage	$Mean_{RMSE}$	Std_{RMSE}	$Minimum_{RMSE}$
IT2FLS			
Training	0.013602776	0.004321	0.00899988
Testing	0.017396815	0.004442	0.0112539
Training time	5,233.95	412.341	4,670
One evaluation average time	0.000114	-	-
Initial GT2FLS (after conversion)			
Training	0.01403025	0.004426	0.00874
Testing	0.01782675	0.004627	0.011498
Loss in accuracy (Training)	-3.142550%	-	2.887594%
Loss in accuracy (Testing)	-2.471343%	-	-2.169026%
GT2FLS			
Training	0.013941433	0.0043457	0.00871978
Testing	0.017707055	0.004528	0.011564
Time	8,597.85	258.264	8,137
One evaluation time	0.000373	-	-

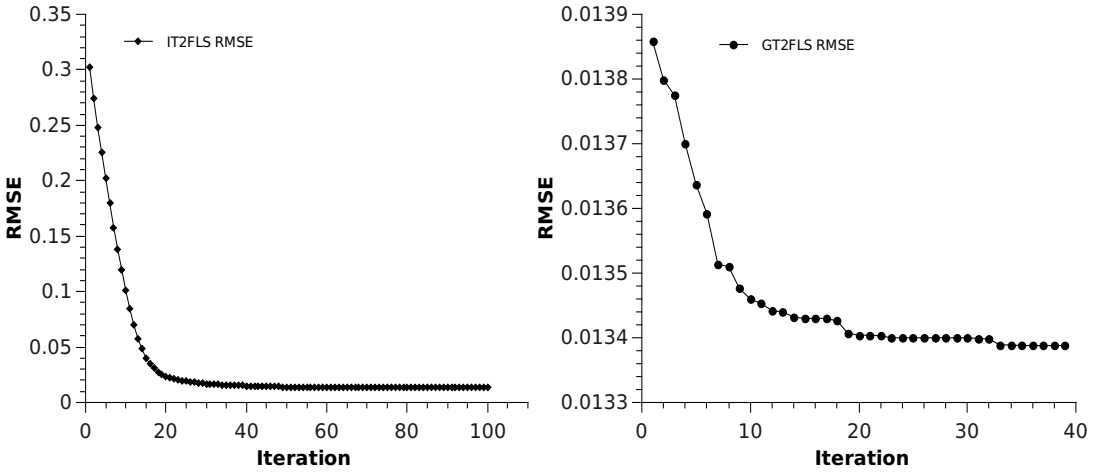

FIGURE 5.6: The average convergence of SA-IT2FLS (left) and GT2FLS (right) for noise-free Mackey-Glass time series problem.

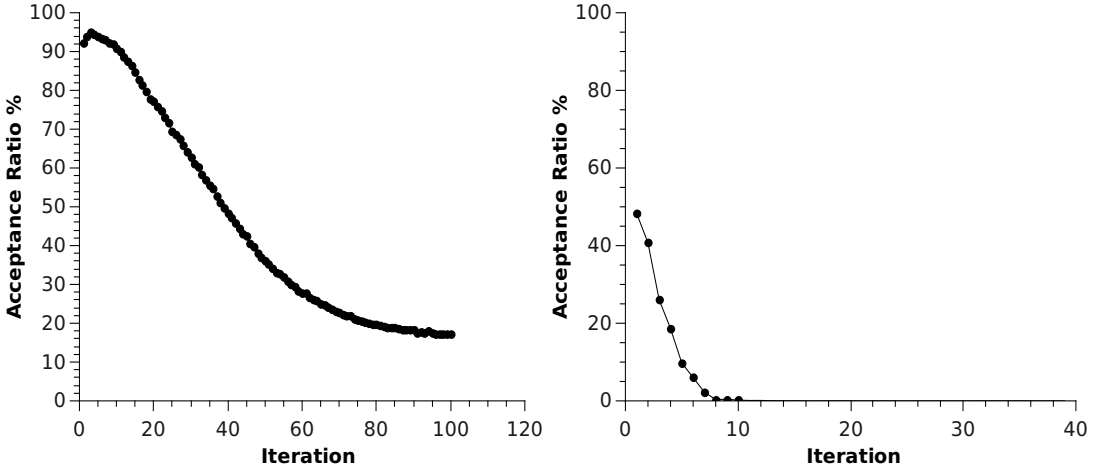

FIGURE 5.7: The average acceptance ratios of SA-IT2FLS (left) and GT2FLS (right) for noise-free Mackey-Glass time series problem.

TABLE 5.5: The forecasting results for Mackey-Glass time series with added noise by simulated annealing with IT2FLS and GT2FLS

Stage	$Mean_{RMSE}$	Std_{RMSE}	$Minimum_{RMSE}$
IT2FLS			
Training	0.10810335	0.002666	0.102858
Testing	0.14300835	0.008324	0.131141
Time	5,575.85	1,094.793	4,677
One evaluation time	0.000121	-	-
Initial GT2FLS (after conversion)			
Training	0.10907035	0.0033947	0.103387
Testing	0.1427435	0.0084819	0.130855
Loss in accuracy (Training)	-0.894514%	-	-0.514301%
Loss in accuracy (Testing)	0.1851989%	-	0.218086%
GT2FLS			
Training	0.10900475	0.003254	0.103773
Testing	0.14293815	0.008373	0.131414
Time	8,104.5	449.260	7,110
One evaluation time	0.000352	-	-

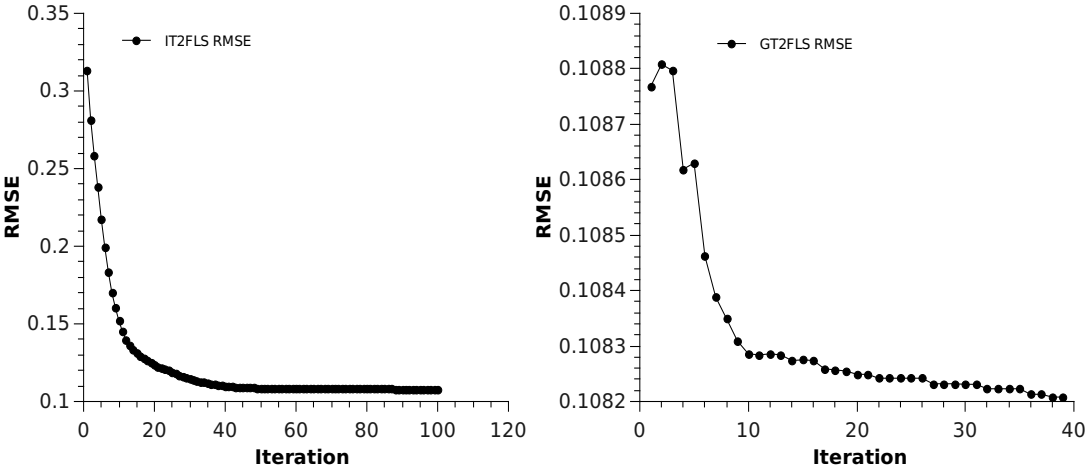

FIGURE 5.8: The average convergence of SA-IT2FLS (left) and GT2FLS (right) for noisy Mackey-Glass time series problem.

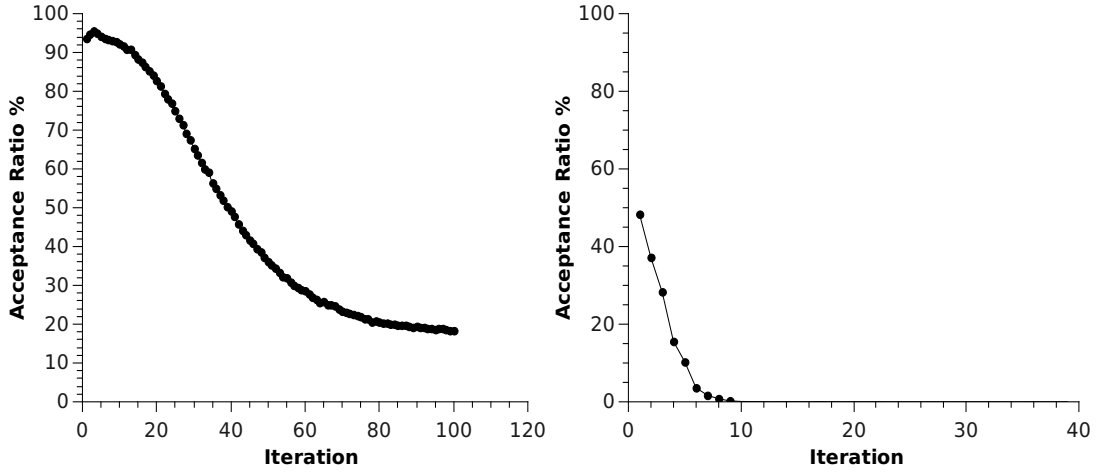

FIGURE 5.9: The average acceptance ratios of SA-IT2FLS (left) and GT2FLS (right) for noisy Mackey-Glass time series problem.

TABLE 5.6: The estimation results for low voltage electrical line length by simulated annealing with IT2FLS and GT2FLS

Stage	$Mean_{RMSE}$	Std_{RMSE}	$Minimum_{RMSE}$
IT2FLS			
Training	576.9342	22.333	541.9515
Testing	593.3093	19.616	570.534
Time	2,134.7	156.7076	1,740
One evaluation time	0.000023	-	-
Initial GT2FLS (after conversion)			
Training	579.0371	24.3067	541.9057
Testing	597.16694	26.9358	570.1701
Loss in accuracy (Training)	-0.364496%	-	0.008451 %
Loss in accuracy (Testing)	-0.650190%	-	0.063782%
GT2FLS			
Training	576.758885	24.415	540.3895
Testing	593.23235	21.8277	570.464
Time	9,772.65	435.998	8,746
One evaluation time	0.000214		

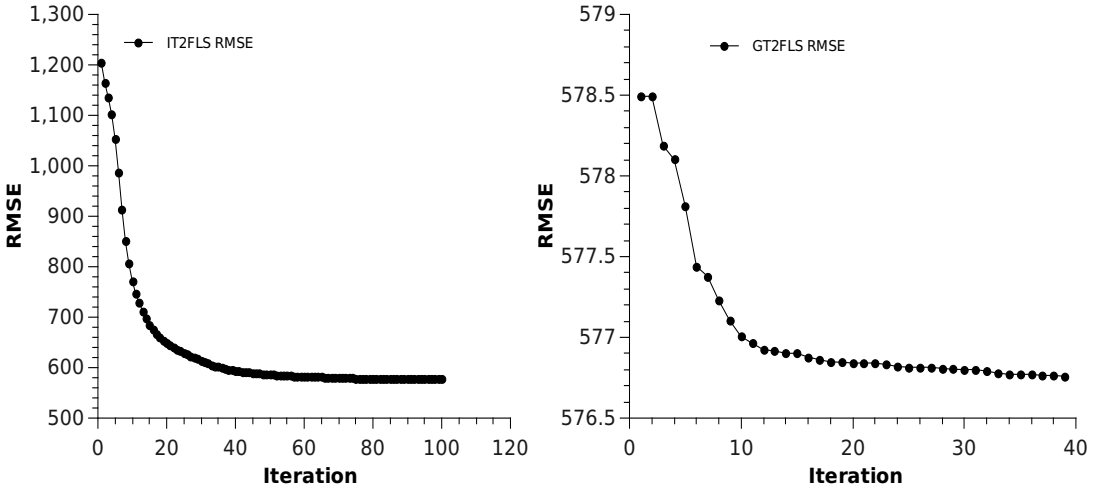

FIGURE 5.10: The average convergence of SA-IT2FLS (left) and GT2FLS (right) for low voltage line problem

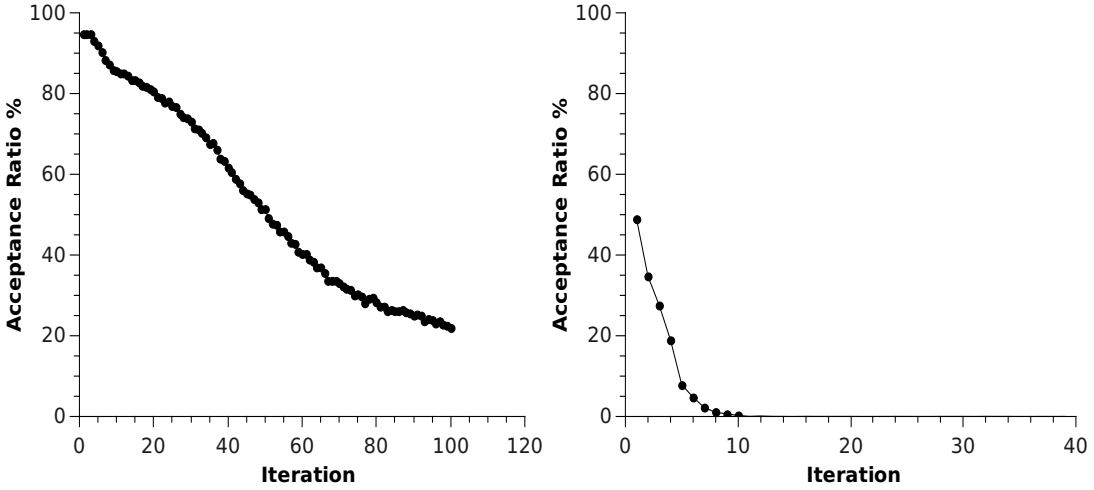

FIGURE 5.11: The average acceptance ratios of SA-IT2FLS (left) and GT2FLS (right) for low voltage line problem

TABLE 5.7: The estimation results for the maintenance cost problem by simulated annealing with IT2FLS and GT2FLS

Stage	$Mean_{RMSE}$	Std_{RMSE}	$Minimum_{RMSE}$
IT2FLS			
Training	116.03324	22.0599	86.4411
Testing	256.6648	68.899	188.883
Time	4,613.5	287.8408	4,042
One evaluation time	0.000100	-	-
Initial GT2FLS (after conversion)			
Training	120.4245465	29.0607	86.45081
Testing	264.132	65.5867	201.2441
Loss in accuracy (Training)	-3.784525%	-	-0.011233%
Loss in accuracy (Testing)	-2.909320%	-	-6.544316%
GT2FLS			
Training	120.030435	29.1178	86.3157
Testing	260.954005	67.0748	197.0781
Time	8,099.3	406.836	7,303
One evaluation time	0.000352	-	-

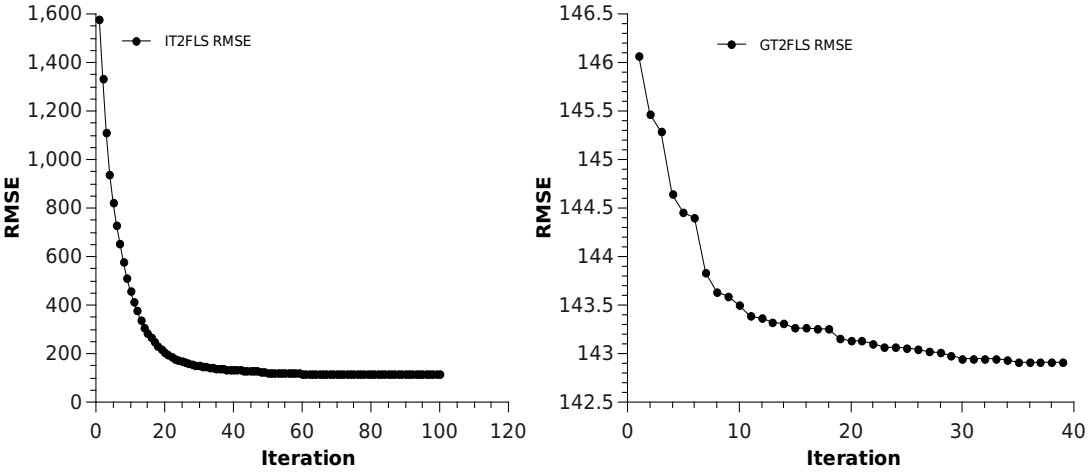

FIGURE 5.12: The average convergence of SA-IT2FLS (left) and GT2FLS (right) for maintenance cost problem

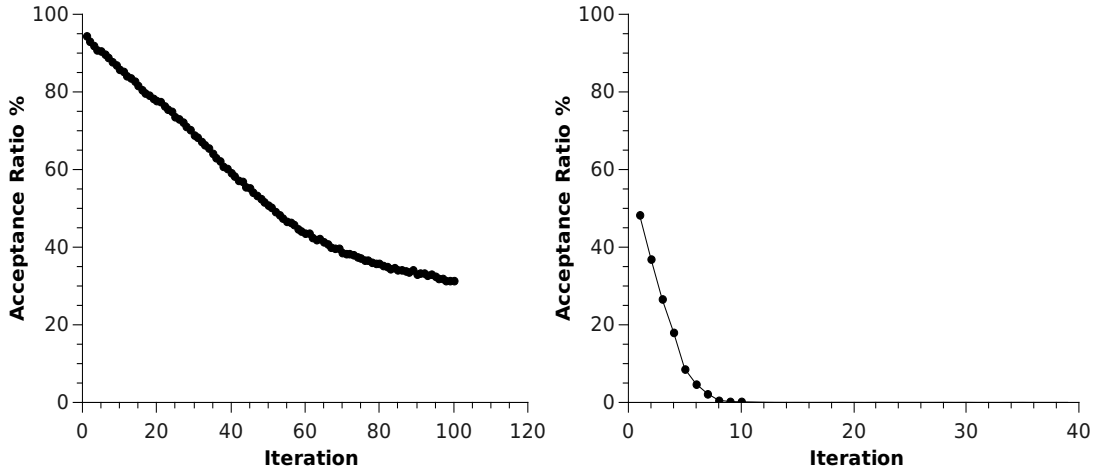

FIGURE 5.13: The average acceptance ratios of SA-IT2FLS (left) and GT2FLS (right) for maintenance cost problem

and reduced the computations time. Although, the comparison between IT2FLS and GT2FLS is not accurate due to the accuracy losses in the conversion stage, the learning of the SMFs brought extra improvements over the initial converted sets which enforces the idea that the third dimension in GT2FLS adds extra ability to model uncertainty over IT2FLS (Mendel and John, 2002) (Christian Wagner, 2009). The next chapter will investigate the design of GT2FLS directly without the need to use IT2FLS.

Chapter 6

Learning of General Type-2 Fuzzy Logic Systems using Simulated Annealing

6.1 Introduction

This chapter reports the work for learning general type-2 fuzzy logic systems using simulated annealing. Unlike the previous chapter, the learning process in this chapter does not depend on optimised interval type-2 fuzzy logic systems to initialise general type-2 fuzzy logic systems. Therefore, the learning process starts from scratch and aims to optimise all parameters involved in the general type-2 fuzzy set in two stages. This is a novel approach as no work has been reported on learning general type-2 fuzzy logic systems using the vertical-slices representation as reported in Chapters 1 and 2. The proposed parameterisation method presented in Chapter 5 has been used to design two models of general type-2 fuzzy logic systems. These two models use two type-reduction techniques. The first is non-deterministic (the sampling method). The second technique

is deterministic (the vertical-slices centroid type-reduction VSCTR). The use of the two type-reduction techniques have been justified in Chapter 5. In addition, both models as well as interval type-2 fuzzy logic system model have been applied to model the four problems presented in previous chapters. The results of modelling these problems using the three models have been analysed and discussed. The question of whether general type-2 fuzzy logic systems can provide abilities to handle more information than interval type-2 fuzzy logic systems has been tackled. The results showed promising results for the general type-2 fuzzy logic systems when using deterministic defuzzification methods.

6.2 The Proposed Method of Learning

Using the proposed form of general type-2 set presented in Chapter 5, we can design GT2FLS using the following two stages procesdure:

- The first step is to design the FOU of the general type-2 set while fixing the secondary membership function. This is done by defining FOU using any function used to define interval type-2 fuzzy sets. The lower and upper membership functions that bound the FOU in interval type-2 fuzzy sets can bound the FOU in general type-2 fuzzy sets. To get a good FOU, experts opinions or automated learning can be applied exactly as the case when dsigning IT2FLS.

- Learning of the secondary membership functions of general type-2 sets. By fixing the optimal FOU, the secondary membership functions can be optimised. This is done by adapting the apex location indicators by a suitable value.

This two stages method seems to be logical as the definition of the uncertainty boundaries (primary memberships) should precedes the definition for how much secondary membership grades (uncertainty distribution) will be given to each primary membership. The other choice is to start by learning the primary and the secondary grades

together but this seems excessive, computationally expensive and might be impossible to have a SMF without defining its FOU.

6.3 Methodology

The whole experiment can be divided into four steps : preparing data, constructing the initial interval and general type-2 fuzzy systems, learning the *FOU* parameters and learning the secondary membership functions.

6.3.1 Data

6.3.1.1 Mackey-Glass time-series

The Mackey-Glass time series is a chaotic time series proposed in (Mackey and Glass, 1977). It has been described in section 3.3.1 and 5.6.1.1. To get the time series, the same configurations used in section 5.6.1.1 were used here. A sample of the noisy data has been shown in last chapter in figure 5.3.

6.3.1.2 Mackey-Glass time-series with added noise

A noisy time series will be used to test our models in this experiment. The noisy Mackey-Glass time series will be generated by adding noise to Mackey-Glass time series as described in section 5.6.1.1. A sample of the noisy data has been shown in last chapter in figure 5.4.

6.3.1.3 Estimation of the low voltage electrical line length in rural towns

This problem has been introduced in section 4.3.1. The data samples were randomly divided into two sets labelled training and testing sets which are randomly selected

from the whole sample as reported in (Cordón, Herrera and Villar, 2001) and (Cordón et al., 2002). As with other authors, 396 samples are used for training while the other 99 samples are used for testing as described in section 5.6.1.3.

6.3.1.4 Estimation of the medium voltage electrical line maintenance cost

This problem has been introduced in section 4.3.1. In order to reduce the training computations and time, the number of samples has been reduced from the one used in Chapter 4. Only 400 data samples from the whole set were divided into two sets labelled training and testing sets with 200 samples for each set. Therefore, 400 samples have been used instead of 1056 samples as described in section 5.6.1.4.

6.3.2 The initial interval type-2 fuzzy logic systems

The fuzzy model consists of a number of independent input fuzzy sets and one independent output fuzzy set for each rule. There are 4 rules while each rule is characterised by a number of fuzzy sets equals to the inputs number (i.e. 4 antecedent fuzzy sets and one consequent fuzzy set). The system is built from scratch rather than using the optimised type-1 sets to initialise the interval type-2 fuzzy sets. Each type-2 fuzzy set is described by Gaussian primary membership functions with uncertain means represented by two means and one standard deviation as follow (Mendel, 2001, p.91):

$$\tilde{f}(x) = \exp^{-(\frac{x-m}{2\sigma})^2} m \in [m_1, m_2] \tag{6.1}$$

Therefore the upper $\overline{\mu}_{\tilde{A}}(x)$ and lower $\underline{\mu}_{\tilde{A}}(x)$ membership functions are defined by following mathematical functions (Mendel, 2001, p.91):

$$\overline{\mu}_{\tilde{A}}(x) = \begin{cases} \exp^{-(\frac{x-m_1}{2\sigma})^2} & \text{if } x < m_1 \\ 1 & \text{if } m_1 \leq x \leq m_2 \\ \exp^{-(\frac{x-m_2}{2\sigma})^2} & \text{if } x > m_2 \end{cases} \tag{6.2}$$

$$\underline{\mu}_{\tilde{A}}(x) = \begin{cases} \exp^{-(\frac{x-m_2}{2\sigma})^2} & \text{if } x \leq \frac{m_1+m_2}{2} \\ \exp^{-(\frac{x-m_1}{2\sigma})^2} & \text{if } x > \frac{m_1+m_2}{2} \end{cases} \tag{6.3}$$

Where the upper $\overline{\mu}_{\tilde{A}}(x)$ and lower $\underline{\mu}_{\tilde{A}}(x)$ membership functions in this equation are used to define FOU_{lower} and FOU_{upper}. All the means and standard deviations are initialised for all the input fuzzy sets by partitioning each input space into the chosen number of fuzzy sets and enabling enough overlapping between them while the output fuzzy sets are initialised randomly around the average value of training outputs. The fuzzification process is based on the minimum t-norm while the centre-of-area has been chosen for type-reduction. The collapsing method proposed by (Greenfield, Chiclana, Coupland and John, 2009) has been used to calculate the centroids of the interval type-2 sets that needed to compute centre-of-area. This is done by using the composite outward right-left variant of the collapsing method as it is described in (Greenfield, Chiclana and John, 2009). The training procedure aims to learn the parameters of the antecedent parts and the consequent parts of the fuzzy system rules. Then, the found parameters are used to predict the next testing data points. The total number of the parameters is $4 * 4 * 3 + 4 * 3 = 60$ in all problems except the line length problem where it is $4 * 2 * 3 + 4 * 3 = 36$ parameters. Hence, only FOU's parameters are optimised in interval type-2 fuzzy sets.

6.3.3 The initial general type-2 fuzzy logic system

The initial general type-2 fuzzy logic system is built by using the proposed parametrisation method described in section 5.3. The fuzzy model consists of a number of independent input fuzzy sets and one independent output fuzzy set for each rule. There are 4 rules while each rule is characterised by a number of antecedent fuzzy sets equals to the inputs number (4 antecedent fuzzy sets and one consequent fuzzy set). However, the number of rules was chosen heuristically and any number of rules can be chosen but we are interested in reducing the system's complexity and saving computations and time. The work in this chapter requires more computations than previous chapter as all FOU's and SMF's parameters will be optimised. Therefore, we reduced the rules number to 4 instead of 8. The system is built from scratch rather than using an optimised type-1 or interval type-2 fuzzy sets to initialise general type-2 fuzzy sets.

6.3.3.1 The General type-2 Sets

The general type-2 sets are defined using its $FOU's$ and $SMF's$ functions as follows:

1. FOU: The same membership functions used to define interval type-2 fuzzy sets in previous subsection (6.3.2) are used to define FOU parameters. The upper $\overline{\mu}_{\tilde{A}}(x)$ and lower $\underline{\mu}_{\tilde{A}}(x)$ membership functions in this equation are used to define FOU_{lower} and FOU_{upper}. All the means and standard deviations are initialised for all the input fuzzy sets by partitioning each input space into the chosen number of fuzzy sets and enabling enough overlapping between them while the output fuzzy sets are initialised randomly around the average value of training outputs.

2. SMF : Our choice for the SMFs in this work is to use a triangular SMF with a normal apex initialised in the middle between (FOU_{lower} and FOU_{upper}) for $k_1, k_2, ...k_n$ points ($n = 9$) by choosing their apexes locations indicators $g(k_1) = g(k_2) = ... = g(k_n) = 0.5$ and then calculating the apex locations for other

x points using the linear interpolation function proposed in section 5.3. This method to parametrise the general type-2 set is shown in figure 5.1.

6.3.3.2 The initial general type-2 fuzzy logic system componenets

The configurations of IT2FLS and GT2FLS used in this experiment are detailed in Table 6.1. The initial general type-2 fuzzy logic system stages will be as follows:

1. **Fuzzification** The fuzzification process will fuzzify each x value into a type-1 fuzzy set (SMF) which is a triangular function as described above. The fuzzified SMF is described by its FOU_{upper} and FOU_{lower} which are derived from the two Gaussian functions for x and its apex location indicator. The output from each fuzzification process is a triangular SMF.

2. **Combination of antecedents** The combination between all antecedent fuzzified values is done using the meet operation proposed by (Coupland and John, 2007) and explained in equation 2.8. The output from this phase is a convex SMF that might be non-normal.

3. **Implication** To do the implication phase, firstly, the consequent sets space is discretised into $n = 101$ points $y_1, y_2, ..., y_n$ in Y domain. Then the implication is done using the same meet operation proposed by (Coupland and John, 2007). The third step is to do a join between all secondary membership grades for each $y \in Y$ using the join operation proposed in (Coupland and John, 2007) and explained in equation 2.3.4.

4. **Type-Reduction** Two methods for type-reduction have been used. The embedded sets based sampling method and VSCTR method. In sampling method, we used 100 samples of the embedded sets. The rationale for using two type-reduction methods is to test the true effects of learning SMF in general type-2

TABLE 6.1: The configurations of IT2FLS and GT2FLS used in this experiment

Stage	**IT2FLS**	**GT2FLS**
Membership Function	Gaussian	Gaussian + triangular SMF
Number of parameters (with four inputs)	60	60+180=240
fuzzification	singleton	singleton
Antecedent combination t-norm	minimum	minimum using Coupland's meet
Implication t-norm	minimum	minimum using Coupland's meet
Join t-conorm	maximum	maximum using Coupland's join
SMF discretised points	none	9
Type-reduction method	centroid by collapsing method	centroid by sampling and VSCTR
Defuzzification method	centroid	centroid
Y Descritisation points	101	101

fuzzy sets without been distracted by the stochastic evaluation using sampling. The output from this phase is a type-1 fuzzy set.

5. **Defuzzification** The centre of area (centroid) defuzzification has been used in this part.

6.3.4 Learning of FOU parameters

The training procedure aims to get the best parameters of the antecedent parts and the consequent parts of the fuzzy system rules. Then, the found parameters are used to predict the next testing data points. The total number of $FOUs$ parameters is $4*4*3+4*3 = 60$ in all problems except the line length problem where it is $4*2*3+4*3 = 36$ parameters. The learning process is done using simulated annealing algorithm that searches for the best configuration of the parameters by trying to modify one parameter each time and evaluate the cost of the new state which is measured by a cost function which is Root Mean Square Error (RMSE) that is defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{k=1}^{n}[f(k) - f(k^*)]^2} \tag{6.4}$$

From an optimisation perspective, the only constraint to the variables of the optimisation problem is that all standard deviations of all fuzzy sets must be ≥ 0. The simulated

annealing algorithm is initialised with a temperature equals to the standard deviation of the mean RMSE's for 1000 runs for the training samples as proposed by (White, 1984). The cooling schedule is based on a static cooling rate of 0.9 updated for each Markov chain. Each Markov chain has a length related to the number of variables in the search space which equals to 5 times the number of variables. The search ends after 40 Markov chains. The new states for a current state are chosen from neighbouring states randomly by adding a small number (step size) to one of the antecedent parameters or the consequent parameters. The step size value is related to the maximum and minimum value for each input space and = max-min/25 while the direction of the search is chosen randomly. After that, the new state is evaluated by examining the 200 data points outputs. Then, the average and the minimum of the cost function of the training and testing results have been calculated.

6.3.5 The learning of the secondary membership functions

The learning process in this stage aims to get the optimal locations of apexes for all the SMF's parameters where the other two points for each triangular SMF are fixed. The optimised parameters in this case are the apexes locations factors $g(k_1), g(k_2), ..., g(k_n)$ for each general type-2 set involved in the system. The learning is done using simulated annealing algorithm with the same configuration used above apart from the following :

1. The constraints for each variable (apexes factors $g(k_i)$ for each k_i) are defined by their ($FOU_{lower}(k_i)$ and $FOU_{upper}(k_i)$) points which constitue the secondary domain boundaries for each k_i.

2. The step size is the value that changes the apex location indicator. The new value must be between $[0, 1]$ and the step size should be large enough to make a difference in the cost function as small values might not change the outputs when it does not overcome the next discretisation step in the secondary domain. The chosen step size is 0.225.

3. The length of each Markov chain is equal to 5 times the number of variables in the search space. The search ends after 10 Markov chains. These choices to reduce the experiment's time.

The number of all parameters being optimised in this stage for each fuzzy set is $n = 9$. Therefore, the total number of all parameters being optimised in the system in this stage is (the number of fuzzy sets $* n$) parameters. That is $4 * 5 * 9 = 180$ parameters in problems with 4 inputs and $4 * 3 * 9 = 108$ parameters in the length line problem. The experiment has been carried out 20 times and the average and the minimum of the cost function of the testing data results have been calculated.

6.4 Results and Discussion

The experiments were developed using C++ language and have been carried out 20 times on a number of PCs with an equal CPU speed of 3 GH's and a memory of 4GB. Results are shown for each problem in a number of points in next paragraphs. The results are summarised for all problems in table 6.6. Extra insights into the convergence behaviours and acceptance ratios in both stages will be discussed to explain some new results.

6.4.1 Mackey-Glass time series results

The results of learning Mackey-Glass time series are detailed in table 6.2 where the average RMSEs curves and the acceptance ratios during search are depicted in figures 6.1 and 6.2 respectively. The main observations are :

1. The best average RMSE in testing samples was obtained by general type-2 fuzzy logic system with VSCTR defuzzification (GT2FLS-VSCTR) followed by interval type-2 fuzzy logic system (IT2FLS).

2. The best average RMSE in training samples was obtained by general type-2 fuzzy logic system with VSCTR defuzzification followed by IT2FLS.

3. The average RMSEs curves when learning FOUs (training samples) have exhibited similar performances by the three models. However, IT2FLS obtained the best average RMSEs in testing phase followed by GT2FLS-VSCTR which was the best in training phase followed by IT2FLS.

4. The learning of SMFs using GT2FLS-VSCTR adds about 11.7% to the average testing RMSEs and about 17.7% to the average training RMSEs over the FOU's learning best results. The learning of SMFs using GT2FLS-Sampling adds about 0.86% to the training RMSEs but worsened the testing RMSEs by about -0.059.

5. The learning curves of SMFs showed a clear difference in performance between GT2FLS-VSCTR and GT2FLS-Sampling models. GT2FLS-VSCTR shows continuous improvements compared to very small improvements obtained by GT2FLS-Sampling.

6. The acceptance ratios curves when learning FOUs show similar behaviours between GT2FLS-VSCTR and IT2FLS better than the narrower acceptance behaviour obtained by GT2FLS-Sampling. The last one shows unpleasant behaviour where it converges to values close to 0% quickly in less than 30 Markov chains which means no improvements were observed in the rest of iterations.

7. The acceptance ratios curves when learning SMFs show a clear difference in behaviours between GT2FLS-VSCTR and GT2FLS-Sampling models. The GT2FLS-Sampling shows unpleasant very narrow acceptance behaviour compared to GT2FLS-VSCTR. Interestingly, the acceptance ratios curves of GT2FLS-Sampling model show narrower behaviour when learning SMFs from its behaviour with FOUs. However, as mentioned above, the initial temperatures were set separately in each stage to be proportional to the objective function differences brought by

these moves in the two parameters groups (FOU and SMF). This is important to avoid starting with very large or very small initial temperatures and to have acceptable curves of best results and acceptance ratios. In other words, the observed acceptance behaviours for GT2FLS-Sampling model are not related to the settings of simulated annealing. This behaviour can be easily explained by the effects of the defuzzification method which is the only difference between the two models of GT2FLS. As explained in section 5.5, the effects of the stochastic objective function when using sampling method can be ignored when moves from a state to a state can bring relatively large differences compared to the random noise but this noise can deteriorate the search when moves bring improvements comparable to that noise. In other words, when learning FOU, the differences brought by moves are large enough to accept very small errors of approximated objective functions due to the larger contributions of FOU's parameters on the objective functions compared to the SMF contributions. Hence, we do not expect large contribution from learning SMF's parameters compared to FOU's learning due to the fact that SMF is dependent on FOU and bounded by its endpoints. This behaviour of acceptance ratios when using GT2FLS-Sampling have been observed with all problems and this explanation is applied to them.

8. The time taken by IT2FLS was the shortest by 5.8 times faster than GT2FLS-VSCTR and by 21.8 times faster than GT2FLS-Sampling. Therefore, IT2FLS is preferred in terms of speed.

6.4.2 Mackey-Glass time series with added noise results

The results of learning Mackey-Glass time series with added noise are detailed in table 6.3 where the average RMSE's curves and the acceptance ratios during search are depicted in figures 6.3 and 6.4 respectively. The main observations from the results are :

TABLE 6.2: The forecasting results for noise-free Mackey-Glass time series by simulated annealing with GT2FLS

Stage	$Mean_{RMSE}$	Std_{RMSE}	$Minimum_{RMSE}$
IT2FLS			
Training	0.04980955	0.0200348	0.026242
Testing	0.0433439	0.010239	0.027117
Time	332.55	21.027488	295
GT2FLS with Sampling Defuzzification			
After FOU's Learning			
Training	0.0553228125	0.01243	0.03761027
Testing	0.0518645455	0.0107249	0.03617023
After SMF's Learning			
Training	0.0548446	0.0119293	0.0372725
Improvement by SMF	0.86%	-	-
Testing	0.051895285	0.010721	0.0362123
Improvement by SMF	-0.059269 %	-	-
Time	7,259.9	992.126	5,724
GT2FLS with VSCTR Defuzzification			
After FOU's Learning			
Training	0.0483079765	0.01089	0.03428513
Testing	0.0446682685	0.0121448	0.02823214
After SMF's Learning			
Training	0.03975027	0.0115896	0.0240021
Improvement by SMF	17.7%	-	-
Testing	0.03943346	0.0116557	0.024325
Improvement by SMF	11.7%	-	-
Time	1,945.45	368.392	1,217

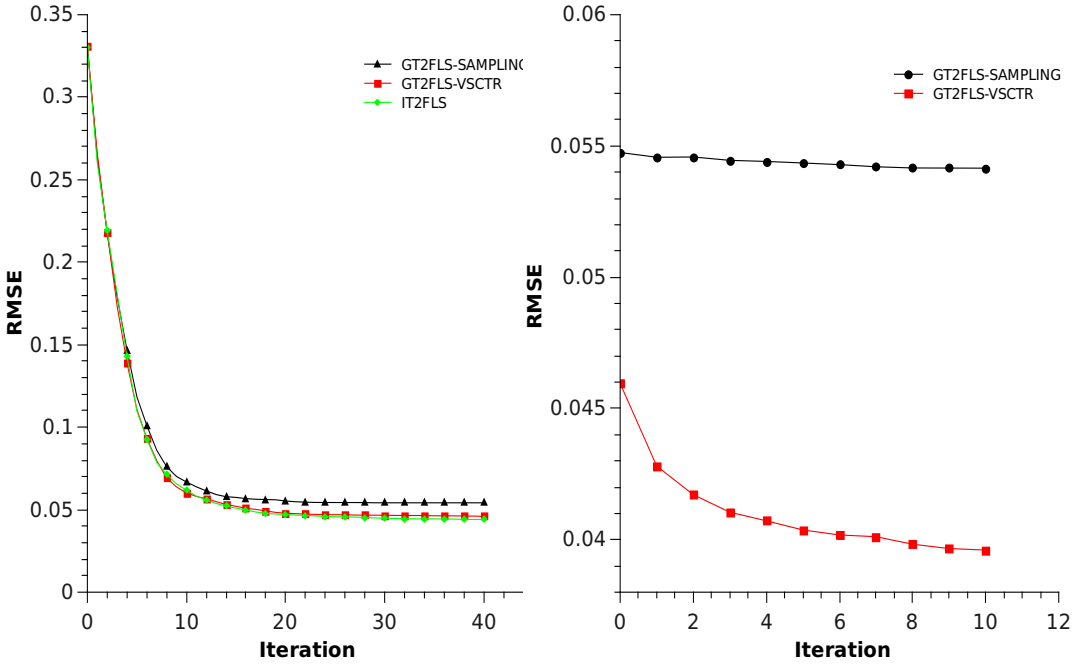

FIGURE 6.1: The average convergence of the three models for noise-free Mackey-Glass time series problem when learning FOU (left) and SMF (right).

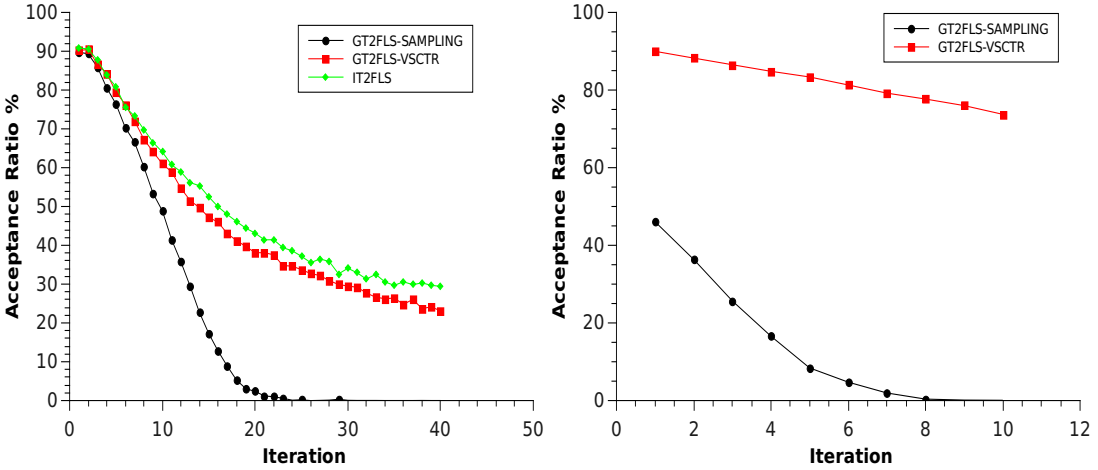

FIGURE 6.2: The average acceptance ratios of the three models when learning FOU (left) and SMF (right) for noise-free Mackey-Glass time series problem.

1. The best average RMSE in testing samples was obtained by GT2FLS-VSCTR followed by GT2FLS-Sampling.

2. The best average RMSE in training samples was obtained by GT2FLS-VSCTR followed by GT2FLS-Sampling.

3. The average RMSE's curves for learning FOUs (training samples) have exhibited similar performances by the three models. However, GT2FLS-VSCTR model obtained best average RMSEs in training and testing followed by GT2FLS-Sampling.

4. The learning of SMFs using GT2FLS-VSCTR adds about 1% to the average testing RMSEs and about 3% to the average training RMSEs over the FOU's learning best results. The learning of SMFs using GT2FLS-Sampling adds about 0.32% to the training RMSEs but worsened the testing RMSEs by about -0.1.

5. The learning curves of SMFs showed a clear difference in performance between GT2FLS-VSCTR and GT2FLS-Sampling models. GT2FLS-VSCTR shows continuous improvements compared to relatively small improvements obtained by GT2FLS-Sampling.

6. The acceptance ratios curves when learning FOUs show similar behaviours between GT2FLS-VSCTR and IT2FLS better than the narrower acceptance behaviour obtained by GT2FLS-Sampling. The last one converges to acceptance ratios close to 0% in less than 25 Markov chains.

7. The acceptance ratios curves when learning SMFs show a clear difference in behaviours between GT2FLS-VSCTR and GT2FLS-Sampling models. The GT2FLS-Sampling shows very narrow acceptance behaviour compared to GT2FLS-VSCTR.

8. The time taken by IT2FLS was the shortest by 6.3 times faster than GT2FLS-VSCTR and by 20 times faster than GT2FLS-Sampling.

TABLE 6.3: The forecasting results for Mackey-Glass time series with added noise by simulated annealing with IT2FLS and GT2FLS

Stage	$Mean_{RMSE}$	Std_{RMSE}	$Minimum_{RMSE}$
IT2FLS			
Training	0.1468528	0.02459	0.125778
Testing	0.1525942	0.014847	0.126217
Time	350	57.217	285
GT2FLS with Sampling Defuzzification			
After FOU's Learning			
Training	0.13835616	0.00835	0.1282838
Testing	0.14152867	0.008688	0.1242401
After SMF's Learning			
Training	0.13790745	0.008148	0.128318
Improvement by SMF	0.32%	-	-
Testing	0.1416725	0.0086397	0.124408
Improvement by SMF	-0.1%	-	-
Time	6,999.45	1,107.276	4,645
GT2FLS with VSCTR Defuzzification			
After FOU's Learning			
Training	0.132636905	0.0045462	0.123189
Testing	0.138335225	0.004243	0.1287115
After SMF's Learning			
Training	0.12860905	0.004316	0.120457
Improvement by SMF	3%	-	-
Testing	0.136965	0.0043126	0.128493
Improvement by SMF	1%	-	-
Time	2,197.5	426.4706	1,460

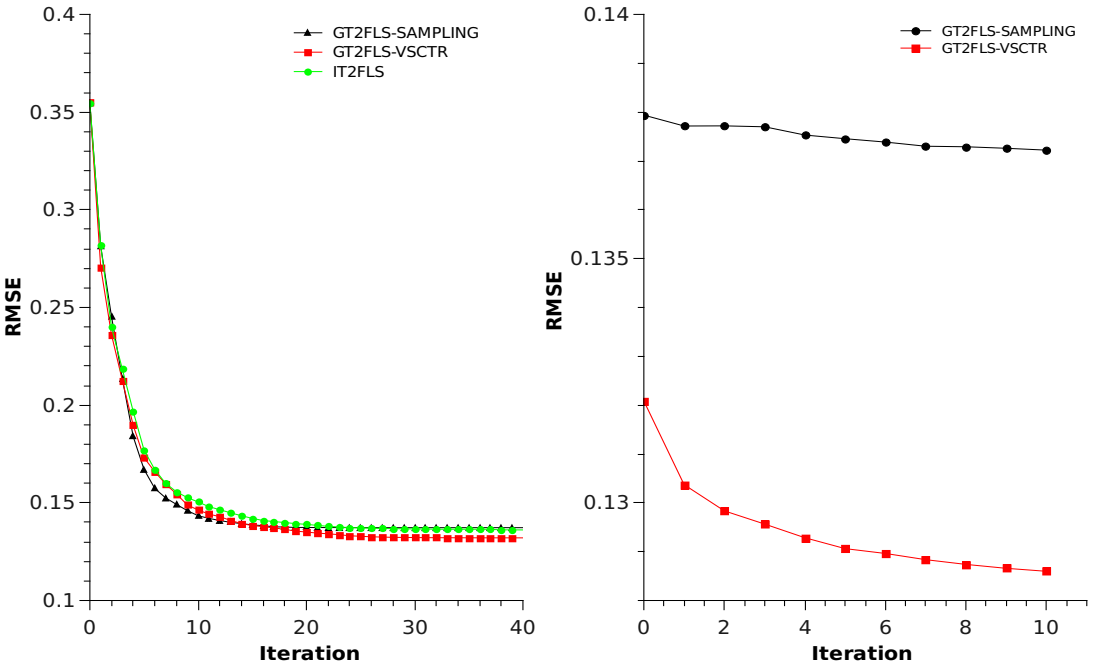

FIGURE 6.3: The average convergence of the three models for noise-free Mackey-Glass time series with added noise problem when learning FOU (left) and SMF (right).

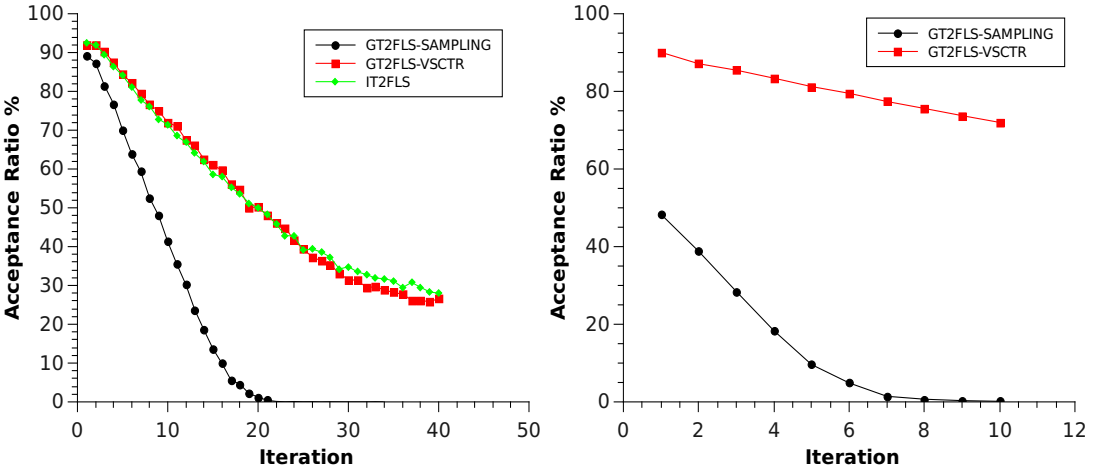

FIGURE 6.4: The average acceptance ratios of the three models when learning FOU (left) and SMF (right) for noisy Mackey-Glass time series problem.

6.4.3 The low voltage electrical line length results

The results of the learning low voltage electrical line length problem are detailed in table 6.4 where the average RMSEs curves and the acceptance ratios during search are depicted in figures 6.5 and 6.6 respectively. The main observations from the results are :

1. The best average RMSE in testing samples was obtained by GT2FLS-VSCTR followed by GT2FLS-Sampling.

2. The best average RMSE in training samples was obtained by GT2FLS-VSCTR followed by GT2FLS-Sampling.

3. The average RMSE's curves for learning FOUs (training samples) have exhibited similar performances by the three models. However, GT2FLS-VSCTR model obtained best average RMSEs in training and testing. The second in testing was GT2FLS-Sampling while IT2FLS was the second in training.

4. The learning of SMFs using GT2FLS-VSCTR adds about 0.88% to the average testing RMSEs and about 4.9% to the average training RMSEs over the FOU's learning best results. The learning of SMFs using GT2FLS-Sampling adds about 0.05% to the testing RMSEs and about 3.15% to the training RMSEs after FOU's learning.

5. The learning curves of SMFs showed a clear difference in performance between GT2FLS-VSCTR and GT2FLS-Sampling models. GT2FLS-VSCTR shows continuous improvements compared to relatively small improvements obtained by GT2FLS-Sampling.

6. The acceptance ratios curves when learning FOUs show similar behaviours between GT2FLS-VSCTR and IT2FLS better than acceptance behaviour obtained

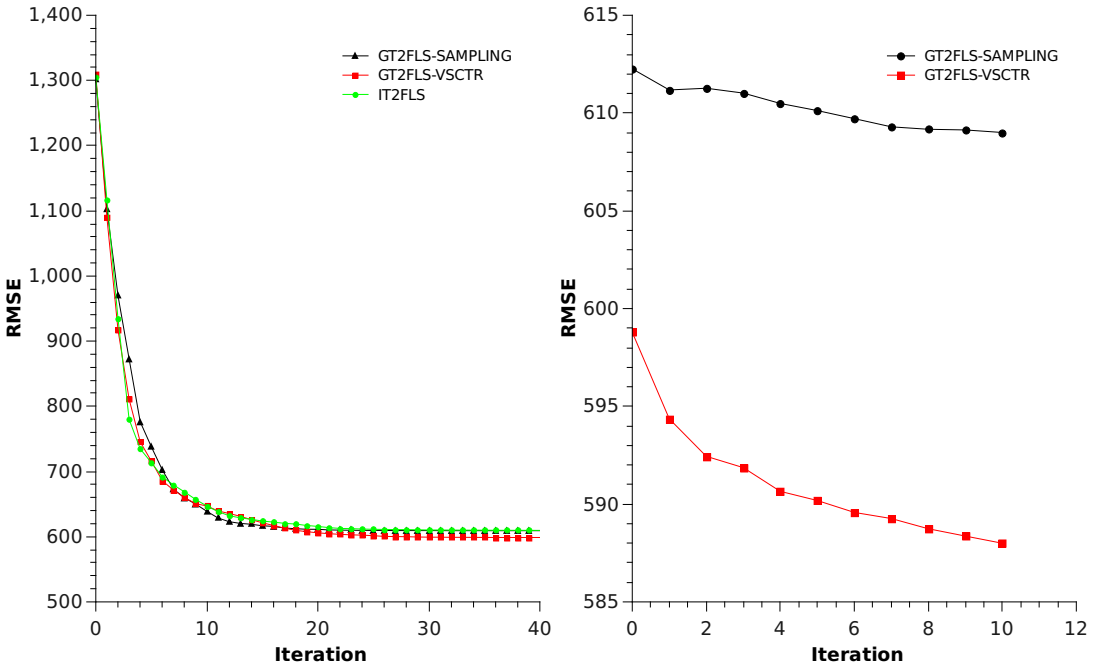

FIGURE 6.5: The average convergence of the three models for low voltage electrical line length problem when learning FOU (left) and SMF (right).

by GT2FLS-Sampling. The last one converges to acceptance ratios close to 0% in less than 25 Markov chains.

7. The acceptance ratios curves when learning SMFs show a clear difference in behaviours between GT2FLS-VSCTR and GT2FLS-Sampling models. The GT2FLS-Sampling shows very narrow acceptance behaviour compared to GT2FLS-VSCTR.

8. The time taken by IT2FLS was the shortest by 3.8 times faster than GT2FLS-VSCTR and by 17.3 times faster than GT2FLS-Sampling.

6.4.4 The maintenance cost problem results

The results of learning the maintenance cost problem are detailed in table 6.5 where the average RMSE's curves and the acceptance ratios during search are depicted in figures 6.7 and 6.8 respectively. The main observations are :

TABLE 6.4: The estimation results for low voltage electrical line length by simulated annealing with IT2FLS and GT2FLS

Stage	$Mean_{RMSE}$	Std_{RMSE}	$Minimum_{RMSE}$
IT2FLS			
Training	627.816	64.10956	580.319
Testing	606.84075	62.6282	568.15
Time	530.8	47.5987	463
GT2FLS with Sampling Defuzzification			
After FOU's Learning			
Training	632.080425	50.4127	595.8498
Testing	594.33905	16.46317	562.3377
After SMF's Learning			
Training	612.13475	10.24457	593.864
Improvement by SMF	3.15%	-	-
Testing	594.02365	16.2959	560.929
Improvement by SMF	0.05%	-	-
Time	9,162.3	2,521.752	4,377
GT2FLS with VSCTR Defuzzification			
After FOU's Learning			
Training	618.412695	52.09535	577.1892
Testing	596.185465	19.2559	571.3894
After SMF's Learning			
Training	588.01895	11.6174	564.773
Improvement by SMF	4.9%	-	-
Testing	590.90565	18.40509	559.914
Improvement by SMF	0.88%	-	-
Time	2,005.65	367.299	1,474

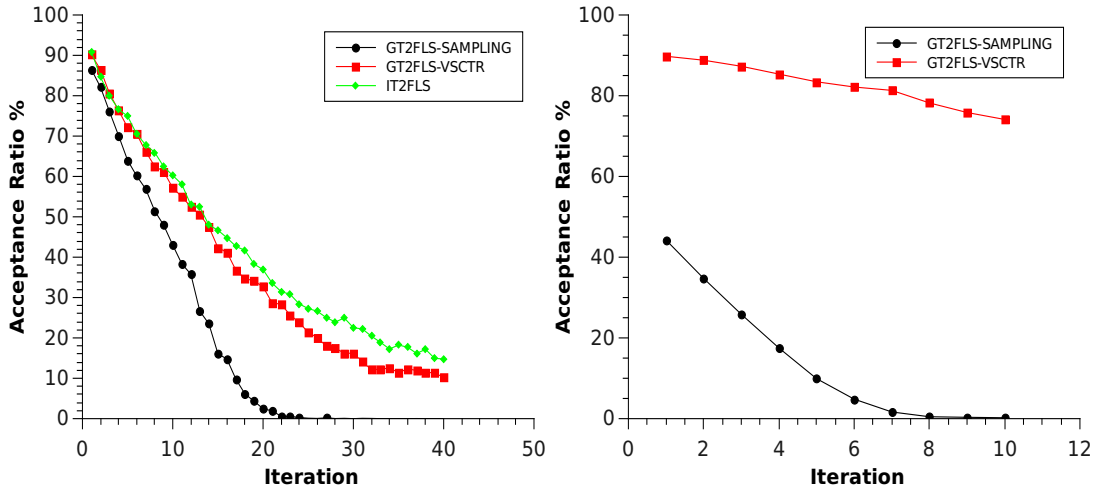

FIGURE 6.6: The average acceptance ratios of the three models when learning FOU (left) and SMF (right) for low voltage line problem

1. The best average RMSE in testing samples was obtained by GT2FLS-VSCTR followed by interval type-2 fuzzy logic system.

2. The best average RMSE in training samples was obtained by GT2FLS-VSCTR followed by interval type-2 fuzzy logic system.

3. The average RMSE's curves for learning FOUs (training samples) have exhibited similar performances by the three models. Again, GT2FLS-VSCTR model obtained best average RMSEs in both training and testing followed by IT2FLS.

4. The learning of SMFs using GT2FLS-VSCTR adds about 6.9% to the average testing RMSEs and about 14.9% to the average training RMSEs over the FOU's learning best results. The learning of SMFs using GT2FLS-Sampling adds about 3.35% to the training RMSEs but worsened the testing RMSEs by about -0.05%.

5. The learning curves of SMFs showed a clear difference in performance between GT2FLS-VSCTR and GT2FLS-Sampling models. GT2FLS-VSCTR shows continuous improvements compared to relatively very small improvements obtained by GT2FLS-Sampling.

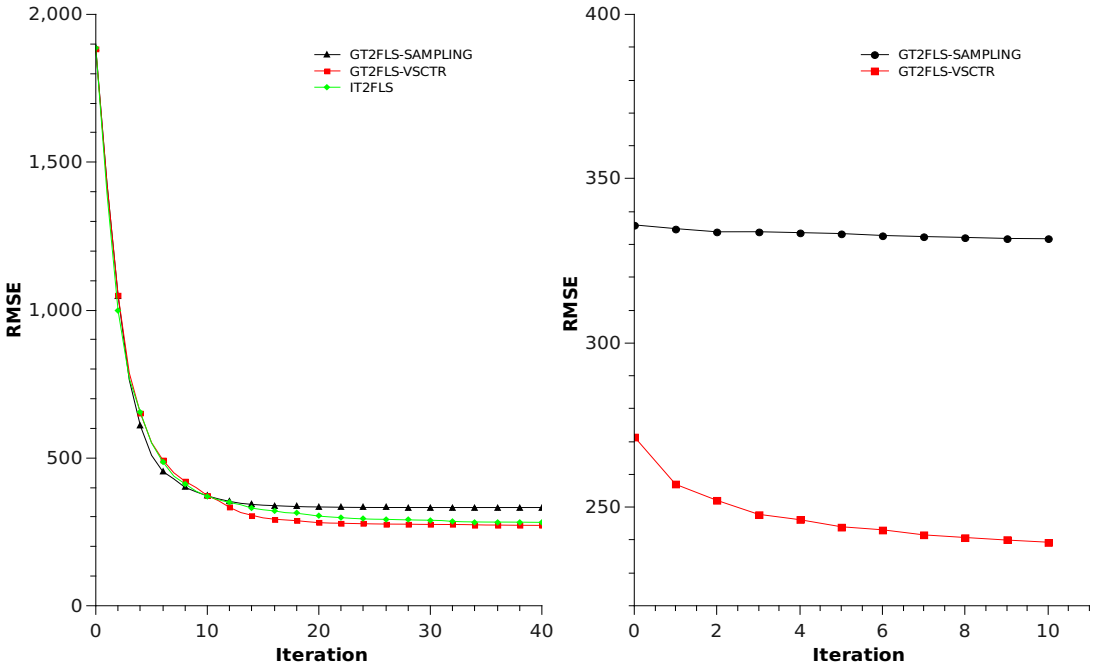

FIGURE 6.7: The average convergence of the three models for the maintenance cost problem when learning FOU (left) and SMF (right).

6. The acceptance ratios curves when learning FOUs show similar behaviours between GT2FLS-VSCTR and IT2FLS better than acceptance behaviour obtained by GT2FLS-Sampling. The last one converges to acceptance ratios close to 0% in less than 25 Markov chains.

7. The acceptance ratios curves when learning SMFs show a clear difference in behaviours between GT2FLS-VSCTR and GT2FLS-Sampling models. The GT2FLS-Sampling shows very narrow acceptance behaviour compared to GT2FLS-VSCTR.

8. The time taken by IT2FLS was the shortest by 3.77 times faster than GT2FLS-VSCTR and by 14.4 times faster than GT2FLS-Sampling.

6.4.5 Results summary

The main conclusions from the results for the four problems are :

TABLE 6.5: The estimation results for the maintenance cost problem by simulated annealing with IT2FLS and GT2FLS

Stage	$Mean_{RMSE}$	Std_{RMSE}	$Minimum_{RMSE}$
IT2FLS			
Training	304.8366	92.320619	145.985
Testing	353.99755	106.36379	207.672
Time	410.95	44.00535	341
GT2FLS with Sampling Defuzzification			
After FOU's Learning			
Training	347.56075	93.88004	172.6323
Testing	424.139295	107.8478	230.5775
After SMF's Learning			
Training	335.91655	81.6298	172.514
Improvement by SMF	3.35%	-	-
Testing	424.3692	108.0096	229.275
Improvement by SMF	-0.05%	-	-
Time	5,936.6	937.4517	4,037
GT2FLS with VSCTR Defuzzification			
After FOU's Learning			
Training	281.416145	96.938	124.3031
Testing	341.1021	112.5648	155.5801
After SMF's Learning			
Training	239.4284	76.2998	109.223
Improvement by SMF	14.9%	-	-
Testing	317.43325	104.8642	145.222
Improvement by SMF	6.9%	-	-
Time	1,556.25	183.7191	1,260

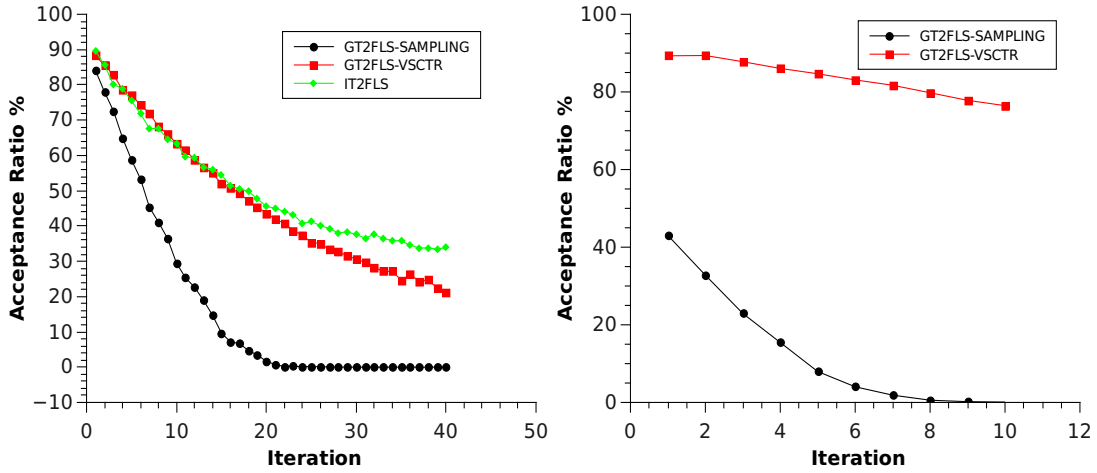

FIGURE 6.8: The average acceptance ratios of the three models when learning FOU (left) and SMF (right) for maintenance cost problem

1. GT2FLS-VSCTR model obtained the best results in all cases for both training and testing results (average RMSEs).

2. GT2FLS-Sampling and IT2FLS were overlapping the second position and therefore showing similar results. However, the learning of SMF when using sampling defuzzification was distracted by the stochastic behaviour for the objective function. In fact, it is unfair to compare a model with a stochastic evaluation (GT2FLS-Sampling) with a model with a deterministic evaluation (IT2FLS) of the objective function. Therefore, the question of weather general type-2 fuzzy logic systems or IT2FLS is better in handling uncertainties should not be based on such case.

3. When learning FOU, GT2FLS-VSCTR obtained the best results in all training cases and most testing cases (three out of four) against IT2FLS. This might be explained by the common sense that the uncertainties in practice are centred and distributed around some points in the SMF. In interval type-2 fuzzy set, all uncertainties are given the same amount of possibilities in their SMF. Therefore,

in general, general type-2 fuzzy sets should be able in practice to handle more information than interval type-2 fuzzy sets even without learning their SMF.

4. The learning of SMFs using GT2FLS-VSCTR adds between 0.88% and 11.7% to the average testing RMSEs and between 3% to 17.7% to the average training RMSEs over the FOU's learning best results. The learning of SMFs using GT2FLS-Sampling adds between 0.32% and 3.35% to the training RMSEs and up to 0.5% to the testing RMSEs but also worsen some results by up to -1%. Again, the comparison against GT2FLS-Sampling should not be tackled between such models. In other words, the learning of SMFs has brought a noticeable improvements when allowing deterministic method of evaluate objective functions meaning that general type-2 fuzzy logic systems add more abilities and flexibilities to modelling than interval type-2 fuzzy logic systems. The problem of type-reduction in general type-2 fuzzy logic systems should be more investigated to find embedded sets based, practical and stable methods to help designing general type-2 fuzzy logic systems automatically.

5. The stochastic evaluations of centroids using sampling method affects the learning performance of general type-2 fuzzy logic systems especially when learning SMF. These effects are shown through their learning and acceptance behaviours curves.

6. The time taken by interval type-2 fuzzy logic system was the shortest by $3.77-6.3$ times faster than GT2FLS-VSCTR and by $14.4-21.8$ times faster than GT2FLS-Sampling. Therefore, in terms of speed, IT2FLS is preferred followed by GT2FLS-VSCTR.

6.5 Summary

The work for learning general type-2 fuzzy logic systems using simulated annealing has been reported. The learning to configure all general type-2 fuzzy logic systems

TABLE 6.6: The results summary for all problem by the three models to model testing samples ordered by their accuracies (1= the best)

Model	Mackey-Glass		Noisy Mackey-Glass		Line length		Maintenance cost	
	Training	Testing	Training	Testing	Training	Testing	Training	Testing
IT2FLS	2	2	3	3	3	3	2	2
GT2FLS-Sampling	3	3	2	2	2	2	3	3
GT2FLS-VSCTR	1	1	1	1	1	1	1	1

parameters in two stages has been applied in both FOU and SMF parts. The learning process starts from scratch rather than using optimised interval type-2 fuzzy logic systems to initialise general type-2 fuzzy logic systems. The novel parametrisation approach presented in Chapter 5 has been used to design two models of general type-2 fuzzy logic systems. These two models using two type-reduction techniques; one is non-deterministic (the sampling method). The other technique is deterministic (the vertical-slices centroid type-reduction VSCTR). The rationale for using the two type-reduction techniques have been described. In addition, both models as well as interval type-2 fuzzy logic system model have been applied to model the four problems presented in previous chapters. The question of weather general type-2 fuzzy logic systems can provide more abilities to handle information has been tackled. The stochastic defuzzification method of sampling embedded sets affects the learning performance in both FOU and SMF learning stages. However, general type-2 fuzzy logic systems with sampling defuzzification have achieved similar performance to interval type-2 fuzzy logic systems but in a very long time. The best results achieved in all problems have been accredited to general type-2 fuzzy logic systems with VSCTR defuzzification. The results showed that when using the deterministic defuzzification method (VSCTR), the learning of general type-2 fuzzy logic systems can provide extra abilities to handle more information and uncertainties than interval type-2 fuzzy logic systems that use uniform SMF's. Although, the use of VSCTR is not based on the concept of using embedded sets to calculate the exact centroids of type-2 sets, the method allows the learning process to be carried out in a practical manner. This achievement opens the doors to using other learning methods to get more modelling capabilities from the general type-2 fuzzy logic

systems in real-world.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Through this thesis, the combination of simulated annealing with type-2 fuzzy logic systems has been proposed and examined for modelling uncertainty. The interval and generalised models of type-2 fuzzy logic systems have been used as well as using type-1 fuzzy logic systems model. The three models have been analysed in their abilities to handle information in problems with associated uncertainties. The main conclusions are :

- Simulated annealing exhibited a good candidate for fuzzy logic system amongst other combinations.
- Interval type-2 fuzzy logic systems added more abilities to modelling information and handling uncertainties than type-1 fuzzy logic systems but require more computations and time.
- The learning of general type-2 fuzzy logic systems can provide extra abilities to handle more information and uncertainties than interval type-2 fuzzy logic

systems especially when using a deterministic defuzzification method. This is a result of exploiting the freedom in the third dimension in general type-2 fuzzy sets.

After the introduction and the literature review in Chapters 1 and 2, the simulated annealing algorithm was applied with variants of type-1 fuzzy logic system models and fuzzification methods in Chapter 3. Two models (Mamdani and TSK) models and two fuzzification methods (singleton and non-singleton) methods have been applied to forecast Mackey-Glass time series with different levels of added noise. This was presented in the third chapter alongside a discussion on issues related to this combination. The main conclusions from this work is :

- Simulated annealing exhibited a good candidate for fuzzy logic systems among other combinations. Its ability to handle different kinds of functions and to suit higher dimensionality allows simulated annealing to be a good choice for this purpose.
- In higher noise levels, the Mamdani model with non-singleton fuzzification showed more modelling abilities than other models.
- In the absence of noise, the singleton system performs better than the non-singleton system.
- TSK performs well as the noise reduced especially with singleton systems.

The novel combination of simulated annealing and interval type-2 fuzzy logic systems has been applied in the fourth chapter where it is compared with type-1 fuzzy logic systems. Four problems have been used to examine the modelling abilities for both combinations. These are noise-free and noisy Mackey-Glass time series forecasting (Mackey and Glass, 1977) and two real world problems which are the estimation of the low voltage electrical line length in rural towns and the estimation of the medium

voltage electrical line maintenance cost (Cordón et al., 1999). The analysis of their results showed that:

- Interval type-2 fuzzy logic systems added more abilities to modelling information and handling uncertainties associated with these problems than type-1 fuzzy logic systems.
- type-1 fuzzy logic system was preferred in terms of computations time.

The fifth and sixth chapters proposed the novel work for learning general type-2 fuzzy logic systems using simulated annealing in two different methodologies. The fifth chapter was concerned with learning the third dimension (SMF) using simulated annealing and with the aid of interval type-2 fuzzy logic systems to model the four problems mentioned above. The use of interval type-2 fuzzy logic systems is proposed to get a quick optimised interval type-2 fuzzy sets that are converted to general type-2 fuzzy sets before applying simulated annealing again to learn the best SMF's parameters. In order to achieve the objective of learning SMF, a novel parametrisation method was proposed. Although this method is not a new representation of general type-2 fuzzy sets, it helps much in getting a practical and simple way for SMF's parametrisation. Such parametrisation is needed for learning general type-2 fuzzy logic systems. In addition, the problem of type-reduction bottleneck in general type-2 fuzzy sets and its effects on learning performance has been discussed and a rationale for the chosen methods has been drawn. The exhaustive method for type-reduction is very difficult to use in practice so that the sampling defuzzifier has been chosen instead. After getting the best FOU from the optimised interval type-2 fuzzy sets, an approximation conversion process has been conducted to convert them to general type-2 fuzzy sets. Then, simulated annealing was used to learn the best SMF's parameters only to get better modelling. The main observations in this part are :

- Although, the conversion process has caused some losses in modelling accuracies, the learning of SMF added some improvements to the best converted FOU's results. However, these improvements are very small compared to the scale of the problem and to the improvements brought by FOU. This issue is a result of the stochastic evaluations of the exact type-reduced sets using sampling as proved in the sixth chapter when another deterministic type-reducer has been applied.

- Learning the third dimension can add extra abilities to modelling information over interval type-2 fuzzy logic systems.

The sixth chapter reported the novel work for learning all parameters of general type-2 fuzzy sets to design general type-2 fuzzy logic systems. The designed systems are initialised from scratch rather than using optimised interval type-2 fuzzy sets to start with. The four problems mentioned above were used to examine the proposed system's performance in modelling. Two methods for type-reduction have been used to avoid the distracted evaluations of objective functions when using the sampling defuzzifier. Both type-reducers, the non-deterministic sampling and the deterministic vertical-slices centroid type-reducer (VSCTR) have been used and compared with interval type-2 fuzzy logic systems. The experimentation for the three models showed interesting results as follows:

- Firstly, the results showed that when using the deterministic defuzzification method (VSCTR), the learning of general type-2 fuzzy logic systems can provide extra abilities to handle more information and uncertainties than interval type-2 fuzzy logic systems that use uniform SMFs. Improvements between 0.88% and 11.7% were added to the average testing results when learning SMFs using general type-2 fuzzy logic systems with the deterministic type-reducer.

- The stochastic defuzzification method of sampling embedded sets affects the learning performance in both FOU and SMF learning stages especially when learning SMF.
- The longer computations time needed for general type-2 fuzzy logic systems constitutes a trade-off between the speed brought by interval type-2 fuzzy logic systems and the extra accuracies brought by general type-2 fuzzy logic systems.

This is the first work on learning general type-2 fuzzy logic systems using vertical-slices representation. The clear conclusion that learning the third dimension can add more abilities to modelling is an important advance in type-2 fuzzy logic system research and should open the doors for more promising research on learning general type-2 fuzzy logic systems. Hence, the proposed parametrisation method allow other learning methods used to design interval type-2 fuzzy logic systems to be potential candidates for general type-2 fuzzy logic systems design.

7.2 Limitations and Future Work

The main obstacle encountered through this research is the lack of a practical type-reducer to calculate the exact type-reduced sets. At current time, the only method that can calculate the exact type-reduced sets is the exhaustive method. However, the astronomical computations associated with this method prevent practitioners from using it in practice. Although sampling method helps to get a practical solution to get approximated centroids, its stochastic behaviour caused some deteriorations on the learning performance. On the other hand, while VSCTR is more practical and does not affect learning process, it is not based on the concept of embedded sets that used to build up type-2 fuzzy set.

Considering the impacts of the work reported in this thesis, many research opportunities with great potential can be tackled. Some of these opportunities are :

- Using other variants of simulated annealing to best suit type-2 fuzzy logic systems. Examples of such variants include Adaptive Simulated Annealing (ASA) (Ingber, 1993), Very Fast Simulated Re-Annealing (Ingber, 1989) and Basin Hopping (Wales and Scheraga, 1999).

- Using different components of simulated annealing to improve simulated annealing search for type-2 fuzzy logic systems parameters. Examples of different components include using adaptive step sizes, dynamic cooling schedules and other acceptance rules.

- Extending the parametrisation method proposed in this work to other membership function shapes such as trapezoidal, s-shaped and z-shaped membership functions. In this direction, extending the method to serve different cases such as handling more than one apex in SMF.

- Regarding the type-2 fuzzy sets and systems side, a high priority should be given to propose a type-reducer that is practical, deterministic and based on the concept of embedded sets as pointed out above. In addition, there is a need for fast meet and join operations to handle cases where more than one apex exists in SMF such as in trapezoidal function. Fast meet and join operations will help speed up type-2 fuzzy logic systems inference but the type-reduction stage is the most computationally expensive part and should be given priority.

- Using other learning methods other than simulated annealing to design general type-2 fuzzy logic systems automatically. There is a large number of such methods that have been used with type-1 and interval type-2 fuzzy logic systems with different performances. Therefore, the start with the less computationally expensive methods might be a preferable direction.

- Investigating for robust approximation methods to convert between interval type-2 fuzzy sets and general type-2 fuzzy sets. Such method will help getting the best

optimised FOU without using the most computationally expensive mathematics of general type-2 fuzzy logic systems.

- Using the proposed combination in this thesis to model other real-world problems especially problems with no time constraints.

This thesis has shown a successful development of an enhanced modelling using a novel hybrid approach. The work has opened the doors for more research opportunities with great potential and for promising practical applications.

Bibliography

Aarts, E. H. L. and Eikelder, H. M. M. T. (2002), Simulated annealing, *in* P. Pardalos and M. Resende, eds, 'Handbook of applied optimization', Oxford University Press, pp. 209–220.

Aarts, E. and Lenstra, J. (2003), *Local search in combinatorial optimization*, Princeton Univ Press.

Al-Jaafreh, M. and Al-Jumaily, A. (2007), Training type-2 fuzzy system by particle swarm optimization, *in* 'Evolutionary Computation, 2007. CEC 2007. IEEE Congress on', IEEE, pp. 3442–3446.

Alcala, R., Casillas, J., Cordón, O., Herrera, F. and Zwir, I. (1999), 'Techniques for learning and tuning fuzzy rule-based systems for linguistic modeling and their applications', *Knowledge Engineering Systems: Techniques and Applications* pp. 889–941.

Almaraashi, M. (2010), Optimisation of fuzzy tsk consequents by simulated annealing, *in* 'The 4th Saudi International Conference', Manchester.

Almaraashi, M. and John, R. (2010), Tuning fuzzy systems by simulated annealing to predict time series with added noise, *in* 'Proceedings of UKCI', Essex, UK.

Almaraashi, M. and John, R. (2011*a*), Tuning of type-2 fuzzy systems by simulated annealing to predict time series, *in* 'Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2011 , WCE 2011', Vol. 2, Newswood Limited, London, U.K, pp. 976–980.

Almaraashi, M. and John, R. (2011*b*), Tuning type-2 fuzzy systems by simulated annealing to estimate maintenance cost, *in* 'proceedings the UKCI 2011', Manchester.

Almaraashi, M., John, R. and Ahmadi, S. (2012*a*), Electrical engineering and intelligent systems book, *in* S. I. Ao and L. Gelman, eds, 'Learning of Type-2 Fuzzy Logic Systems by Simulated Annealing with Adaptive Step Size', Vol. 130 of *Lecture Notes in Electrical Engineering*, Springer, chapter 5.

Almaraashi, M., John, R. and Ahmadi, S. (2012*b*), 'Learning of type-2 fuzzy logic systems by simulated annealing algorithm', *the International Journal of Approximate Reasoning* . under review.

Almaraashi, M., John, R. and Coupland, S. (2012), Designing generalised type-2 fuzzy logic systems using interval type-2 fuzzy logic systems and simulated annealing, *in* 'Fuzzy Systems (FUZZ), 2012 IEEE International Conference on', IEEE. Accepted.

Almaraashi, M., John, R., Coupland, S. and Hopgood, A. (2010), Time series forecasting using a tsk fuzzy system tuned with simulated annealing, *in* 'Fuzzy Systems (FUZZ), 2010 IEEE International Conference on', pp. 1 –6.

Branke, J., Meisel, S. and Schmidt, C. (2008), 'Simulated annealing in the presence of noise', *Journal of Heuristics* **14**(6), 627–654.

Casillas, G. (2011), 'Fuzzy modeling library (fmlib)', Available at `http://decsai.ugr.es/~casillas/fmlib/index.html`. [Accessed: 28 March 2011].

Christian Wagner, H. H. (2009), Novel methods for the design of general type-2 fuzzy sets based on device characteristics and linguistic labels surveys, *in* '2009 IFSA World Congress, EUSFLAT World Conference', Lisbon, Portugal, pp. 537–543.

Cordon, O., Gomide, F., Herrera, F., Hoffmann, F. and Magdalena, L. (2004), 'Ten years of genetic fuzzy systems: current framework and new trends', *Fuzzy Sets and Systems* **141**(1), 5–32.

Cordón, O., Herrera, F., Hoffmann, F. and Magdalena, L. (2001), *Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases*, World Scientific Pub Co Inc.

Cordón, O., Herrera, F. and Sánchez, L. (1999), 'Solving electrical distribution problems using hybrid evolutionary data analysis techniques', *Applied Intelligence* **10**(1), 5–24.

Cordón, O., Herrera, F. and Villar, P. (2000), 'Analysis and guidelines to obtain a good uniform fuzzy partition granularity for fuzzy rule-based systems using simulated annealing', *International Journal of Approximate Reasoning* **25**(3), 187–215.

Cordón, O., Herrera, F. and Villar, P. (2001), 'Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base', *Fuzzy Systems, IEEE Transactions on* **9**(4), 667–674.

Cordón, O., Herrera, F. and Zwir, I. (2002), 'Linguistic modeling by hierarchical systems of linguistic rules', *Fuzzy Systems, IEEE Transactions on* **10**(1), 2–20.

Coupland, S. and John, R. (2007), 'Geometric type-1 and type-2 fuzzy logic systems', *Fuzzy Systems, IEEE Transactions on* **15**(1), 3 –15.

Coupland, S., M.Gongora, John, R. I. and K.Wills (2006), A comparative study of fuzzy logic controllers for autonomous robots, *in* 'Proc. IPMU 2006', Paris, France, pp. 1332 – 1339.

Dadone, P. (2001), Design Optimization of Fuzzy Logic Systems, PhD thesis, Virginia Polytechnic Institute and State University.

De Oliveira, J. (1999), 'Semantic constraints for membership function optimization', *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* **29**(1), 128 –138.

Drack, L. and Zadeh, H. (2006), 'Soft computing in engineering design optimisation', *Journal of Intelligent and Fuzzy Systems* **17**(4), 353–365.

Dubois, D. and Prade, H. (1980), *Fuzzy sets and systems: theory and applications*, Vol. 144, Academic Pr.

Gafa, C. and Coupland, S. (2011), A new recursive type-reduction procedure for general type-2 fuzzy sets, *in* 'Advances in Type-2 Fuzzy Logic Systems (T2FUZZ), 2011 IEEE Symposium on', pp. 44 –49.

Garibaldi, J. and Ifeachor, E. (1999), 'Application of simulated annealing fuzzy model tuning to umbilical cord acid-base interpretation', *Fuzzy Systems, IEEE Transactions on* **7**(1), 72 –84.

Goonatilake, S. and Khebbal, S. (1995), *Intelligent Hybrid Systems*, John Wiley & Sons.

Greenfield, S., Chiclana, F., Coupland, S. and John, R. (2009), 'The collapsing method of defuzzification for discretised interval type-2 fuzzy sets', *Information Sciences* **179**(13), 2055–2069. ISSN: 0020-0255.

Greenfield, S., Chiclana, F. and John, R. (2009), The collapsing method: Does the direction of collapse affect accuracy?, *in* 'IFSA-EUSFLAT 2009 Conference'.

Greenfield, S., Chiclana, F., John, R. and Coupland, S. (2012), 'The sampling method of defuzzification for type-2 fuzzy sets: Experimental evaluation', *Information Sciences* **189**(0), 77 – 92.

Greenfield, S. and John, R. (2007), Optimised generalised type-2 join and meet operations, *in* 'Fuzzy Systems Conference, 2007. FUZZ-IEEE 2007. IEEE International', pp. 1 –6.

Greenfield, S., John, R. and Coupland, S. (2005), A novel sampling method for type-2 defuzzification, *in* 'Proceedings of UKCI 2005', London, pp. 120–127.

Guely, F., La, R. and Siarry, P. (1999), 'Fuzzy rule base learning through simulated annealing', *Fuzzy Sets and Systems* **105**(3), 353 – 363.

Hagras, H. (2007), 'Type-2 flcs: A new generation of fuzzy controllers', *Computational Intelligence Magazine, IEEE* **2**(1), 30–43.

Hamrawi, H., Coupland, S. and John, R. (2010), A novel alpha-cut representation for type-2 fuzzy sets, *in* 'FUZZ IEEE 2010 (WCCI 2010)', IEEE, IEEE, Barcelona, Spain, pp. 1 – 8.

Herrera, F. (2005), 'Genetic fuzzy systems: status, critical considerations and future directions', *International Journal of Computational Intelligence Research* **1**(1-2), 59–67.

Hoffmann, F. (2001), 'Evolutionary algorithms for fuzzy control system design', *Proceedings of the IEEE* **89**(9), 1318–1333.

Horikawa, S., Furuhashi, T. and Uchikawa, Y. (1992), 'On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm', *Neural Networks, IEEE Transactions on* **3**(5), 801–806.

Huyghe, E. and Hamam, Y. (1995), Simulated annealing for fuzzy controller optimization: principles and applications, *in* 'Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on', Vol. 5, pp. 4509 –4514 vol.5.

Hyndman, R. J. and Koehler, A. B. (2006), 'Another look at measures of forecast accuracy', *International Journal of Forecasting* **22**(4), 679 – 688.

Ingber, L. (1989), 'Very fast simulated re-annealing', *Mathematical and Computer Modelling* **12**(8), 967–973.

Ingber, L. (1993), 'Simulated annealing: Practice versus theory', *Mathematical and computer modelling* **18**(11), 29–57.

Jang, J.-S. (1993), 'Anfis: adaptive-network-based fuzzy inference system', *Systems, Man and Cybernetics, IEEE Transactions on* **23**(3), 665 –685.

Jang, J. and Sun, C. (1995), 'Neuro-fuzzy modeling and control', *Proceedings of the IEEE* **83**(3), 378–406.

Jang, J., Sun, C. and Mizutani, E. (1997), *Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence. Solutions manual*, Prentice Hall.

Jeng, W., Yeh, C. and Lee, S. (2009), General type-2 fuzzy neural network with hybrid learning for function approximation, *in* 'Fuzzy Systems, 2009. FUZZ-IEEE 2009. IEEE International Conference on', IEEE, pp. 1534–1539.

John, R. (2000), Perception modelling using type-2 fuzzy sets / R. I. John., PhD thesis, De Montfort University.

John, R. and Coupland, S. (2006), 'Extensions to type-1 fuzzy: type-2 fuzzy logic and uncertainty', *Computational Intelligence: Principles and Practice* pp. 89–102.

John, R. and Coupland, S. (2007), 'Type-2 fuzzy logic: A historical view', *Computational Intelligence Magazine, IEEE* **2**, 57 –62.

John, R. and Czarnecki, C. (1998), A type 2 adaptive fuzzy inferencing system, *in* 'Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on', Vol. 2, IEEE, pp. 2068–2073.

Juang, C.-F. and Lin, C.-T. (1998), 'An online self-constructing neural fuzzy inference network and its applications', *Fuzzy Systems, IEEE Transactions on* **6**(1), 12 –32.

Karnik, N. and Mendel, J. (2001*a*), 'Centroid of a type-2 fuzzy set', *Information Sciences* **132**(1), 195–220.

Karnik, N. and Mendel, J. (2001*b*), 'Operations on type-2 fuzzy sets', *Fuzzy Sets and Systems* **122**(2), 327–348.

Karnik, N., Mendel, J. and Liang, Q. (1999), 'Type-2 fuzzy logic systems', *Fuzzy Systems, IEEE Transactions on* **7**(6), 643–658.

Kim, D. and Kim, C. (2002), 'Forecasting time series with genetic fuzzy predictor ensemble', *Fuzzy Systems, IEEE Transactions on* **5**(4), 523–535.

Kirkpatrick, S., Gelatt, C. and Vecchi, M. (1983), 'Optimization by simulated annealing, 1983', *Science* **220**, 671–680.

Klir, G. and Wierman, M. (1998), *Uncertainty-Based Information: Elements of Generalized Information Theory*, Physica Verlag.

Kukolj, D. (2002), 'Design of adaptive takagi-sugeno-kang fuzzy models', *Applied Soft Computing* **2**(2), 89 – 103.

Lin, C.-J. and Lin, C.-T. (1997), 'An art-based fuzzy adaptive learning control network', *Fuzzy Systems, IEEE Transactions on* **5**(4), 477 –496.

Linda, O. and Manic, M. (2010), Importance sampling based defuzzification for general type-2 fuzzy sets, *in* 'Fuzzy Systems (FUZZ), 2010 IEEE International Conference on', pp. 1 –7.

Liska, J. and Melsheimer, S. (1994), Complete design of fuzzy logic systems using genetic algorithms, *in* 'Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on', pp. 1377 –1382 vol.2.

Liu, F. (2008), 'An efficient centroid type-reduction strategy for general type-2 fuzzy logic system', *Information Sciences* **178**(9), 2224–2236.

Liu, G. and Yang, W. (2000), Learning and tuning of fuzzy membership functions by simulated annealing algorithm, *in* 'Circuits and Systems, 2000. IEEE APCCAS 2000. The 2000 IEEE Asia-Pacific Conference on', pp. 367–370.

Lo, J.-C. and Yang, C.-H. (1999), 'A heuristic error-feedback learning algorithm for fuzzy modeling', *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* **29**(6), 686 –691.

Locatelli, M. (2000), 'Simulated annealing algorithms for continuous global optimization: convergence conditions', *Journal of Optimization Theory and applications* **104**(1), 121–133.

Locatelli, M. (2002), 'Simulated annealing algorithms for continuous global optimization', *Handbook of global optimization* **2**, 179–229.

Lucas, L., Centeno, T. and Delgado, M. (2007), General type-2 fuzzy inference systems: Analysis, design and computational aspects, *in* 'Fuzzy Systems Conference, 2007. FUZZ-IEEE 2007. IEEE International', pp. 1 –6.

Mackey, M. and Glass, L. (1977), 'Oscillation and chaos in physiological control systems', *Science* **197**(4300), 287–289.

Mamdani, E. (1974), 'Application of fuzzy algorithms for control of simple dynamic plant', *Electrical Engineers, Proceedings of the Institution of* **121**(12), 1585 –1588.

Martinez, R., Castillo, O. and Aguilar, L. T. (2009), 'Optimization of interval type-2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms', *Information Sciences* **179**(13), 2158–2174.

Mendel, J. (2001), *Uncertain rule-based fuzzy logic systems: introduction and new directions*, Prentice Hall.

Mendel, J. (2003), Fuzzy sets for words: a new beginning, *in* 'Fuzzy Systems, 2003. FUZZ'03. The 12th IEEE International Conference on', Vol. 1.

Mendel, J. (2007), 'Advances in type-2 fuzzy sets and systems', *Information Sciences* **177**(1), 84–110.

Mendel, J. and John, R. (2002), 'Type-2 fuzzy sets made simple', *Fuzzy Systems, IEEE Transactions on* **10**(2), 117 –127.

Mendel, J., John, R. and Liu, F. (2006), 'Interval type-2 fuzzy logic systems made simple', *Fuzzy Systems, IEEE Transactions on* **14**(6), 808 –821.

Mendel, J. and Liu, F. (2008), On new quasi-type-2 fuzzy logic systems, *in* 'Fuzzy Systems, 2008. FUZZ-IEEE 2008.(IEEE World Congress on Computational Intelligence). IEEE International Conference on', IEEE, pp. 354–360.

Mendel, J., Liu, F. and Zhai, D. (2009), 'Alpha plane representation for type-2 fuzzy sets: Theory and applications', *Fuzzy Systems, IEEE Transactions on* **17**(5), 1189 –1207.

Méndez, G. M. and de los Angeles Hernandez, M. (2009), 'Hybrid learning for interval type-2 fuzzy logic systems based on orthogonal least-squares and back-propagation methods', *Information Sciences* **179**(13), 2146 – 2157.

Miki, M., Hiroyasu, T. and Ono, K. (2002), Simulated annealing with advanced adaptive neighborhood, *in* 'Second international workshop on Intelligent systems design and application', Dynamic Publishers, Inc., pp. 113–118.

Miller, S., Gongora, M. and John, R. (2011), Interval type-2 fuzzy modelling and simulated annealing for real-world inventory management, *in* 'Hybrid Artificial Intelligence Systems 2011 (HAIS2011) 23-25 May, 2011 Wroclaw, Poland', pp. 231–238.

Mizumoto, M. and Tanaka, K. (1976), 'Some properties of fuzzy sets of type 2', *Information and Control* **31**(4), 312 – 340.

Mouzouris, G. and Mendel, J. (1994), Non-singleton fuzzy logic systems, *in* 'Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on', pp. 456–461.

Musikasuwan, S., Ozen, T. and Garibaldi, J. M. (2004), An investigation into the effect of number of model parameters on performance in type-1 and type-2 fuzzy logic systems, *in* 'In Proc. 10 th Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU'.

Negnevitsky, M. (2002), *Artificial intelligence: a guide to intelligent systems*, Addison-Wesley.

Nie, M. and Tan, W. W. (2008), Towards an efficient type-reduction method for interval type-2 fuzzy logic systems, *in* 'Fuzzy Systems, 2008. FUZZ-IEEE 2008. (IEEE World Congress on Computational Intelligence). IEEE International Conference on', pp. 1425 –1432.

Nolle, L., Goodyear, A., Hopgood, A., Picton, P. and Braithwaite, N. (2001), On step width adaptation in simulated annealing for continuous parameter optimisation, *in* B. Reusch, ed., 'Computational Intelligence. Theory and Applications', Vol. 2206 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 589–598.

Oliveira, H., Petraglia, A., Ingber, L., Machado, M. and Petraglia, M. (2012), *Stochastic global optimization and its applications with fuzzy adaptive simulated annealing*, Springer, New York.

Onbaşoğlu, E. and Özdamar, L. (2001), 'Parallel simulated annealing algorithms in global optimization', *Journal of Global Optimization* **19**(1), 27–50.

Ram, D., Sreenivas, T. and Subramaniam, K. (1996), 'Parallel simulated annealing algorithms', *Journal of parallel and distributed computing* **37**(2), 207–212.

Reznik, L. (1997), *Fuzzy controllers*, Butterworth-Heinemann.

Ross, T. J. (2004), *Fuzzy Logic with Engineering Applications*, Wiley.

Russo, M. (2000), 'Genetic fuzzy learning', *Evolutionary Computation, IEEE Transactions on* **4**(3), 259 –273.

Salamon, P., Sibani, P. and Frost, R. (2002), *Facts, conjectures, and improvements for simulated annealing*, Society for Industrial Mathematics.

Sepúlveda, R., Castillo, O., Melin, P., Rodríguez-Díaz, A. and Montiel, O. (2007), 'Experimental study of intelligent controllers under uncertainty using type-1 and type-2 fuzzy logic', *Information Sciences* **177**(10), 2023–2048.

Shann, J. and Fu, H. (1995), 'A fuzzy neural network for rule acquiring on fuzzy control systems', *Fuzzy Sets and Systems* **71**(3), 345–357.

Starczewski, J. (2009*a*), 'Extended triangular norms', *Information Sciences* **179**(6), 742–757.

Starczewski, J. T. (2009*b*), 'Efficient triangular type-2 fuzzy logic systems', *International Journal of Approximate Reasoning* **50**(5), 799–811.

Takagi, T. and Sugeno, M. (1985), 'Fuzzy identification of systems and its applications to modeling and control', *Systems, man, and cybernetics, IEEE transactions on* **15**(1), 116–132.

Wagner, C. and Hagras, H. (2007), A genetic algorithm based architecture for evolving type-2 fuzzy logic controllers for real world autonomous mobile robots, *in* 'Fuzzy Systems Conference, 2007. FUZZ-IEEE 2007. IEEE International', pp. 1 –6.

Wagner, C. and Hagras, H. (2009), zslices based general type-2 flc for the control of autonomous mobile robots in real world environments, *in* 'Fuzzy Systems, 2009. FUZZ-IEEE 2009. IEEE International Conference on', pp. 718 –725.

Wagner, C. and Hagras, H. (2010*a*), 'Toward general type-2 fuzzy logic systems based on zslices', *Fuzzy Systems, IEEE Transactions on* **18**(4), 637 –660.

Wagner, C. and Hagras, H. (2010*b*), Uncertainty and type-2 fuzzy sets and systems, *in* 'Computational Intelligence (UKCI), 2010 UK Workshop on', pp. 1 –5.

Wales, D. and Scheraga, H. (1999), 'Global optimization of clusters, crystals, and biomolecules', *Science* **285**(5432), 1368.

Wang, L. and Mendel, J. (1992), 'Generating fuzzy rules by learning from examples', *IEEE Transactions on systems, man and cybernetics* **22**(6), 1414–1427.

White, S. (1984), Concepts of scale in simulated annealing, *in* 'American Institute of Physics Conference Series', Vol. 122, pp. 261–270.

Wu, D. and Mendel, J. (2009), 'Enhanced karnik–mendel algorithms', *Fuzzy Systems, IEEE Transactions on* **17**(4), 923 –934.

Wu, D. and Tan, W. (2005*a*), Type-2 fls modeling capability analysis, *in* 'Fuzzy Systems, 2005. FUZZ'05. The 14th IEEE International Conference on', IEEE, pp. 242–247.

Wu, D. and Tan, W. W. (2005*b*), 'Computationally efficient type-reduction strategies for a type-2 fuzzy logic controller', pp. 353 –358.

Wu, D. and Wan Tan, W. (2006), 'Genetic learning and performance evaluation of interval type-2 fuzzy logic controllers', *Engineering Applications of Artificial Intelligence* **19**(8), 829–841.

Wu, H. and Mendel, J. (2002), 'Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems', *Fuzzy Systems, IEEE Transactions on* **10**(5), 622 – 639.

Yanar, T. and Akyrek, Z. (2011), 'Fuzzy model tuning using simulated annealing', *Expert Systems with Applications* **38**(7), 8159–8169.

Zadeh, L. (1965), 'Information and control', *Fuzzy sets* **8**(3), 338–353.

Zadeh, L. (1975), 'The concept of a linguistic variable and its application to approximate reasoning. i', *Inform. Sciences* **8**, 199–249.

Zimmermann, H. (2001), *Fuzzy set theory–and its applications*, Kluwer Academic Publishers.

Appendix A

Publications by Majid Almaraashi that are Directly Related to this Thesis

Almaraashi, M., John, R., Coupland, S. and Hopgood, A. (2010), Time series forecasting using a tsk fuzzy system tuned with simulated annealing, in Fuzzy Systems (FUZZ), 2010 IEEE International Conference on, pp. 1 6.

Almaraashi, M. and John, R. (2010), Tuning fuzzy systems by simulated annealing to predict time series with added noise, in Proceedings of UKCI, Essex, UK.

Almaraashi, M. (2010), Optimisation of fuzzy tsk consequents by simulated annealing, in The 4th Saudi International Conference, Manchester.

Almaraashi, M. and John, R. (2011), Tuning of type-2 fuzzy systems by simulated an- nealing to predict time series, in Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2011 , WCE 2011, Vol. 2, Newswood Limited, London, U.K, pp. 976980. This paper won The Best Student

Paper Award of The 2011 International Conference of Computational Intelligence and Intelligent Systems. London, July 2011.

Almaraashi, M. and John, R. (2011), Tuning type-2 fuzzy systems by simulated annealing to estimate maintenance cost, in proceedings the UKCI 2011, Manchester.

Almaraashi, M., John, R. and Ahmadi, S. (2012), Electrical engineering and intelligent systems book, in S. I. Ao and L. Gelman, eds, Learning of Type-2 Fuzzy Logic Systems by Simulated Annealing with Adaptive Step Size, Vol. 130 of Lecture Notes in Electrical Engineering, Springer, chapter 5.

Almaraashi, M., John, R. and Ahmadi, S. (2012), Learning of type-2 fuzzy logic systems by simulated annealing algorithm, the International Journal of Approximate Reasoning . Under review.

Almaraashi, M., John, R. and Coupland, S. (2012), Designing generalised type-2 fuzzy logic systems using interval type-2 fuzzy logic systems and simulated annealing, in Fuzzy Systems (FUZZ), 2012 IEEE International Conference on, IEEE. Accepted.