FORECASTING AND MODELING STATISTICAL PHENOMENA USING NEUROFUZZY LOGIC: A CASE STUDY OF RAINFALL FORECASTING FOR ZARIA

BY

MUHAMMED BASHIR MU'AZU
(PhD/ENG/31659/01-02)

A Dissertation Submitted To The Department of Electrical Engineering, Ahmadu Bello University, Zaria In Partial Fulfillment of The Requirements For The Award of A Doctor of Philosophy (Ph.D) Degree In Electrical Engineering.

DECLARATION

I, MU'AZU, Muhammed Bashir, **hereby declare that this dissertation titled** "Forecasting And Modeling Statistical Phenomena Using NeuroFuzzy Logic: A Case Study Of Rainfall Forecasting For Zaria" **presented to the Department of Electrical Engineering, Ahmadu Bello University, Zaria is the result of my own research and that it has not been presented in any form, anywhere for the award of a degree.**

All literature herein referenced and cited have been duly acknowledged. All shortcomings in this work are entirely my responsibility.

…….. ……

MU'AZU, Muhammed Bashir Date

CERTIFICATION

This dissertation titled "Forecasting And Modeling Statistical Phenomena Using NeuroFuzzy Logic: A Case Study Of Rainfall Forecasting For Zaria" **satisfies one of the requirements for the award of a Doctor of Philosophy (Ph.D) degree in Electrical Engineering and has been approved by the Department of Electrical Engineering, Ahmadu Bello University, Zaria.**

…….. ……

Prof. B. G. Bajoga Date
(Supervisory Committee)

…….. ……

Prof. M. A. Gulma, MFR Date
(Supervisory Committee)

…….. ……

Dr. B. A. Adegboye Date
(Head of Department)

…….. ……

Prof. J. Umoh Date
(Dean, Postgraduate School)

DEDICATION

This work is dedicated to the **GLORY** of **ALMIGHTY ALLAH** (SWT) who **"*Taught man that which he knew not*" (Quran 96:5)**

And to my **Parents** and the entire **Mu'azu** family for always been there through thick and thin.

ACKNOWLEDGEMENTS

In the name of ALLAH, THE MOST BENEFICIENT, THE MOST MERCIFUL.

This work was a journey into 'unknown' territories for me and as such I humbly acknowledge the contributions of authors, so referred, in the areas of Soft Computing and Time Series forecasting.

I am most grateful for the guidance of my supervisor, Prof. B. G. Bajoga. My motivation to carry out this investigation was largely due to him and his suggestions have been quite useful as they helped me to keep sight of the real issues involved in research works of this nature. I am also grateful to Prof. M. A. Gulma and Dr. P. F. U. Taylor for their respective contributions.

I am also grateful for the contributions of Jamilu Abubakar (Microsoft, UK), Amina Ado (Total, Nig), Ehi Anenih (Glo, Nig) and Wale Adedokun (ICT, A.B.U.) in helping me secure books and relevant literature in Soft Computing. I am also grateful to my colleagues, especially Yusuf Jibril, and Boyi Jimoh for all their encouragements.

I am also grateful to Mr. F. Eshimiakhe and Mr. A. Kazzah of NiMet Zaria and Mr. Badakin of the Meteorological Unit, Soil Science Department, A.B.U. Zaria for being quite helpful in providing me the data and other useful suggestions. The same goes to Mr. Sawa of Geography Department, A.B.U. Zaria, who gave me insight to the issue of rainfall.

And, to my family, Hassana, Hameed and Nabeelah, for persevering as this work took a lot of my time and attention. THANKS.

Finally, and most importantly, I remain eternally grateful to **ALLAH** (SWT) for His grace and mercies that enabled me to start and reach this stage of the work. Nothing better can be said at this juncture than ***"ALHAMDULILLAHI"***.

I sincerely hope that this work will be a useful contribution to knowledge.

MU'AZU, Muhammed Bashir
(ABUDEE, Zaria)
June 2006

TABLE OF CONTENTS

TITLE PAGE	i
DECLARATION	ii
CERTIFICATION	iii
DEDICATION	iv
ACKNOWLEDGEMENTS	v
TABLE OF CONTENTS	vii
LIST OF FIGURES	xi
LIST OF TABLES	xv
LIST OF SYMBOLS	xvi
ABSTRACT	xviii
CHAPTER ONE: INTRODUCTION AND BACKGROUND	
1.0 INTRODUCTION	1
1.1 OVERVIEW OF SOFT COMPUTING TECHNIQUES	2
1.2 LITERATURE REVIEW	7
1.3 STATEMENT OF PROBLEM	13
1.4 RESEARCH OBJECTIVES AND METHODOLOGY	19
1.5 THESIS OUTLINE	22
CHAPTER TWO: SOFT COMPUTING TECHNIQUES	
2.0 INTRODUCTION	23
2.1 ARTIFICIAL NEURAL NETWORK	23
2.1.1 NEURAL NET MATHEMATICAL MODEL	26

2.1.2 NEURAL NET CLASSIFICATIONS 28

2.1.3 ACTIVATION FUNCTION 31

2.1.4 NEURAL NET LEARNING: THE ERROR BACK PROPAGATION 34

2:2 FUZZY LOGIC SYSTEM 37

2.2.1 MATHEMATICAL UNCERTAINTY 38

2.2.2 FUZZY SET THEORY 38

2.2.2.1 FUZZY SET OPERATORS AND HEDGES 39

2.2.3 FUZZY LOGIC SYSTEM 40

2.2.4 FUZZY LOGIC SYSTEM LAYER STRUCTURE 43

2.2.4.1 FUZZIFICATION 44

2.2.4.2 FUZZY INFERENCE 45

2.2.4.3 DEFUZZIFICATION 48

2.3 NEURO-FUZZY SYSTEMS 50

2.3.1 NEURAL NETS AND FUZZY LOGIC SYSTEMS INTEGRATION 50

2.3.2 NEURO-FUZZY SYSTEM STRUCTURES 53

2.3.2.1 MAMDANI NEURO-FUZZY SYSTEM 53

2.3.2.2 TAKAGI-SUGENO (TS) NEURO-FUZZY SYSTEM 55

2.3.3 NEURO-FUZZY MODEL: THE MODIFIED *nfMod* MODEL 57

2.3.3.1 TRIANGULAR-NORMS (T-norm) AND CONORMS (T-conorm or S-norm) 59

2.3.4 LEARNING ALGORITHMS 61

2.3.4.1 HEBBIAN LEARNING RULE 61

2.3.4.2 WINNER TAKES ALL (WTA) LEARNING RULE 62

2.3.4.3 DELTA RULE AND ERROR BACK PROPAGATION (EBP) 62

2.3.5 NEURO-FUZZY LEARNING ALGORITHM 64

CHAPTER THREE: DEVELOPMENT OF FORECAST MODEL

3.1 DEVELOPMENT OF NEUROFUZZY-BASED FORECAST MODEL 69

3.2 RAIN FORECAST MODEL 87

CHAPTER FOUR: TIME SERIES PREDICTION OF WEATHER VARIABLES

4.1 INTRODUCTION 95

4.2 SIMPLE MOVING AVERAGES 97

4.3 SIMPLE EXPONENTIAL SMOOTHING METHOD 99

4.4 TESTING WITH THE RAIN FORECAST MODEL (RFM) **102**

CHAPTER FIVE: FUZZY TIME SERIES FORECASTING

5.1 INTRODUCTION **106**

5.2 FUZZY TIME SERIES **107**

5.3 FORECASTING METHODOLOGY **108**

CHAPTER SIX: RESULTS AND ANALYSIS

6.1 INTRODUCTION **117**

6.2 RAIN FORECAST MODEL (RFM) **118**

6.3 FORECASTING RAINFALL USING FORECASTED WEATHER VARAIABLES **121**

6.4 FORECASTING RAINFALL USING FUZZY TIME SERIES **123**

6.5 RAINFALL FORECAST RESULTS **125**

6.6 SIGNIFICANCE OF RESULTS OBTAINED FROM INVESTIGATION **127**

CHAPTER SEVEN: CONCLUSIONS AND SUGGESTIONS FOR FURTHER WORK

7.1 INTRODUCTION **130**

7.2 MAJOR HIGHLIGHTS **131**

7.3 CONSTRAINTS **132**

7.4 CONCLUSIONS **133**

7.5 SUGGESTIONS FOR FURTHER WORK **134**

REFERENCES **137**

APPENDIX A1: COMPLETE TRAINING DATA (1993 – 2002) 150

APPENDIX A2: COMPLETE RULE BASE (625 RULES) 152

APPENDIX A3: **RULE BASE OF THE RAIN FORECAST MODEL (57 RULES)** **167**

APPENDIX A4: COMPLETE PROGRAM LISTING *169*

APPENDIX A5: FORECAST RESULTS OF MOVING AVERAGES AND EXPONENTIAL SMOOTHING METHODS 177

APPENDIX A6: THE COMPLETE FORECASTED RAINFALL DATA OBTAINED FROM THE RAIN FORECAST MODEL (RFM) 180

APPENDIX A7: **FUZZY TIME SERIES DATA** **182**

LIST OF FIGURES

Figure 1.1: Fuzzy Logic Block Diagram 4

Figure 1.2: A Simple Neural Network 6

Figure 1.3: Plot of Relative Humidity v. Amount of Rainfall 16

Figure 1.4: Plot of Wind Direction v. Amount of Rainfall 17

Figure 1.5: Plot of Wind Speed v. Amount of Rainfall 17

Figure 1.6: Rainfall and Weather Variables Trend For 1993 – 2002. 18

Figure 2.1: McCulloch-Pitts Unit 24

Figure 2.2: Simplified Human Neuron Model. 25

Figure 2.3: Basic Mathematical Model of A Neuron. 27

Figure 2.4: Simple Feedforward Neural Net With Three Layers 29

Figure 2.5: Logistic Sigmoid 32

Figure 2.6: Neural Net Activated With The Sigmoid Function 34

Figure 2.7: The Basic Flowchart For The Error Back Propagation 37

Figure 2.8: a) Intersection of A and B 40

Figure 2.8: b) Union of A and B 40

Figure 2.9: Typical Fuzzy Logic Controller Block Diagram 42

Figure 2.10: Schematic of Fuzzy Logic System Layer Structure. 43

Figure 2.11: Standard Membership Functions 44

Figure 2.12 Centre of Maximum Method 49

Figure 2.13: Centre of Area Method 49

Figure 2.14: Concurrent NeuroFuzzy Model 53

Figure 2.15: Concurrent Neuro-Fuzzy Model 53

Figure 2.16: Mamdani NeuroFuzzy System 55

Figure 2.17: Takagi-Sugeno (TS) NeuroFuzzy System 56

Figure 2:18: Modified ***nfMod*** NeuroFuzzy System Structure 58

Figure 2.19: Triangular Membership Function 67

Figure 3.1: Generated Data File Using Notepad 70

Figure 3.2: Sample Rule Base of Empty Fuzzy System (DoS of 0) 71

Figure 3.3: Fuzzy Design Wizard Dialog Box For Number of Variables 72

Figure 3.4 Fuzzy Design Wizard Dialog Box For Selecting Defuzzification Method 72

Figure 3.5: Rainfall Membership Functions Opened For Learning 74

Figure 3.6: NeuroFuzzy Configuration Dialog Box 75

Figure 3.7: Test Plot for 200 Iterations 78

Figure 3.8: Test Plot for 250 Iterations 79

Figure 3.9: Test Plot for 300 Iterations 79

Figure 3.10: Test Plot for 350 Iterations 79

Figure 3.11: Test Plot for 370 Iterations 80

Figure 3.12: Test Plot for 400 Iterations 80

Figure 3.13: Test Plot for 500 Iterations 80

Figure 3.14: Test Plot for 1000 Iterations 81

Figure 3.15: Interactive Debug Mode With The Watch Window Activated 82

Figure 3.16: NeuroFuzzy Learning Control Dialog Before Start of Training 83

Figure 3.17: NeuroFuzzy Learning Control Dialog After Training 83

Figure 3.18: 3D Plot Showing Transfer Surface Before Training 86

Figure 3.19: Alpha-Cut Dialog Box 86

Figure 3.20: Structure of the Rain Forecast Model Fuzzy Logic System 88

Figure 3.21: Membership Function (MBF) of Relative Humidity (Rel_Hum) 88

Figure 3.22: Membership Function (MBF) of Wind Direction (Wind_Dir) 89

Figure 3.23: Membership Function (MBF) of Wind Speed (Wind_Spd) 89

Figure 3.24: Membership Function (MBF) of Rainfall (Rainfall) 89

Figure 3.25: Interactive Watch Window With Example Data of August 2003. 90

Figure 3.26: Interactive Watch Window With Example Data of July 2004. 91

Figure 3.27: Interactive Watch Window With Example Data of September 2005. 91

Figure 3.28: Rules Fired For Example Data of August 2003. 92

Figure 3.29: Rules Fired For Example Data of July 2004. 92

Figure 3.30: Rules Fired For Example Data of September 2005. 93

Figure 3.31: 3D Plot of Rel_Hum, Wind_Dir and Rainfall (July 2004) 93

Figure 3.32: 3D Plot of Rel_Hum, Wind_Spd and Rainfall (July 2004) 94

Figure 3.33: 3D Plot of Wind_Dir, Wind_Spd and Rainfall (July 2004) 94

Figure 4.1: Time Series Plot of Relative Humidity. 96

Figure 4.2: Time Series Plot of Wind Direction 96

Figure 4.3: Time Series Plot of Wind Speed 96

Figure 4.4: Sample Microsoft Excel® Spreadsheet Showing Exponential Smoothing Procedure 100

Figure 4.5: Time Series Plot of Relative Humidity (1993-2010). 101

Figure 4.6: Time Series Plot of Wind Direction (1993-2010) 101

Figure 4.7: Time Series Plot of Wind Speed (1993-2010) 101

Figure 4.8: Watch Window Output For October 2005 102

Figure 4.9: Watch Window Output For April 2006 103

Figure 4.10: Watch Window Output For August 2008 103

Figure 4.11: Watch Window Output For July 2010 104

Figure 4.12: Plot of Actual (2003-2005) And Forecasted (2003-2010) Rainfall Data 105

Figure 4.13: Plot of Actual (2003-2005), Validation (2003-2005) And Forecasted (2003-2010) Rainfall Data 105

Figure 5.1: Membership Function (MBF) of Set of Fuzzy Intervals 111

Figure 5.2: Plot of Actual Rainfall And Forecasted Rainfall Using Fuzzy Time Series 116

Figure 5.3: Plot of Actual Rainfall, Forecasted Rainfall Using Fuzzy Time And Forecasted Rainfall Using The Rain Forecast Model (RFM) 116

Figure 6.1: Structure of the Rain Forecast Model Fuzzy Logic System 119

Figure 6.2: Plot of Actual Rainfall And Forecasted Rainfall Using Validation Data of 2003 – 2005 120

Figure 6.3: Plot of Actual (2003-2005) And Forecasted (2003-2010) Rainfall Data 121

Figure 6.4: Plot of Actual Rainfall And Forecasted Rainfall Using Fuzzy Time Series 123

Figure 6.5: Plot of Forecasted Rainfall Using Rain Forecast Model (RFM) And Fuzzy Time Series 125

LIST OF TABLES

Table 2.1: Number of Hidden Layers And Situations Where They Can Be Applied 31

Table 2.2: Characteristic Differences Between Neural Nets And Fuzzy Logic Systems 51

Table 3.1 Parameters of The Rain Forecast Model Rule Block 90

Table 4.1: MSE Values For n=3, n=5, n=7 and n=9 98

Table 5.1: Variation and Fuzzification of Variations 112

Table 5.2: Forecasting Errors With Various Values of **w** 112

Table 5.3: Forecasted Rainfall For September 2003 to 2010 115

Table 6.1: Root Mean Square Error (RMSE) Between Actual And Forecasted Rainfall 120

Table 6.2: Root Mean Square Error (RMSE) Between Actual And Forecasted Rainfall Using Forecasted Weather Variables 122

Table 6.3: Root Mean Square Error (RMSE) Between Actual And Forecasted Rainfall Using Fuzzy Time Series 124

Table 6.4: Root Mean Square Error (RMSE) Between Forecasted Rainfall Using The Rain Forecast Model (RFM) And Forecasted Rainfall Using Fuzzy Time Series 126

Table 6.5: Summary of Results 128

Table 6.6: Summary of Result of Comparison of Models 129

LIST OF SYMBOLS

FTL:	Fuzzy Technology Language
EFuNN:	Evolving Fuzzy Neural Network
FIS:	Fuzzy Inference System
ANFIS:	Adaptive NeuroFuzzy Inference System
TS:	Takagi-Sugeno
NFEG:	Neuro-Fuzzy Expectation Generator
NFE-editor:	NeuroFuzzy Expectations-editor
SANFIS:	Similar Adaptive NeuroFuzzy Inference System
nfMod:	neurofuzzy model
NFIDENT:	NeuroFuzzy Identification
GMDH:	Group Method for Data Handling
SST:	Sea Surface Temperatures
MIT:	Massachusetts Institute of Technology
ITD:	Inter Tropical Divergence
FNN:	Fuzzy Neural Network
EBP:	Error Back Propagation
MBF:	Membership Function
PID:	Proportional-Integral-Derivative
FAM:	Fuzzy Associative Map
MAX/MIN:	Maximum/Minimum Operator
MAX/PROD:	Maximum/Product Operator

DARPA	Defence Advanced Research Project Agency
DoS:	Degree of Support
BSUM:	Bounded Sum Operator
CoM:	Centre of Maximum
CoA:	Centre of Area
CoG:	Centre of Gravity
MoM:	Mean of Maximum
RH:	Relative Humidity
WD:	Wind Direction
WS:	Wind Speed
T-norm:	Triangular-Norms
T-conorm or S-norm:	Triangular-Conorms
WTA:	Winner Takes All
NiMet:	Nigeria Meteorological Agency
ASCII:	American Standard Code for Information Interchange
FDW:	Fuzzy Design Wizard
α - cut:	Alpha Cut
RFM:	Rain Forecast Model
3D:	Three-Dimensional
MSE:	Mean Squared Error

ABSTRACT

With the advent of modern computing facilities, there is the need to research into new forecasting techniques that are applicable to natural phenomena characterized by uncertainties in their problem formulation. A natural phenomenon of particular research interest is rainfall because of its critical importance to the agricultural economy and food security in Nigeria. The aim of this thesis is the application of the NeuroFuzzy Soft Computing tool and time series techniques in the field of forecasting and modeling of statistical phenomenon using rainfall forecasting of Zaria as a case study. The emerging NeuroFuzzy technique is a hybrid intelligent system unifying the benefits of the computational techniques of Fuzzy Logic and Artificial Neural Networks (Neural Nets): Herein the learning capabilities of Neural Nets and the representation power (knowledge base) of Fuzzy Logic Systems are optimally exploited. The goal of this research effort is to quantify the performance of the proposed technique when compared with other techniques using monthly thirteen-year data on weather variables collected from Nigerian Meteorological Agency (NiMet), Zaria and Department of Soil Science, Ahmadu Bello University, Zaria. The developed NeuroFuzzy System is implemented as a 'pure' Fuzzy Logic System using the Fuzzy Technology Language (FTL) as the Rain Forecast Model (RFM). A fuzzy time series technique is then applied to forecast rainfall so as to have a basis for comparison with the developed Rain Forecast Model (RFM). The developed model has been tested extensively relying on the validation data of 2003 to 2005. For the sake of comparison, existing fuzzy time series method has been applied for the same period. It is shown that the developed Rain Forecast Model (RFM) performed significantly better than the fuzzy time series method within the validation period based

on the root mean square returned by the two methods**.** Furthermore, the results obtained from the developed Rain Forecast Model (RFM) and the fuzzy time series methods for the entire forecast period of 2003 to 2010 are also compared. The results obtained are presented and discussed from the standpoint of a degree of consistency exhibited by the two methods and their computational time requirements.

CHAPTER ONE

INTRODUCTION AND BACKGROUND

1.0 INTRODUCTION

Soft Computing (or Computational Intelligence) is a field of knowledge in the realm of Artificial Intelligence that deals with machine incorporations of human expertise. Soft Computing tools include Fuzzy Logic, Artificial Neural Networks, Neuro-Fuzzy, and Rough Sets. These tools are generally robust as they can handle imprecise or noisy data, and non-linear data and are quite useful when the underlying relationships between the data are not fully understood or impractical to model. Unlike hard computing, which is conventional computing, that requires a precisely stated analytical model and often a lot of computation time, the soft computing technique is tolerant of imprecision, uncertainty, partial truth, and approximation. In essence the role model of the Soft Computing technique is the human mind.

The application of soft computing to forecasting is the main focus of this research. This introductory chapter, therefore, presents the motivations for this research; the relevant literature review of soft computing technique is also presented.

1.1 OVERVIEW OF SOFT COMPUTING TECHNIQUES

Machine intelligent behaviour is determined by the ability to realize machine incorporations of human expertise, flexibility of the architecture, laws of inference procedure and the speed of learning. All these form the main constituents of the field of

knowledge called **Computational Intelligence** or **Soft Computing** [1]."**Intelligent**" in this case refers to the utilization of engineering techniques that have, to one extent or another, been borne out of human reasoning, adaptation or learning, biological cognitive structures or principles of evolution [2].

Soft Computing can be defined as a consortium of methodologies that work synergistically and provides in one form or another flexible information processing capability for handling real-life ambiguous situations [3]. When exploited in a synergistic manner, Soft Computing tools can be used for construction of powerful computationally intelligent systems [4][5] that could possess human-like expertise within a specific domain, adapt themselves and learn to do better in changing environments, and explain how they make decisions [6][7]. Typical Soft Computing tools include Fuzzy Logic, Artificial Neural Networks, Neuro-Fuzzy and Rough Sets. They offer real advantages over conventional (hard computing) modeling including the ability to handle large amounts of dynamic, non-linear or noisy data and they can especially be useful when the underlying relationships are not fully understood [4][8].

Soft Computing techniques solve complex problems by utilizing the knowledge of an expert and incorporating the imprecision and uncertainty associated with various aspects of the problem [3][4]. This knowledge can be obtained by instruction and/or learning rather than by investigating the problem in details. The objective is getting an acceptable solution at minimum cost by seeking an approximate solution to an imprecisely or precisely formulated problem. In traditional hard computing, the main issues are

precision, certainty and rigorous computations while in contrast, in soft computing the principal notion is that precision and certainty carry a cost. As such the challenge is to exploit the tolerance for imprecision by devising methods of computation that lead to an acceptable solution at minimal cost [3]. And because many contemporary problems such as pattern information processing and classification (handwriting, speech, objects, images, etc.) and forecasting do not lend themselves to precise solutions, soft computing tools are becoming useful considering the following properties:

i) Ability to learn from experimental data; and

ii) Power of generalization, which is derived from approximating or interpolating to produce outputs from previously unseen inputs by using outputs from previous learned inputs.

If a network is trained on a data set and the error is e_1 and then a new data set is applied that produces error e_2, $|e_1 - e_2|$ is then a measure of the generalizing ability of the network [9]

Computation, reasoning and decision-making should exploit (wherever possible) the tolerance for imprecision, uncertainty, approximate reasoning and partial truths for obtaining low cost solutions. This is possible with Soft Computing taking the following features into consideration:

i) **Parallel Processing:** The mathematical power of machine intelligence is normally attributed to the neural-like system architecture used and the fault-tolerance arising from the massively interconnected structure [1]. Neural Nets, Fuzzy Logic and other hybrid Neuro-Fuzzy Systems are dynamic, parallel

processing systems that can estimate a function without any mathematical model and learn from experience with sample data [3].

ii) **Fuzzy Levels:** Instead of "**0**" and "**1**" digital levels, Soft Computing systems use fuzzy/continuous levels. This ensures that much more information is passed through the system [1].

iii) **Survivability:** Soft Computing systems can survive in the presence of faults. As such, they may work correctly if they are partially damaged [1] or are operating in an environment of uncertainty and imprecision. This is responsible for Soft Computing systems being able to understand distorted speech, decipher sloppy handwriting, recognize and classify images, summarize text and more generally making rational decisions in an environment of uncertainty and imprecision [3].

Fuzzy Logic is therefore a technology capable of modeling vagueness, handling uncertainty and supporting human-type-reasoning [3]. It allows representation of human decision and evaluation processes in algorithmic form (**IF** <situation> **THEN** <action>. Thus, if the desired performance of a technical system can be described for certain distinctive cases by rules, Fuzzy Logic can effectively put the knowledge into a solution [10].

A Fuzzy Logic System is developed in essentially three (3) steps, as shown in Figure 1.1:



Figure 1.1: Fuzzy Logic Block Diagram

i) **FUZZIFICATION** is using Fuzzy Sets to translate real values into linguistic variable values and using Membership Functions to graphically represent the variables.

ii) **FUZZY INFERENCE** is evaluating fuzzy '**IF** <situation> - **THEN** <action>' rules that define relationships between linguistic variables. The **IF** part is called the 'Antecedent' while the **THEN** part is called the 'Consequent'

iii) **DEFUZZIFICATION** is obtaining crisp output values from results of fuzzy inference

Fuzzy Logic Systems are designed using linguistic variables (first described by Lofti Zadeh in 1965). A linguistic variable can belong to more than one set in the universal Fuzzy Set of variables according to its membership degree (from 0 to 1). This is a major departure from conventional Boolean logic with only two states: High or Low. The fuzzy rule base is developed in a manner that 'nearly' reflects a human decision base for a similar situation.

Neural Net is an information processing technique that is inspired by the way biological nervous systems, such as the brain, process information. It is, therefore, a system composed of many simple processing elements (neurons) operating in parallel whose function is determined by network structure (e.g. feedforward) and connection (synapses) strength [11]. The Neural Net, just like the Fuzzy Logic System, also has three basic layers: INPUT LAYER, HIDDEN LAYER and OUTPUT LAYER as in Figure 1.2.

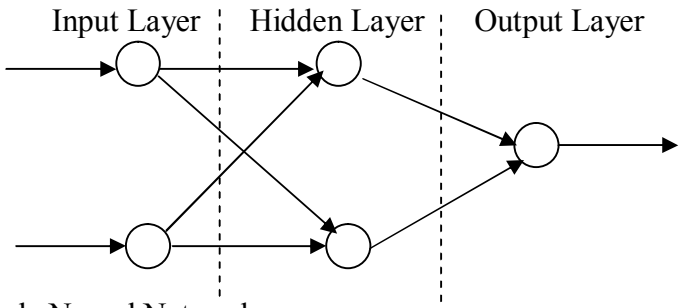


Figure 1.2: A Simple Neural Network

Developing a Neural Net solution means teaching the net desired behaviour using sample data sets of inputs and corresponding outputs. This is called the LEARNING PHASE. The Neural Net is created 'dumb' and it is the LEARNING ALGORITHM that modifies the individual neurons of the net and their connection weights in such a way that reflects the desired behaviour. After learning, the Neural Net enters into a WORKING PHASE, where its behaviour is now deterministic [10].

The basic idea of Neuro-Fuzzy approach is to set up a complete fuzzy rule base not by entering the rules but by presenting a set of training samples and selecting the set of rules that represent the samples. This is important because in many real world applications, knowledge that describes desired system behaviour is contained in data sets. It then means that the "IF <situation> THEN <action>" rules of a Fuzzy System must be derived from the data set manually, which can be difficult and time consuming especially with large data set. This is where a Neural Net presents a solution since it can train itself from data [10][12].

Fuzzy Logic Systems can be used for knowledge representation in Neural Systems because FUZZIFICATION can represent INPUT LAYER, FUZZY RULES can represent HIDDEN LAYER and DEFUZZIFICATION can represent OUTPUT LAYER. However, in each of these mechanisms, parameters must be found that can be used as weights in the Neural Net training. In a fuzzification (**input**) layer, each neuron represents an input Membership Function of the antecedent of a fuzzy rule. In a fuzzy inference (**hidden**) layer, fuzzy rules are fired and the value at the end of each rule represents the initial weight of the rule, and will be adjusted to its appropriate level at the end of the training. In the defuzzification (**output**) layer, each neuron represents a consequent proposition [3].

1.2 LITERATURE REVIEW

Forecasting is the process of generating information for the possible future development of a process from data about its past and its present whereas modeling is the process of finding global underlying structures, models and formulae that can explain the behaviour of the process in the long run and that can be used for long term prediction as well as understanding the past [25].

Methodologies, basically statistical, have and still are being used for forecasting. These include:

i) **Time-Series,** which is a method that uses a set of historical values at equally spaced time intervals to predict an outcome.

ii) **Regression Analysis,** which is a technique in which one random variable is predicted from the value of another random variable and/or several other random variables.

iii) **Group Method for Data Handling (GMDH),** which is a technique that is able to construct regressive mathematical model of high order that can accept a great number of variables. It divides the set of observations into 'Training' and 'Checking' sets.

iv) **Moving Average,** which is a technique used to forecast values based on a weighted average of past values [26].

These approaches are ill defined to represent vague input data and human judgment. They also have the major shortcoming of the difficulty in deriving mathematical models that approximate highly non-linear behaviour. The NeuroFuzzy methodology, as a Soft Computing tool, has the advantages of improving the adaptability of the forecasting system to sudden changes and the fact that it can simulate arbitrarily non-linear systems [13][27].

One of the most important functions inherent in human thinking is the ability to learn how to predict as no action is performed without predicting, in one way or the other the results of the action. The simplest method of predicting the future is based on the assumption that the future will be like the recent past and present [28]. The existence of past data constitutes observable time series and time-series forecasting assumes that a time series is a combination of a pattern and some random error. The goal is to separate

the pattern from the error by understanding the pattern's trend (its long-term increase or decrease), and its seasonality (the change caused by seasonal factors) [29].

The Neuro-Fuzzy technique as an evolving Soft Computing technique is being used to develop prediction models that are relatively more accurate and more robust than previous statistical time-series based models. This is because these models are able to handle noisy, vague and non-linear situations amongst others. The average forecasting error of the fuzzy time series approach is also smaller than that of most statistical time-series based models. Its main advantage is the fact that human experience and knowledge can be applied to the forecasting procedure.

Wang et al [13] used a Neuro-Fuzzy methodology that combined Neural Nets trained on historical data with PROTREN (a Fuzzy Logic algorithm that detects the onset of transients with extremely high efficiency and reliability) for nodal load forecasting. PROTREN extracts various features from the on-line signals and processes them in a fuzzy way to acquire the trend information that is needed by the Neural Nets for load forecasting. The methodology improves the adaptability of the system to sudden changes or special events that may influence the load by temporarily distorting the general pattern and thus making the load signal highly unpredictable.

Abraham et al [6][7] used an Evolving Fuzzy Neural Network (EFuNN) Neuro-Fuzzy methodology implementing a Mamdani type Fuzzy Inference System (FIS) to develop a monthly rainfall time series prediction model for Kerala, India. Based on information

from the previous four years, the network would predict the amount of rain in each month of the fifth year. In order to achieve good generalization properties, three-month information centred over the predicted month of the fifth year in each of the previous four years was used. The codes were executed using MATLAB® and C++.

Ahmad et al [14] used a Neural Net with Error Back-Propagation learning algorithm to forecast weather. The model developed is able to learn the pattern of rainfall in order to produce a precise forecasting result. The Neural Net technique is applied to the hourly local weather of Johor, Malaysia with 77-78% forecast accuracies reported. The reliability of the Neural Net approach is constrained by the fact that its operation cannot be explained in addition to the heavy computational requirements of the Error Back-Propagation algorithm.

Lotfi [15] used the learning Fuzzy Inference System (FIS) to forecast maximum daily electricity load based on previous data available for electricity load and average daily temperature. The problem was tackled in two different stages by creating two different models. The first model will predict the temperature and the second model uses the predicted temperature to forecast the maximum electricity load. Initial fuzzy rules are generated and then the numerical data from the Eastern Slovakian Electricity Corporation are used to learn the parameters of the learned Fuzzy Inference System,

Cuevas et al [16] used the Adaptive Neuro-Fuzzy Inference System (ANFIS) Takagi-Sugeno (TS) Neuro-Fuzzy model to predict target movements with minimum error for a

real time visual tracking system. Real time visual tracking is a complicated problem due to the dynamics of the objects involved in the process. The prediction model developed helps to reduce the delay's (caused by motors and mechanisms used for the camera's movements being significantly slow) effects in the control for visual tracking.

Kooths [17] used a Neuro-Fuzzy system called the Neuro-Fuzzy Expectation Generator (NFEG) connected to a business cycle simulation model using MAKROMAT-nfx software and Neuro-Fuzzy Expectations-editor (NFE-editor) in order to predict rate of inflation based on the current unemployment rate and the observable money growth rate for expectations modeling in macroeconomic theory. The software allows for analyzing how the Neuro-Fuzzy Expectation Generator (NFEG) interacts with the economic system when the system is exposed to shocks.

Lee et al [18] used a Similar Adaptive Neuro-Fuzzy Inference System (SANFIS) Mamdani Neuro-Fuzzy model to qualitatively evaluate a candidate's next moves in a 19 x 19 Go board game. This is a game for two players, who alternately play a single stone onto the 19 x 19 intersection (361 points), with the aim of surrounding more territory than the opponent.

This work, which aims to develop a rainfall forecast model for Zaria, as a case study, uses a modified neuro-fuzzy model (nfMod) based on Mamdani type Neuro-Fuzzy System, the fuzzyTECH Professional Edition V5.54h Neuro-Fuzzy module, Fuzzy Technology Language (FTL) and time series prediction techniques. According to **Gorzalczany et al**

[19], the neuro-fuzzy (nfMod) model is one of the least sensitive to removal of weakest rules from the rule base (using such techniques as α-cut (alpha-cut)) and is also better able to generalize from the learned rules than other such models like Adaptive Neuro-Fuzzy Inference System (ANFIS), Neuro-Fuzzy Identification (NFIDENT), etc. The neuro-fuzzy (nfMod) model is characterized by its high transparency and interpretability when compared from the other types of models.

A Time Series is a sequence of observations which are ordered in time and according to **Tomé et al [20] and Easton et al [21]**, time series prediction is a problem with wide range of applications, including Economics, Finance, Meteorology, Energy Systems Planning, Population or Traffic prediction amongst others. Typical statistical time series techniques include the smoothing techniques of Moving Average and Exponential Smoothing. According to **Easton et al [21]** and **Roberts [22]**, a moving average is a form of average which has been adjusted to allow for seasonal or cyclical components of a time series while exponential smoothing is used to reduce irregularities in the time series data thus providing a means of predicting future values of the time series.

Time series prediction is basically a modeling problem, which can better be solved using Soft Computing techniques. This is because it is possible that the underlying relationship between the data may not be known [17]. The first step in the solution is establishing a non-linear mapping between inputs and outputs, after which the model can be used to predict future values based on past and present observations [30].

The fuzzy time series technique used to forecast the rainfall is based on the technique adopted by **Abbasov et al [23]** in forecasting the annual population growth of Azerbaijan. **Lee et al [24]** also used a similar fuzzy time series model with the Box-Jenkins time series technique to evaluate the Taipei's monthly unemployment rate series.

1.3 STATEMENT OF THE PROBLEM

Weather is very difficult to predict despite the data and information that may be available. This is because weather can change very quickly (e.g. if the wind changes direction slightly, the rain may pass over and fall elsewhere) which means the forecast can quickly be out of date [31]. In general, climate and rainfall in particular, are highly non-linear phenomena in nature [32][7] exhibiting what is known as the **"Butterfly Effect".** Edward Lorenz (1961) first explained this butterfly effect whilst at Massachusetts Institute of Technology (MIT) [7} as "Small variations of the initial condition of a dynamical system may produce large variations in the long term behavior of the system". The phrase refers to the idea that a butterfly's wings might create tiny changes in the atmosphere that ultimately cause a tornado to appear (or prevent a tornado from appearing). The flapping wing represents a small change in the initial condition of the system, which causes a chain of events leading to large-scale phenomena.

As a result of this effect, while some regions of the world (e.g. Africa) are noticing systematic decrease in annual rainfall, others notice increase in flooding and severe storms (El-Nino, Hurricane Ivan, Tropical storm Jeanne, Hurricane Katrina etc in the Caribbean and America). Typical reported cases include:

i) El Niño warming in the Central Eastern Equatorial Pacific tends to cause droughts in Indonesia and Australia and floods in California.

ii) Above normal Sea Surface Temperatures (SSTs) off Angola tend to produce wetter conditions in Brazil's Nordeste [32]

Soft Computing techniques are well suited for developing the prediction model as a result of the non-linear nature of rainfall. The idea of using a Neuro-Fuzzy approach came both, from Neural Nets' abilities to learn and generalize from sets of historical patterns, and the complexities of the problem (unpredictability of rainfall), which is more suitable to be modeled using fuzzy techniques.

Rain is one of nature's greatest gifts and is very vital to the agricultural economies and food security of developing countries like Nigeria and also sustenance of the environment. In Nigeria, rainfall determines the zonal pattern of crops and the seasonal activities of farmers. It is then important that rainfall trends be identified as any unforeseen deviations can cause unwanted disruptions. This has assumed an even greater importance due to the threats posed by global warming and greenhouse effect [6].

Rainfall is governed by the interaction of the moist tropical maritime air mass and the dry, cool tropical continental air mass. Air mass is defined as a large uniform (with respect to temperature and water vapour) body of air within the atmosphere. Rainfall is characterized by its extreme variability, both of intensity and duration with a temporal and spatial pattern. The characteristic and intensity of the prevailing weather conditions are determined by the surface location of the moisture boundary zone separating the two

air masses. The boundary is known as Inter Tropical Divergence (ITD). It is the location of a place in relation to the position of the Inter Tropical divergence (ITD) that determines its weather situation [33].

Zaria is located within the Sudan Savannah zone lying on a plateau of about 670.56m above sea level and on latitude 11°8/N and longitude 7°41/E. It lies within a region with distinct dry and wet seasons, which are influenced by two distinct air masses. One from the North is dry and continental in origin and the other from over the Atlantic in the South is moist, cool and equatorial maritime. It is the decline in air pressure over land that shifts convection towards the land bringing in maritime moisture, rainfall and cloudiness [33]. The wet season occurs in the high sun period and is dominated by the South-West winds coming in around April or May and lasting till around October. About 65% of the rains occur between July and September. The dry season lasts between November and March and is practically rainless. The months of December to February, the harmattan season, are usually cold and dry due to the influence of the continental air mass (dry-dusty wind) from the desert regions of North Africa [33].

Using weather variables (Wind Direction, Wind Speed and Relative Humidity) collected from the Nigeria Meteorological Agency (NiMet), Zaria and the Meteorological Unit of the Department of Soil Science, Ahmadu Bello University, Zaria on a monthly average basis over a thirteen- (13) year period (1993 – 2005), a rainfall prediction model is to be developed. Data from 1993 to 2002 is used as the training set whilst the data from 2003 to 2005 is used as the validation data. However, since the exact relationships between

these variables are not known, data driven methods are more suitable in developing the prediction model. These methods perform a kind of function fitting by using multiple parameters on the existing information in order to predict the possible relationships in the near future. The non-linear relationships between rainfall and the weather variables used in this work are shown in the scatter diagrams of Figures 1.3 – 1.5 whilst their trends for the period 1993 – 2002 are shown in Figure 1.6. The scatter diagram, which is a graphical representation of the pairs of data, is a tool for making assessments about the relationships between random variables.

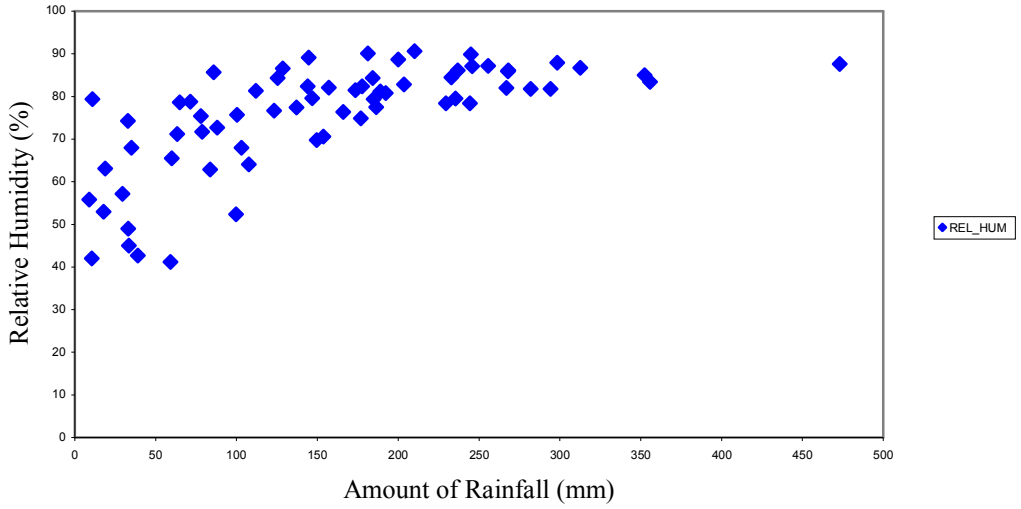


Figure 1.3: Plot of Relative Humidity v. Amount of Rainfall

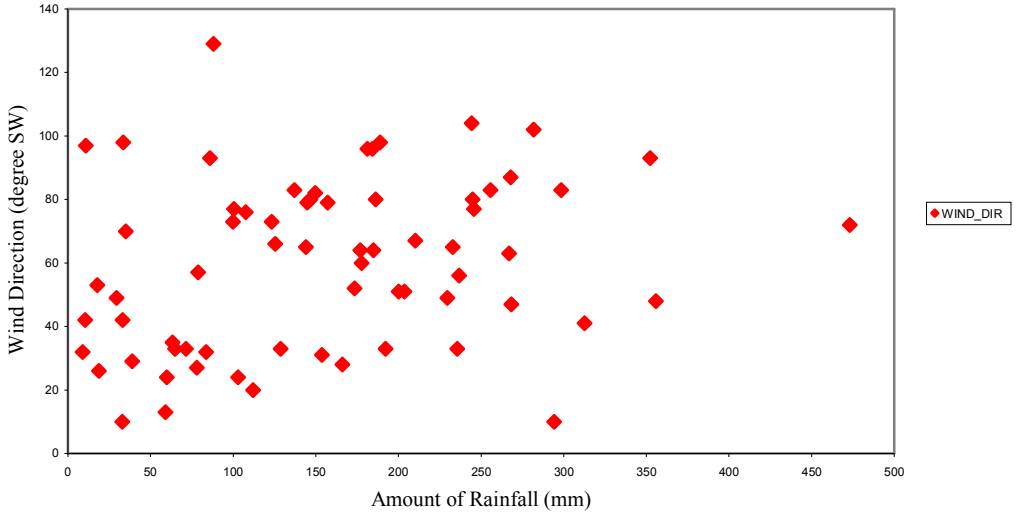


Figure 1.4: Plot of Wind Direction v. Amount of Rainfall

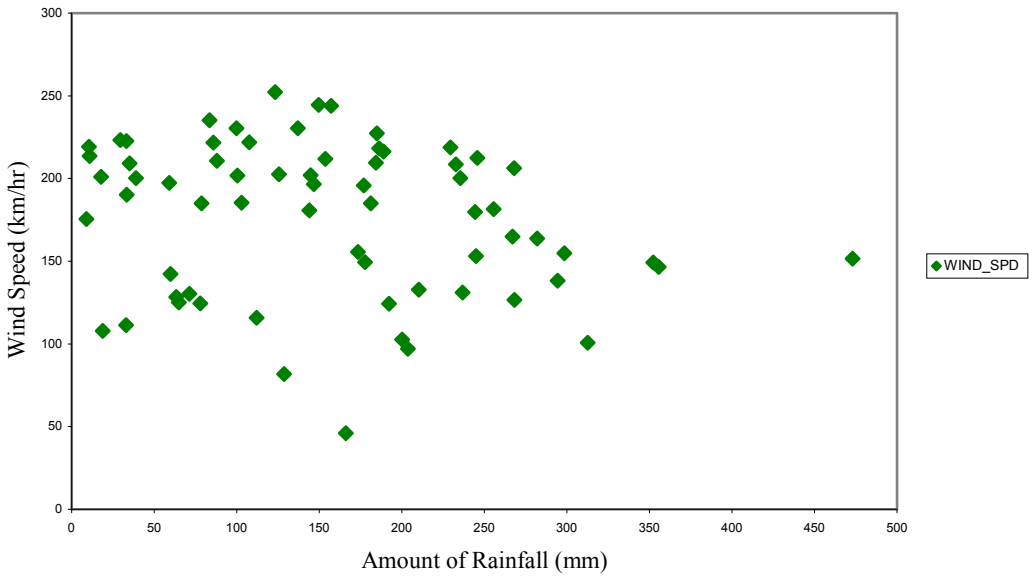


Figure 1.5: Plot of Wind Speed v. Amount of Rainfall

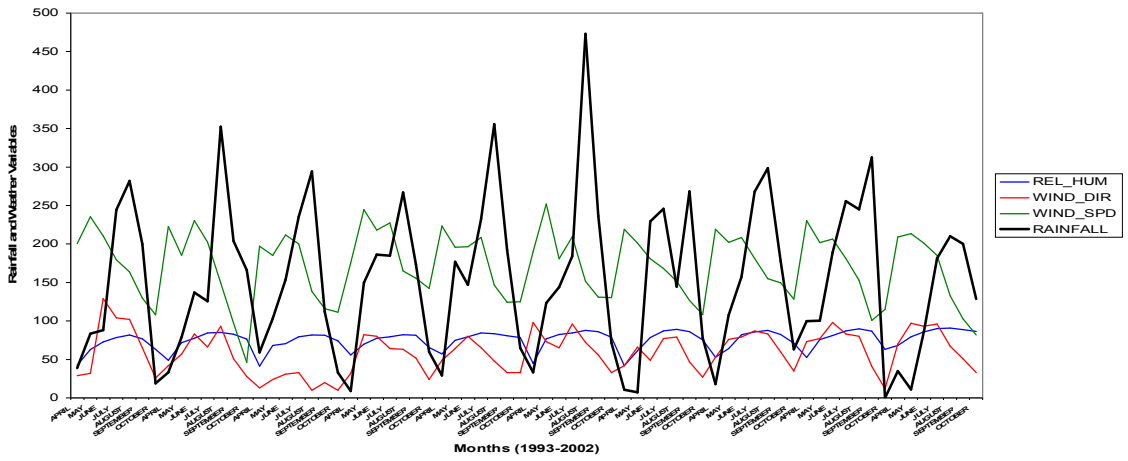


Figure 1.6: Rainfall and Weather Variables Trend For 1993 – 2002.

The Soft Computing model is trained on rainfall data corresponding to a certain period in the past (training set) and cross validation of prediction made by the network over some other period (validation set)[28]. One of such methods is Neuro-Fuzzy and the design issues with respect to Neuro-Fuzzy solution development include the following:

i) Number and types of input and output Membership Functions: How many Membership Functions can model a decision properly?

ii) Number of Rules

iii) Performance Function

iv) Optimization Method

v) Data Partitioning

vi) Number of Epochs [34][35].

The Neuro-Fuzzy Module provides training methods for supervised learning [36]. The standard method employed is a combination of Error Back Propagation and Competitive

Learning (Winner Neurons). After a system output is computed by forward propagation, an output error is identified by comparing the system output with the given sample output data [37][38]. The error is then used to determine the fuzzy rule most suited for influencing system behaviour. Using the selected Learning Rate, the plausibility of the fuzzy rule is modified before the subsequent sample data set is read [10].

1.4 RESEARCH OBJECTIVES AND METHODOLOGY

The objective of the research is applying the Soft Computing technique of Neuro-Fuzzy to develop a model to forecast (or predict) future values of an event, in this case the amount of rainfall, on a monthly average basis, using Zaria as a case study:

- The fuzzyTECH Professional Edition V5.54h Neuro-Fuzzy Module is used to carry out data training.
- The forecast model, Rain Forecast Model (RFM), is developed using the Fuzzy Technology Language (FTL).
- Time series techniques are used to predict the weather variables that are used with the forecast model to predict the rainfall.

The Neuro-Fuzzy Module uses an iterative computation [10] to find the Fuzzy System that best matches all given samples. In most cases, the samples must be trained several times in order to reach a satisfactory result.

The methodology adopted in data training and in developing the Neuro-Fuzzy System using the fuzzyTECH Neuro-Fuzzy module involves the following:

a) Obtain training sample data.

b) Cluster the sample data (if necessary). The data obtained may have to be preprocessed to remove redundant data and to resolve conflicts in the data.

c) Create an empty Fuzzy Logic System i.e. the rule set will have all rules with DoS (Degree of Support) = 0. This means the rule set is **COMPLETE** but **FALSE;** this is required as Neuro-Fuzzy training can only start with an existing rule set. A DoS value gives the weight for each value to be used in the rule aggregation step of fuzzy inference. The value is between 0 and 1.

d) Collection of expert knowledge about the process and entering all existing knowledge (if any) in the solution.

e) Selection of the components of the Fuzzy Logic System to be trained. All or specific components can be opened for learning.

f) Configuring the Neuro-Fuzzy module by specifying the **LEARNING METHOD** and setting parameters for it. The learning method can be any of **RealMethod, RandomMethod, Batch_Learn** and **Batch_Random.**

g) Train with the sample data to learn parameters of the Fuzzy System.

h) Evaluate system performance and validation of results. This is accomplished by testing trained system with sample **test data.** This will help in minimizing the occurrence of "**over training".**

i) Manual optimization using an interactive approach with the aid of the watch window to eliminate functionally redundant or unnecessary rules..

The methodology adopted in developing the forecast solution using fuzzy time series involves the following:

i) Definition of universal set U containing the interval between the minimum and maximum variations in the rainfall amounts for the month under consideration.

ii) Division of universal set U into equal-length intervals containing variation values corresponding to different rainfall amounts for the month under consideration.

iii) Determining the set of fuzzy sets or the respective values of the linguistic values.

iv) Fuzzifying the input data that is, converting numerical values into fuzzy values.

v) Selection of parameter ($w > 1$) corresponding to the time period prior to the concerned month and calculation of the relationships matrix.

vi) Defuzzifying obtained results, that is, conversion into quantitative values.

The overall methodology adopted in carrying out this investigation involves the following:

1) Data training and development of the Neuro-Fuzzy System using the fuzzyTECH Neuro-Fuzzy module, as described above.

2) Implementation as a pure Fuzzy Logic System using the Fuzzy Technology Language (FTL) resulting in the development of the Rain Forecast Model (RFM).

3) Validation of the Rain Forecast Model (RFM) using the validation data.

4) The weather variables, Relative Humidity, Wind Direction and Wind Speed are forecasted using the time series techniques of Simple Moving Average and Exponential Smoothing. The forecasted values are used to forecast for the rainfall using the Rain Forecast Model (RFM) developed.

5) Obtaining rainfall forecast values using fuzzy time series technique, which are compared with the values obtained from the Rain Forecast Model (RFM).

1.5 THESIS OUTLINE

This work is divided into seven chapters. Chapter One is the introductory chapter where the objective of the research is defined and the methodology to be applied explained. Chapter Two describes the theoretical background for the Soft Computing techniques of Neural Nets, Fuzzy Logic System and Neuro-Fuzzy Systems. The Rain Forecast model (RFM) is developed in Chapter Three, while Chapter Four describes the method used in forecasting the weather variables used as inputs for the developed model. The fuzzy time series method to be used for comparison purposes is described in Chapter Five while the obtained results are analyzed and compared in Chapter Six. The conclusion is in Chapter Seven. References and Appendices are provided at the end of the thesis.

CHAPTER TWO

SOFT COMPUTING TECHNIQUES

2.0 INTRODUCTION

According to Zadeh [39][40], Soft Computing is an emerging approach to computing that parallels the remarkable ability of the human mind to reason and learn in an environment of uncertainty and imprecision. The seamless integration of such methodologies as Neural Nets, Fuzzy Logic, and Rough Sets etc forms the core of Soft Computing; the synergism allows Soft Computing to incorporate human knowledge effectively, deal with imprecision and uncertainty and learns to adapt to unknown or changing environment for better performance. This work is based on Neuro-Fuzzy, which is an amalgamation of the strengths and weaknesses of Neural Nets and Fuzzy Logic Systems.

2.1 ARTIFICIAL NEURAL NETWORK

According to Defence Advanced Research Project Agency (DARPA) Neural Net study, "A Neural Net is a system composed of many simple processing elements operating in parallel whose function is determined by network structure, connection strengths, and the processing performed at computing elements or nodes" [41]. An Artificial Neural Network (henceforth referred to as Neural Net) is an information processing system that tries to replicate the behaviour of a human brain by emulating the operations and connectivity of a biological neuron [42][5]. The human brain contains about 10^{11} nerve cells of more than twenty types, with about 10^{14} connections (synapses) to one another [10][42]. The study of brain-style computation has its roots in the work of McCulloch and

Pitts (1943), where the output of the processing unit is described as a 'squashed' linear function of the inputs [43][44]. This is described by the expression:

$$a_i \leftarrow g(in_i) = g\left(\sum jw_{j,i}a_j\right) \qquad 2.1$$

(where w_{ji} = Weight in the first layer (from input i to hidden unit j)) and Figure 2.1 [40].

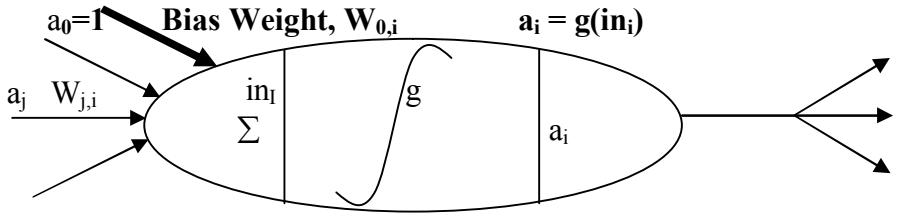


Figure 2.1: McCulloch-Pitts Unit

Each processing unit (neuron) is characterized by an activity level (representing the state of polarization of a neuron), an output value (representing the firing rate of the neuron), a set of input connections (representing synapses on the cell and its dendrites), a bias value (representing an internal resting level of the neuron) and a set of output connections (representing a neuron's axonal projections) [47]. A simplified model of a human neuron is as shown in Figure 2.2 [45][46].

The transmission of signals from one neuron to another is a complex chemical process in which specific transmitter substances are released from the sending end of the junction. The effect is to raise or lower the electrical potential inside the body of the receiving cell. If the potential reaches a threshold, a pulse is sent down the axon and a cell is said to have 'fired' [47]. From Figure 2.2, synapses exist to increase or decrease the activation of neurons as a result of inputs from other neurons. Each neuron has an activation level that ranges between a maximum and a minimum, which is in contrast with Boolean logic that

has only two logic levels (0 or 1). The output of each neuron is transferred by an axon, which ends in several other synapses influencing other neurons. Synapses carry an activation level from a sending neuron to a receiving neuron. For an exciting synapse, the activation level from the sending neuron increases the activation of the receiving neuron while for an inhibiting synapse, the activation from the sending neuron decreases the activation of the receiving neuron [10].

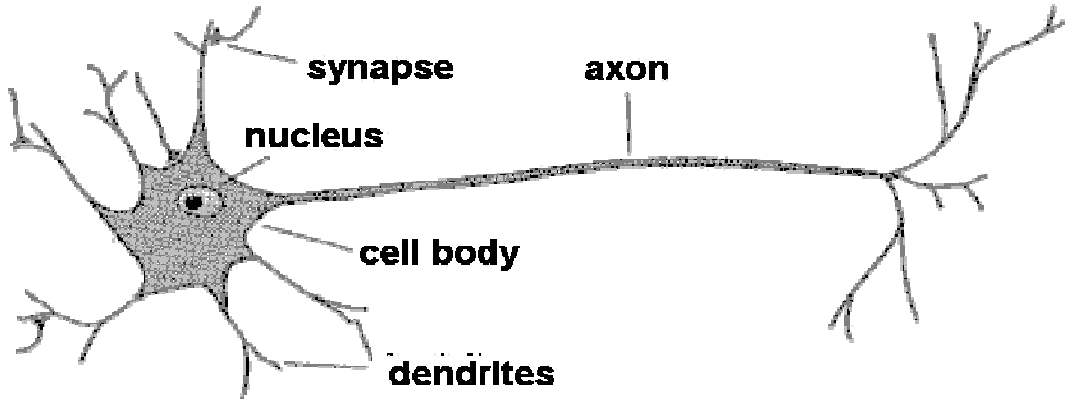


Figure 2.2: Simplified Human Neuron Model.

Neural Nets use simple computational units called neurons and interconnecting many of these neurons in a known configuration creates a Neural Net model [8]. The mapping of inputs to outputs is performed by a series of simple processing nodes distributed in a set of interconnected layers. The connections (synapses) are associated with adaptable weights that change as the network learns the correct patterns [5].

The broad set of characteristics that both real (biological) and Neural Nets share is the ability to learn and adapt, generalization to unseen data, distributed processing, robustness and fault tolerance [5]. However, Neural Nets need not be initially supplied

with expert knowledge and can be applied to virtually any data set without presuppositions concerning the distribution of the training data [48]. It is, however, essential that the Neural Net consist of the following three elements:

i) An organized architecture of interconnected processing elements.

ii) A method of encoding information during training.

iii) A method of recalling information during testing [49]

2.1.1 Neural Net Mathematical Model

Neural Nets can handle complex input/output relationships without detailed mathematical models thus avoiding modeling difficulties associated with complex physical systems [50]. From a mathematical point of view, Neural Net is a complex non-linear function with many parameters that can be adjusted (calibrated or trained) in such a way that the output becomes similar to the measured output on a known data [42]. This means that the Neural Net is able to generalize relevant output for a set of previously unseen input data [6]. In essence, Neural Nets can be considered as universal approximators of non-linear dependencies trained by experimental data [51]. Neural Nets can, therefore, be trained to approximate any continuous function to any desired accuracy, without a need to specify its type. They can also be applied to incomplete or corrupted data and still yield acceptable results [48] because the Neural Nets are relatively fault-tolerant having many processing nodes [52].

The most common mathematical model based on the simple neuron model is as shown in Figure 2.3. McCulloch-Pitts [43], Isik [45], Chujo et al. [53] and Gupta and Rao [54]

have also proposed other models. In a simplified mathematical model of the neuron, the effects of the synapses are represented by 'weights' that modulate the effects of the associated input signals, and the non-linear characteristics exhibited by neurons are represented by a transfer function, usually the Sigmoid function. The neuron impulse is then computed as the weighted sum of the input, transformed by the Sigmoid function [34].

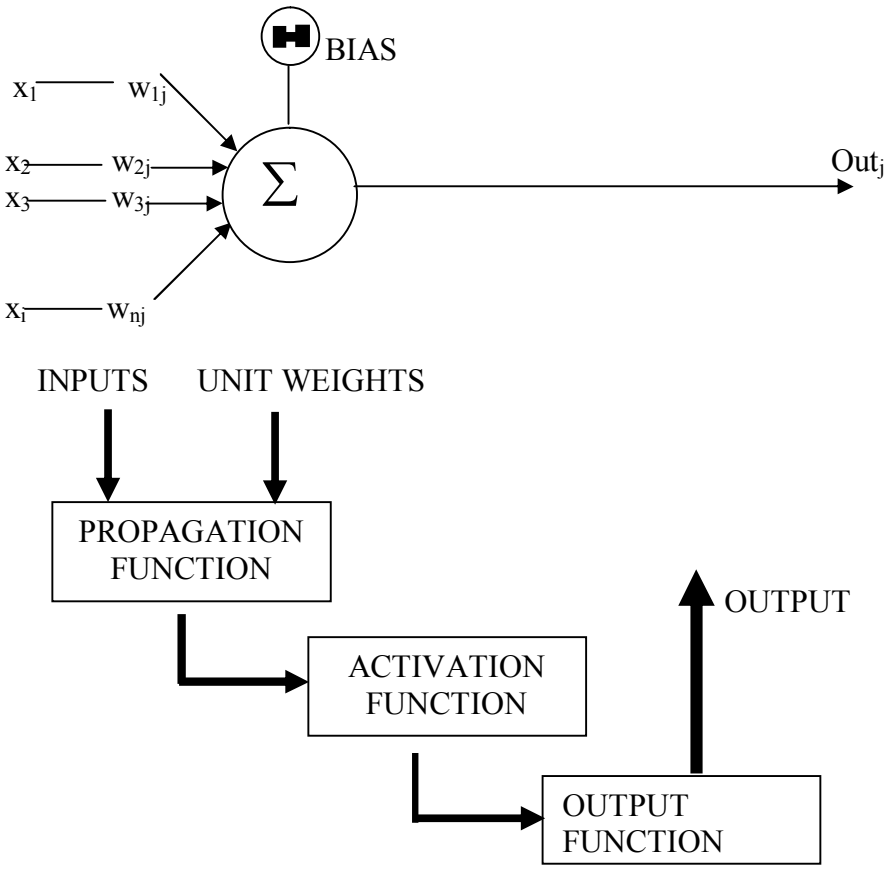


Figure 2.3: Basic Mathematical Model of A Neuron.

Information is first given to the system via the input units, then processed through the net and read from the output units. The system is described by the functions of the various

units in the model. The result of the Propagation Function is used to compute the activation of the neuron with the Activation Function. The most common Activation Function is the Sigmoid Function. The Output Function then further processes the calculated result generated by the Activation Function. This is required to provide additional filtering of the output information of every unit [10].

2.1.2 Neural Net Classifications

Neural Nets consist of several layers of simple processing elements called Neurons that form a highly parallel computational network [55][46]. Neural Nets are broadly classified into three classes:

i) Feedforward Network: All of the hidden and output layer neurons receive inputs from the preceding layer only. There is the Single-Layer and Multi-Layer type.

ii) Recurrent Network: This has at least one feedback loop i.e. cyclic connection, which means that at least one of its neurons feed its signal back to the inputs to all other neurons. The behaviour of this type of network can be extremely complex. The Hopfield, Holographic Associative Memory and Boltzman networks are all examples of recurrent Neural Nets.

iii) Lattice Structure: This has output neurons arranged in rows and columns [56][43].

The feedforward Neural Net is the most common with only a unidirectional (from left to right) signal flow [1][46][57] and is composed of a hierarchy of processing units, organized in a series of two (2) or more mutually exclusive sets of layers (no feedback within layers or to layers proceeding that under consideration)[9]. The feedforward

Neural Nets are organized in layers: Input Layer (i), Hidden Layer (h) and output layer (o) [1][57] as in Figure 2.4.

The interconnections within the network are such that every neuron in each layer is connected to every neuron in the adjacent layers [48]. Each node in a layer receives its input from the output of the previous layer nodes or from the network input. The connections between nodes are associated by scalar connection (synaptic) weights (w) that are iteratively adjusted during the training processes [58][37]. An additional node with a constant output (usually 1) is often added to the input and hidden layers and they are called the Bias Nodes [58].

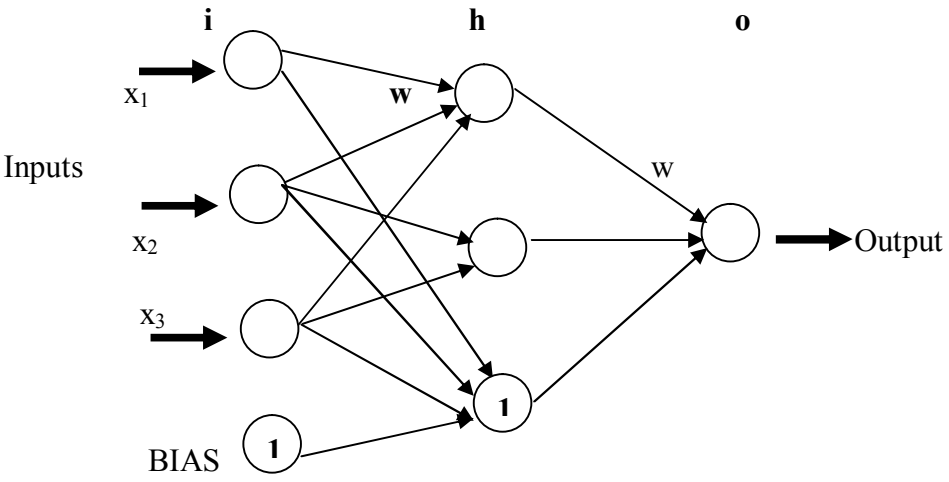


Figure 2.4: Simple Feedforward Neural Net With Three Layers

The basic feedforward network forms a non-linear transformation of input data in order to approximate the output data. Because of the nature of the modeling problem being tackled, the output data representation and the form of the network output required determine the number of input and output nodes. Determining the number of neurons that

a hidden layer must have in order to carry out the given task is a very critical aspect of Neural Net solution [59]. Having too few neurons in the hidden layer can result in "Underfitting", that is, too few neurons to adequately detect the signals in a complicated data set. Having too many neurons on the other hand can result in "Overfitting" that is, having so much information processing capacity that the limited amount of information contained in the training set is not enough to train the neurons in the hidden layer [57]. The following are compromise rule-of-thumb for determining the correct number of neurons to use in the hidden layer:

i) The number of hidden neurons should be two-thirds $(\frac{2}{3})$ of input layer size plus the size of the output layer or

ii) The number of hidden neurons should be less than twice the input layer size [57].

An impediment to the widespread acceptance of Neural Nets is the absence of a capability to explain to the user, in a mathematical form, how it arrives at a particular decision. Neither can something be said about the knowledge encoded within the 'Black Box' (hidden layer). This is because the hidden layer, whose number of hidden layer nodes is related to the complexity of the system being modeled [37][57], does not interact with the outside world. This is the reason why the Neural Net is essentially a 'Black Box' as only the input/output (I/O) behaviour is observed. Fuzzifying a Neural Net is one way of extracting knowledge from the Neural Net for users to have a better understanding of the prediction process. Rules are a form of knowledge that humans can easily understand

and verify. Thus a Fuzzy Neural Network (FNN) is a Neural Net equipped with the capability of handling fuzzy information [3][57].

Problems that require more than two hidden layers are rarely encountered and for most practical problems, there is no strong justification to use more than one hidden layer [57]. This is because the technique is aimed at obtaining approximate solutions and not precise ones and as such using one hidden layer is more often than not sufficient. Table 2.1 relates the number of hidden layers and situations where they can be applied. They assist in performing classification and non-linear feature extraction tasks on information provided by the input and output layers [57].

Table 2.1: Number of Hidden Layers And Situations Where They Can Be Applied

Number of Hidden Layers	Applications
0	Only capable of representing linear separable functions or decisions
1	Can approximate arbitrarily any functions that contain continuous mapping from one finite space to another
2	Can represent an arbitrary decision boundary to arbitrary accuracy with rational Activation Functions and can approximate any smooth mapping to any accuracy.

2.1.3 Activation Function

The Activation Function finds the weighted average of all input elements in each processing node. It multiplies the input values (x_i) by weights (w_i) and adds them together to obtain a weighted sum, and for d inputs into a processing element,

$$Y(x) = \sum_{i=0}^{d} x_i w_i \qquad 2.2$$

Where d = the dimension of input space [52].

This function calculates the internal simulations or activation level of the neuron. Based on the calculated level, the neuron may or may not produce an output [52]. The neurons are interconnected in such a way that information relevant to the input/output (I/O) mapping is stored in the weights, as each neuron contains the weighted sum of its inputs filtered by a Sigmoid Function [55]. This means that each hidden and output node is associated to an Activation Function [58].

The Activation Function to be used in Neural Nets must be continuous, differentiable and monotonically non-decreasing. Differentiability is essential because the derivative of the Activation Function is used to compute the weight updates (i.e. backpropagate the error) [60]. The most common Sigmoid Function used in Neural Nets is the Logistic Sigmoid shown in Figure 2.5.

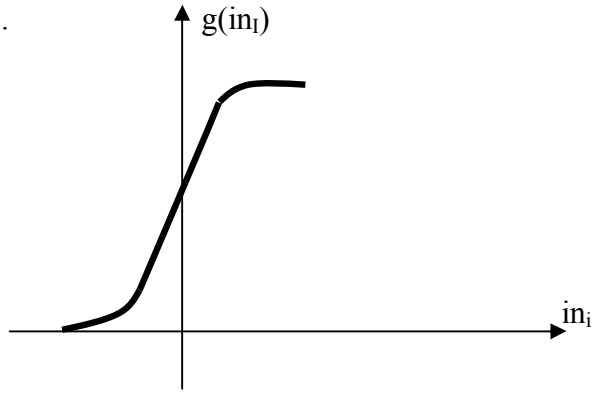


Figure 2.5: Logistic Sigmoid

The Logistic Sigmoid is used to introduce non-linearity in the model. It shows early exponential growth for negative in_i, which slows to linear growth of slope ¼ near $in_i = 0$, then approaches $g(in_i) = 1$ in an exponentially decaying manner [64]. It is mathematically expressed as:

$$g(v) = \frac{1}{1 + e^{-v}} \qquad 2.3$$

The expression for the weighted sum Y(x) can be generalized using a monotonic linear or non-linear Activation Function that acts on a weighted sum as:

$$Y(x) = g\left(\sum_{i=0}^{d} x_i w_i\right) \qquad 2.4$$

Where g(x) = Logistic Sigmoid

A function g: R→R is called a Sigmoid Function, if the following criterion is satisfied:

$$\begin{cases} \underset{x \to -\infty}{Lim}\, g(v) = 0 \\ \underset{x \to \infty}{Lim}\, g(v) = 1 \end{cases} \qquad 2.5$$

If $|v|$ is small then g approximates a linear function

The Sigmoid Function is very useful because:

i) It is non-linear,

ii) It is monotonic,

iii) It is bounded above and below (it will never approach infinity no matter the value of "v"), and

iv) It has a simple derivative ($g'(v) = g(v)(1 - g(v))$) [56][62].

A Neural Net activated by a Sigmoid Function is shown in Figure 2.6. The summing junction of the hidden unit is obtained by the following weighted linear combination:

$$v_j = \sum_{i=0}^{d} w_{ji} x_i \qquad 2.6$$

Where w_{ji} = Weight in the first layer (from input i to hidden unit j) and w_{jo} = Bias for hidden unit j.

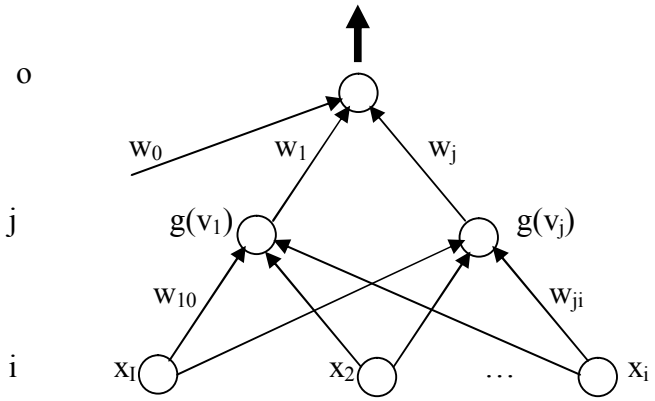


Figure 2.6: Neural Net Activated With The Sigmoid Function

The activation (output) of the hidden unit j is then obtained from:

$$o_{hiddenj} = g(v_j) \qquad 2.7$$

For output of whole network the following activation is constructed:

$$Y = g\left(\sum_{i=0}^{k} w_j o_{hiddenj}\right) \qquad 2.8$$

2.1.4 Neural Net Learning: The Error Back Propagation

The problem of learning in Neural Networks is that of finding a set of connection strengths (weights) that allow the network to carry out the desired computation. The network is provided with a training set and its role is to modify its connections in order to approximate the function from which the training set was drawn. The network is then tested for ability to generalize [44].

A Neural Net works at its best if all its synaptic weights have been properly adjusted. If the hidden layer node is non-linear, an Error Back Propagation (EBP) Neural Net is able to approximate its function [58]. The Error Back Propagation algorithm is a way to compute these weights and involves four steps:

i) The network is initialized by assigning random values to synaptic weights;

ii) A training pattern is fed and propagated forward through the network to compute an output value for each output node;

iii) The computed outputs are compared with the expected (target) outputs and a match is computed;

iv) A backward pass through the network is performed, changing the synaptic weights to some of its connections on the basis of the observed output errors, if the output differs from the target. If, however, the output and target match, no change is made to the net [58][44].

Steps (ii) through (iv) are iterated for each pattern in a training set. The network performance is then checked (usually on the basis of a mean squared error) and a new set of training patterns is submitted to the network (i.e. a new epoch is started) if it needs further optimization [58].

In the case of Neural Networks with a single hidden layer, the forward propagation step is carried out as follows:

$$h_k = \sum_{j=1}^{n_1+1} i_j w_{jk} \quad ,(\mathrm{k} = 1, 2, \ldots, \mathrm{n_k}) \qquad 2.9$$

where i_j are the outputs of the input layer (i.e. the network inputs and 1 for the bias node) and w_{jk} are the weights of the connections between input and hidden layers. To compute

the outputs of the hidden layer, these weighted sums are passed to the Activation Function, but for the Bias node, that is forced to have an output equal to 1 [58][44]:

$$h_k = f(h_k)$$
$$h_{n_k+1} = 1 \qquad 2.10$$

Then, the network outputs are computed in the same way:

$$o_i = \sum_{k=1}^{n_k+1} h_k z_{kl} \text{ , (k = 1, .., } n_o) \qquad 2.11$$

$$o_l = f(o_l) \qquad 2.12$$

After the forward propagation, estimated outputs o_l are compared with expected outputs y_l and a mean quadratic error for the current pattern is computed as:

$$E = \frac{1}{n_o} \sum_{l=1}^{n_o} (y_l - o_l)^2 \qquad 2.13$$

Then, in the back-propagation step, all the synaptic weights are adjusted in order to follow a gradient descent (in order to minimize the quadratic error) on the error surface. For the connections between hidden and output layers, z_{kl} are changed as:

$$z_{kl} = z_{kl} + \eta \delta_l^o h_k \text{ , (k = 1, …, } n_k+1; \text{ l = 1, …, } n_o) \qquad 2.14$$

where η is a constant (learning rate) and:

$$\delta_l^o = (y_l - o_l) f'(o_l) = (y_l - o_l) o_l (1 - o_l) \qquad 2.15$$

The Learning Rate is continuously adjusted during training to allow large initial corrections while avoiding weight oscillations around the minimum when converging near the solution. The use of Learning Rate less than 1 initially enables the network to slowly converge. If the value of Learning Rate is too high, the network may oscillate and become unstable and if it is too small, the network will take too long to converge [9].

The weights w_{jk} of the connections between hidden and input layer are also adjusted:

$$w_{jk} = w_{jk} + \eta\delta_k^h i_j, \text{ (k = 1, …, } n_k\text{+1; j= 1, …, } n_I\text{+1)} \qquad 2.16$$

where

$$\delta_k^h = f'(h_k)\sum_{l=1}^{n_o}\delta_l^o z_{kl} = h_k(1-h_k)\sum_{l=1}^{n_o}\delta_l^o z_{kl} \qquad 2.17$$

The network training is iterated until the STOP condition is met [58]. The basic flowchart for the error back propagation is shown in Figure 2.7

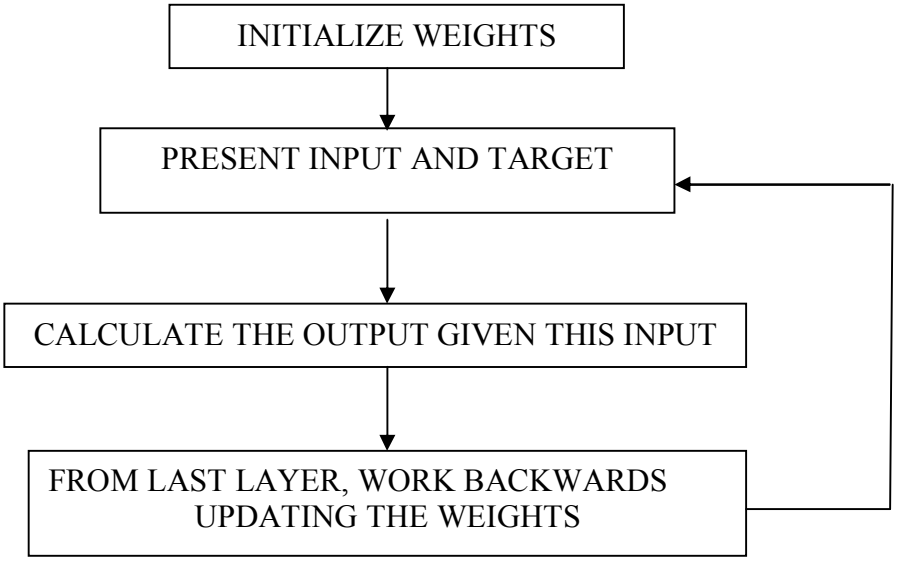


Figure 2.7: The Basic Flowchart for the Error Back Propagation

2.2 FUZZY LOGIC SYSTEM

The Fuzzy Logic System is a computing framework based on the concepts of Fuzzy Set theory, fuzzy **IF-THEN**, and fuzzy reasoning. It has applications in such areas as automatic control, data classification, decision analysis, time series prediction, robotics, and pattern recognition. Its basic structure consists of three components: a Rule-Base, which contains a selection of fuzzy rules; a Data-Base, which defines the membership

functions (MBF) used in the fuzzy rules; and a Reasoning Mechanism, which performs the inference procedure [40].

2.2.1 Mathematical Uncertainty

Plato first proposed that there was a third region between 'TRUE' and 'FALSE' called the 'UNCERTAIN' [40]. Many mathematical disciplines deal with the description of uncertainties, such as Probability Theory, Information Theory, Fuzzy Set Theory, etc. These can broadly be classified as **Stochastic** and **Lexical** [10].

Stochastic methods deal with the uncertainty towards the occurrence of a certain event in which the event is well defined and the uncertainty is quantified by a degree of probability. Lexical Methods deal with the imprecision that is inherent in most words human use to evaluate concepts and derive conclusions. In this case, the definition of the event is subjective and the uncertainty does not have quantitative content [10].

2.2.2 Fuzzy Set Theory

Fuzzy Logic was first proposed in 1965 by Lofti Zadeh [39] based on Fuzzy Set - part of conventional Set Theory that operates on the range 0.0 - 1.0 [63]. Fuzzy Logic was specifically designed to mathematically represent uncertainty and vagueness and to provide formalized tools for dealing with the imprecision intrinsic to many problems [67]. Fuzzy Sets are a true generalization of conventional sets [10] and they have ambiguous boundaries and therefore gradual transitions between defined sets. These transitions allow the uncertainty and ambiguity associated with the concepts that they

represent to be modeled directly [5][3]. It means that Fuzzy Sets can deal with imprecise objects whose behaviours are too complex or ill defined to admit precise mathematical analysis [50][65].

The use of Fuzzy Sets defined by Membership Functions (MBF) in logical expressions is called Fuzzy Logic. The Membership Functions (MBFs) define the degree to which the value of a variable belongs to the universal set and is usually a linguistic term, for example Low, High, etc. The linguistic terms normally describe a concept related to the value of the linguistic variable. The degree of membership in a set becomes the degree of truth of a statement [1]. This is the reason why in Fuzzy Logic, everything is a matter of 'degree' and as such exact reasoning is viewed as a limiting case of approximate reasoning [64].

A Fuzzy Set A in X, where X is a nonempty set, is characterized by its Membership Function:

$$\mu_A(x): X \to [0,1] \qquad 2.18$$

and $\mu_A(x)$ is interpreted as the degree of membership of element x in Fuzzy Set A for each $x \in X$ [64].

2.2.2.1 Fuzzy Set Operators And Hedges

The basic operators: Intersection (AND), Union (OR) and Complement (NOT) also exist for Fuzzy Set but are defined differently as follows:

AND: The intersection of A and B (Figure 2.8a) is defined as:

$$\mu(A \cap B) = \mathbf{Min}\{\mu(A), \mu(B)\} \quad 2.19$$

OR: The union of A and B (Figure 2.8b) is defined as:

$$\mu(A \cup B) = \mathbf{Max}\{\mu(A), \mu(B)\} \quad 2.20$$

NOT: The complement of a Fuzzy Set A is defined as:

$$\mu(A') = 1 - \mu(A). \quad 2.21$$

Where μ = Membership Function. This is defined over the range of input and output variable values and linguistically describes the Universe of Discourse [66].

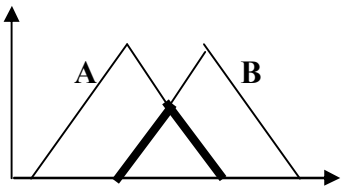


Figure 2.8: a) Intersection of A and B

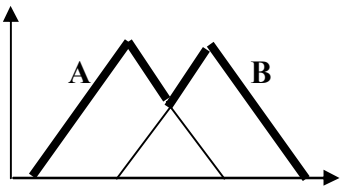


Figure 2.8: b) Union of A and B

Hedges are operators that are independently created to modify fuzzy values. These operations are provided in an effort to maintain close ties to natural language, and to allow for the generation of fuzzy statements through mathematical calculations [66]. Typical hedges include:

'Very ': $\mu(\text{'very' } A) = \mu(A)^2$ 2.22

'Somewhat': $\mu(\text{'somewhat' } A) = \mu(A)^{0.5}$ 2.23

'Extremely': $\mu(\text{'extremely' } A) = \mu(A)^3$ 2.24

2.2.3 Fuzzy Logic System

The use of linguistic modeling instead of mathematical modeling greatly enhances system characterization and modification potential leading to quick development cycles. Fuzzy

Logic Systems always remain plain and simple to understand, as the 'intelligence' of a system is not hidden in differential equations or source code [42].

The primary building block of any Fuzzy Logic System is the linguistic variable and the possible values of a linguistic variable are represented by linguistic terms [10]. The knowledge base of the Fuzzy Logic System is composed of a database and a rule base. The database defines the Membership Functions for the linguistic terms and the rule base represents fuzzy control rules. Therefore, the Membership Functions and the fuzzy rules greatly affect control performance [50][5]. Rules are influential in selecting the number of variables and Membership Functions to be modeled with Fuzzy Logic because the model becomes exponentially more complex as the number of variables or Membership Functions increase [68]. This is because a rule must be available for each possible combination of input variable Membership Functions (MBFs) and potential output Membership Functions (MBFs).

A typical structure of the Fuzzy Logic (controller) System is shown in Figure 2.9. For control purposes, it is an appealing alternative to such conventional methods as PID (Proportional-Integral-Derivative) control and model-based control. This is especially true where system follows some general operating characteristics and a detailed process understanding is unknown or traditional models become overly complex [66]. Conventional PID fails if the control task is non-linear, and model-based methods also suffer considering the kinds of assumptions that may be made in deriving the model, and the kinds of differential equations that may be derived [10].

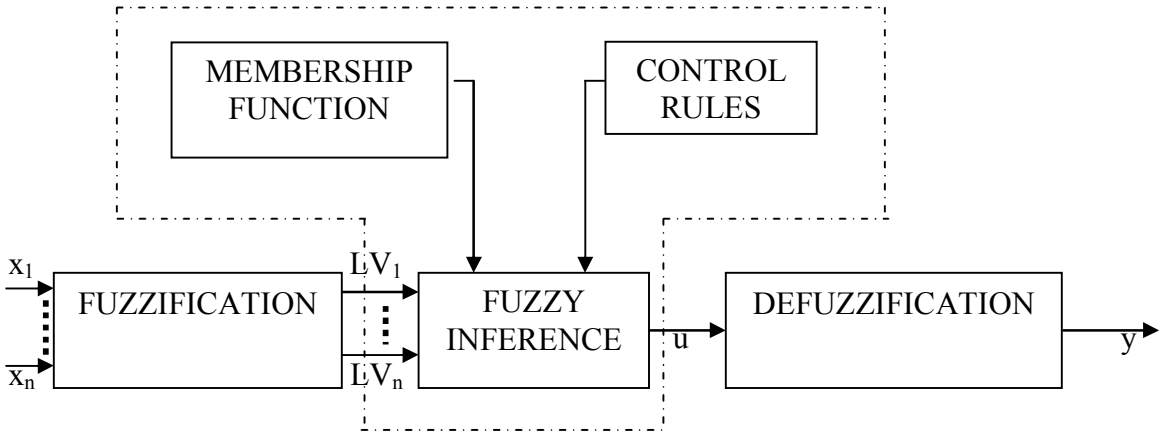


Figure 2.9: Typical Fuzzy Logic Controller Block Diagram

Fuzzy Logic Systems are broadly grouped into two: MAMDANI - and TAKAGI-SUGENO (TS) Types.

i) **MAMDANI -TYPE:** This includes linguistic models based on collections of IF-THEN rules, whose antecedents and consequents utilize fuzzy values. It uses fuzzy reasoning and the system behaviours can be described in natural terms [3]. It is generally expressed as:

$$R^i : If\ x_1\ is\ A_1^i\ AND\ x_2\ is\ A_2^i\ AND\ ...\ AND\ x_n\ is\ A_m^i\ THEN\ y^i\ is\ B^i \qquad 2.25$$

Where R^i (i = 1, 2, …,l) = the i^{th} fuzzy rule

x_j (j=1, 2, …, n) = the input of the fuzzy rule R^i

A^i_1, A^i_2…, A^i_m , B^i (I = 1, 2, …, l) = Fuzzy Membership Functions, usually associated with linguistic terms [3].

ii) **TAKAGI-SUGENO (TS) TYPE:** This uses a rule structure that has fuzzy antecedents and functional consequents. It approximates a non-linear system with

a combination of several linear systems, by decomposing the whole input space into several partial fuzzy spaces and representing each output space with linear equations. It is generally expressed as [3][69].

$$R^i : If\ \ x_1\ is\ A_1^i\ \ AND\ \ x_2\ is\ A_2^i\ \ AND\ \ ...\ \ AND\ \ x_n\ is\ A_m^i\ \ THEN\ \ \ y^i = a_0^i + a_1^i x_1 + ...a_n^i x_n$$

2.26

If a precise solution is required without much bother for interpretability, then the Takagi-Sugeno type systems are used otherwise the choice is the Mamdani type [3].

2.2.4 Fuzzy Logic System Layer Structure

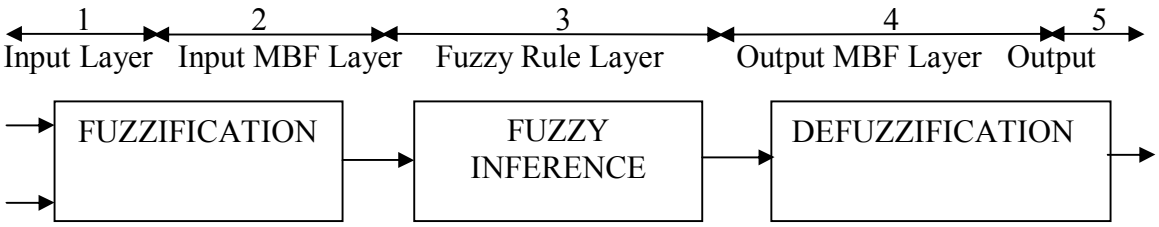


Figure 2.10: Schematic of Fuzzy Logic System Layer Structure.

Figure 2.10 shows the schematic of the Fuzzy Logic System layer structure. Nodes at the Input Layer are input nodes that represent input linguistic variables. Nodes at Layers 2 and 4 act as Membership Functions. Each node in Layer 3 is a rule node, which represents one fuzzy rule. Thus, all the Layer 3 nodes form a Fuzzy Logic rule base. Links between Layer 2 and Layer 3 define preconditions of fuzzy rules and links between Layer 3 and Layer 4 define the consequences of fuzzy rules. The links in Fuzzy Logic Systems do not have weights (unlike Neural Nets), but the order of connection is important as it defines knowledge of the system [70].

A Fuzzy Logic System is developed in essentially three (3) steps: Fuzzification, Fuzzy Inference and Defuzzification.

2.2.4.1 Fuzzification

This translates a numeric value into a linguistic value with a membership grade [69]. It is, therefore, a mapping from the observed numerical input space to the Fuzzy Sets defined in the corresponding Universe of Discourse. The fuzzifier maps a numerical value denoted by $x_i = (x_1, x_2, \ldots, x_n)$ into Fuzzy Sets represented by Membership Functions [71]. The Membership Function represents the degree to which crisp values belong to a given Fuzzy Set.

Technical quantities are represented on the horizontal axis of the function and membership degrees on the vertical axis. The technical quantity is called the Base Variable and represents the Universe of Discourse. The standard Membership Functions used in most practical applications include Z, Lambda, Pi and S types and they are shown in Figure 2.11 [10]. Regardless of the shape of the Membership Functions, they can be adjusted during training [15].

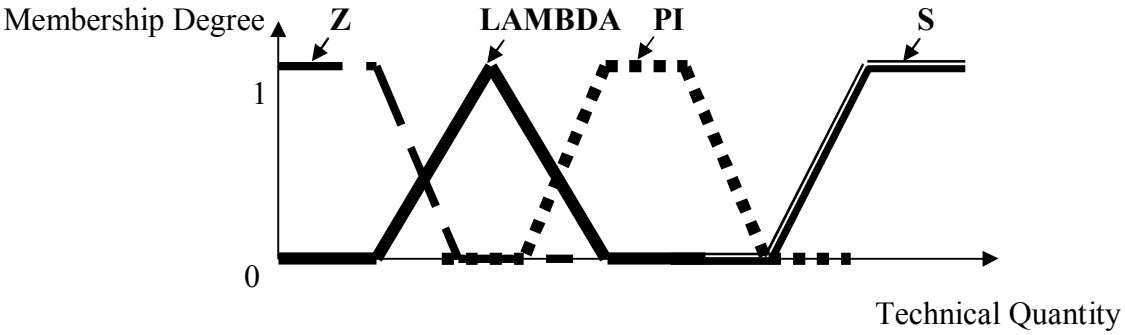


Figure 2.11: Standard Membership Functions

All of these Membership Functions can be mathematically represented as piecewise linear functions. The membership degree of a Base Variable value as given by its piecewise linear Membership Function is computed using the following algorithm:

i) The Membership Function is defined as a set of points ($P(x_i, \mu_i)$ with $i = 1, \ldots, n$, $x_1 \leq x_i \leq x_n$ and $\mu_i \in [0,1]$. $[x_1, \ldots, x_n]$ is the Base Variable range. The current operational value is $X \in [x_1, \ldots, x_n]$

ii) Evaluate the valid Base Variable interval with $x_k \leq X \leq x_{k+1}$

iii) Compute membership degree by:

$$\mu = \mu_i + (X - x_i) * (\mu_{i+1} - \mu_i)/(x_{i+1} - x_i) \qquad 2.27$$

where μ = Membership Degree, $\mu(x)$ = Membership Function, X = Universe of Discourse and x = Element of X [10].

2.2.4.2 Fuzzy Inference

The fuzzy inference mechanism is the Fuzzy Logic reasoning process that determines the output corresponding to fuzzified inputs and can be expressed thus:

$$R^i : If\ x_1\ is\ A_1^i\ AND\ x_2\ is\ A_2^i\ AND\ \ldots\ AND\ x_m\ is\ A_m^i\ THEN\ y\ is\ B^i \qquad 2.28$$

Where R^i is the i^{th} rule and $x_{1,\ldots,}\ x_m$ and y are respectively the input and output system variables. A_j^i are the antecedent linguistic terms (or Fuzzy Sets) in rule i with $1 \leq j \leq m$ being the number of antecedent variables and B^i is the rule conclusion being, for that type of rule, a real number usually a fuzzy singleton.

Each IF - THEN rule defines a fuzzy implication between condition and conclusion rule parts and is denoted by: A^i_1 x … x $A^i_n \rightarrow B^i$ [44]. The IF part can consist of more than one precondition linked together by linguistic conjunctions like AND and OR [10].

Fuzzy inference is a calculus consisting of three steps:

i) **AGGREGATION** (Computation of the IF part of the rules) **:** This step computes the support of the rule relative to the conditions and is done using the following algorithm:

- Aggregation of i terms (i condition parts) of linguistic variables
- Use Minimum operator for AND aggregation:

$$(\mu_{IF} = \text{Min}(\mu_i) \qquad 2.29$$

- Use Maximum operator for OR aggregation [10][67]:

$$(\mu_{IF} = \text{Max}(\mu_i) \qquad 2.30$$

ii) **COMPOSITION** (Computation of the THEN part of the rules): This step computes the degree of truth for the rule. It uses the support of the precondition to calculate support of the consequence. Standard methods include MAX/MIN (MAX/PROD) and Fuzzy Associative Map (FAM) [10].

- **MAX/MIN:** The consequence of a rule is considered equally as true as the condition and the rule base optimization often consists of arbitrary rule addition and deletion. The individual importance of a rule can be expressed only as a 0 or 1 [10].

- **FAM:** This is an advanced inference method, where each rule is assigned a Degree of Support (DoS) representing the individual importance of the rule. Rules themselves can be 'fuzzy' - with support between 0 and 1 [10]

The support of a consequence is calculated by linking the support of the entire condition with the Degree of Support (DoS) by a composition operator. Usually the product (PROD) operator is used as the composition operator because the Degree of Support (DoS) reflects rule significance [10].

The computation of the THEN part is done using the following algorithm:

- Combination of the IF part with the DoS (Degree of Support)
- Use the PROD operator to represent rule significance [10]:

$$\mu_{THEN} = \mu_{IF} \times DoS \qquad 2.31$$

iii) **RESULT AGGREGATION:** After the degree of truth for the rules are computed, this step determines which rules will contribute to the defuzzified result. If more than one rule produces the same consequence, an output must aggregate the results of these rules. A result aggregation step determines the maximum Degree of Support (DoS) for each consequence, which is used for all further processing. The MAX or BSUM operator can be used for result aggregation [10][68].

Result aggregation for rules having same consequence is done using the following algorithm:

- Aggregation of the results of the rules
- Use the MAX or BSUM operator [10]:

$$\text{MAX: } \mu_{RESULT} = \text{Max}_i\ (\mu_{THEN,\ RULEi}) \qquad 2.32$$

$$\text{BSUM: } \mu_{RESULT} = \text{Min}(1, \sum (\mu_{THEN,\ RULEi})) \qquad 2.33$$

2.2.4.3 Defuzzification

This maps output Fuzzy Sets defined over an output Universe of Discourse to crisp outputs with the aim of producing the non-fuzzy output that best represents the distribution of an inferred fuzzy output [71]. It is a process to extract an easily comprehensible answer from the solution set [68].

The Membership Functions are used to retranslate the fuzzy output into a crisp value. First a typical value is computed for each term in the linguistic variable and finally a best compromise is determined by balancing out the results using such methods as Centre of Maximum (CoM), Centre of Area (CoA) (or Centre of Gravity (CoG)), Mean of Maximum (MoM) etc [12].

The Centre of Maximum (Figure 2.12) and Centre of Area (Figure 2.13) defuzzification methods result in the "best compromise solution" while the Mean of Maximum results in the "most plausible solution". In control applications, the Centre of Maximum method is now more commonly used because the output value represents the best compromise of all inferred results [10]. Though the Centre of Area method used to be the most frequently

used, most of its implementations are only approximations since they neglect overlapping and can therefore be represented with the Centre of Maximum method [10].

For the Centre of Maximum method, the gray arrows indicate the locations of the individual term membership maxima and the inference result is shown by the height of the darker bar in the arrows.

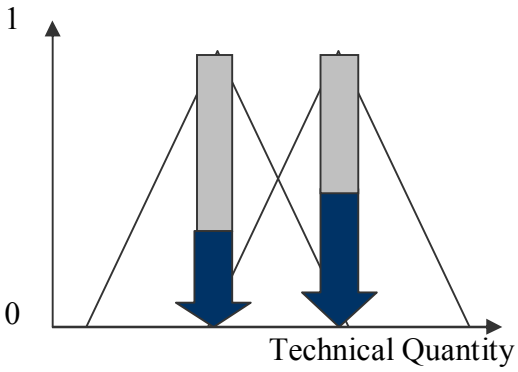


Figure 2.12 Centre of Maximum Method

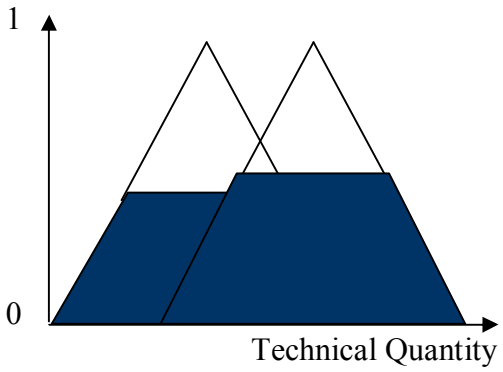


Figure 2.13: Centre of Area Method

Centre of Maximum method computes a crisp output as a weighted average of the term membership maxima weighted by the inference results. The algorithm is as follows:

- Calculate the crisp output Y
- Use the Centre of Maximum method
- Compromise between the aggregated results of the different terms j of the linguistic variable
- Based on the maximum Y_j of each term j [10]:

$$Y = \frac{\sum j\left(\mu_{RESULT,TERMi}\ x\ Y_j\right)}{\sum j\ \mu_{RESULT,TREM,J}} \qquad 2.34$$

2.3 NEURO-FUZZY SYSTEMS

Neuro-Fuzzy is a technology that provides flexible information process capability by devising methodologies and algorithms on a massively parallel system for representation and recognition of real life ambiguous situations [3]. As previously mentioned, Neuro-Fuzzy refers to a combination of Neural Nets and Fuzzy Logic Systems. This integration leads to a symbiotic relationship in which the Fuzzy Logic Systems provide a powerful framework for explicit knowledge representation, while the Neural Nets provide learning capabilities and exceptional suitability for computationally efficient hardware implementation [3][72]. The NeuroFuzzy System, therefore, enjoys the following advantages of both Fuzzy Logic Systems and Neural Nets:

i) It can handle numerical, linguistic, or logical kind of information.

ii) It can manage imprecise, partial, vague or imperfect information.

iii) It resolves conflicts by collaboration and aggregation.

iv) It has self -learning, self-organizing and self-tuning capabilities.

v) It has no need of prior knowledge of relationships of data.

vi) It can mimic human decision-making process.

vii) It has fast computation using fuzzy number operations [72].

2.3.1 Neural Nets And Fuzzy Logic System Integration

Neural Nets were derived from neuroscience and in particular from neurons, the fundamental units of the human brain. The Neural Net is essentially a 'Black Box' as it is difficult to really know what happens in the Net because it reveals little about its predictions. As such, what really matters is the external behaviour of the Net [38]. Neural

Nets can, therefore, be considered as being based on the actual means of processing information. Fuzzy Logic on the other hand is based on the way information is represented and is meant to represent human reasoning about information. It is this inherent difference that makes the combination of Neural Net and Fuzzy Logic System, NeuroFuzzy, better at representing complex systems [57] by enhancing some of its characteristics like flexibility, speed and adaptability [3].

Neuro-Fuzzy Systems combine the advantages of Fuzzy Logic Systems, which deal with explicit knowledge that can be explained and understood, and Neural Nets, which deal with implicit knowledge, which can be acquired by learning [42][65][10]. The learning capability is seen as an advantage from the point of view of Fuzzy Logic Systems while the formation of linguistic rule base will be an advantage from the point of view of Neural Nets. The main characteristic differences between Neural Nets and Fuzzy Logic Systems are shown in Table 2.2 [73][10].

Table 2.2: Characteristic Differences Between Neural Nets And Fuzzy Logic Systems

S/N	NEURAL NETS	FUZZY LOGIC SYSTEMS
1	Prior rule-based knowledge cannot be used	Prior rule-based knowledge can be used
2	Trains itself by learning from data sets	Cannot learn as it uses linguistic knowledge
3	Complicated learning algorithm	Simple interpretation and implementation
4	Difficult to extract knowledge	Knowledge must be available

The Neuro-Fuzzy System is designed to realize the process of fuzzy reasoning, where the connection weights of the network correspond to the parameters of fuzzy reasoning [3]. Neural Nets are used to tune the Membership Functions of the Fuzzy Logic Systems and

to extract fuzzy rules from numerical data [35][74]. Extraction of rules from Neural Nets enables humans to understand the Net's prediction process in a better manner. This is because rules can easily be verified [3].

In essence, the objective of Neuro-Fuzzy technology is to combine these techniques in a coherent manner so that their different strengths can complement each other [48]. Fuzzy Logic Systems are appropriate if sufficient expert knowledge about the process is available, while Neural Nets are useful if sufficient process data are available or measurable. Neural Nets can generate and refine fuzzy rules from training data. Both approaches build non-linear systems based on bounded continuous variables, the difference being that Neural Nets are treated in a numeric quantitative manner, whereas Fuzzy Logic Systems are treated in symbolic qualitative and/or numeric quantitative manner [3]. There are essentially three (3) different approaches for combining Neural Nets and Fuzzy Logic Systems:

i) **CONCURRENT NEURAL-FUZZY MODEL:** In this model, the Fuzzy Logic System and Neural Net are used concurrently for the same task. The Fuzzy Logic System is used either before or after the processing of the Neural Net. They are not related tightly and the Neural Net does not change any parameters in the Fuzzy Logic System [75]. This type of model is shown in Figure 2.14.

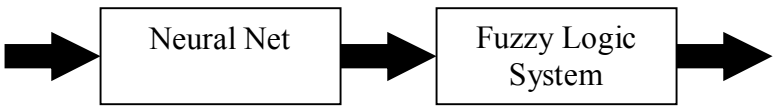


Figure 2.14: Concurrent Neuro-Fuzzy Model

ii) **CO-OPERATIVE NEURO-FUZZY MODEL:** In this model, a Neural Net or just a simple neural learning algorithm is used to learn certain parameters of Fuzzy Sets, fuzzy rules or weights of the Fuzzy Logic System. After learning, the Neural Net no longer exists, as the result is a pure Fuzzy Logic System [75][76]. This type of model is shown in Figure 2.15.

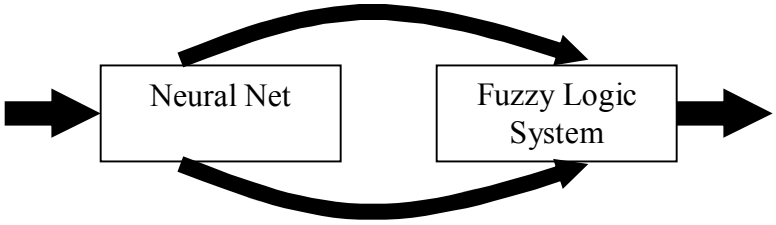


Figure 2.15: Concurrent Neuro-Fuzzy Model

iii) **HYBRID NEURO-FUZZY MODEL:** This model, which is newer than the others, can be interpreted as a Neural Net or as a Fuzzy Logic System. The Neural Net and Fuzzy Logic System are no longer separated. Modern Neuro-Fuzzy approaches use this model [35].

2:3.2 Neuro-Fuzzy System Structures

The NeuroFuzzy System structure is based on either the Mamdani structure or Takagi-Sugeno (TS) structure. The Mamdani NeuroFuzzy System has a five (5)-layer structure while the Takagi-Sugeno (TS) NeuroFuzzy System has a six (6)-layer structure.

2:3.2.1 Mamdani Neuro-Fuzzy System

The Mamdani NeuroFuzzy System has a five (5) layer structure as shown in Figure 2.16.

i) **LAYER-1 (INPUT LAYER):** It consists of the input nodes and directly transmits the input linguistic variable to the next layer. No computation is done at

this layer and the link weight is unity (1). The input linguistic variables are Relative Humidity (RH), Wind Direction (WD) and Wind Speed (WS).

ii) **LAYER-2 (FUZZIFICATION LAYER):** It is the input term layer where each node acts as a Membership Function and represents the terms of the respective linguistic variables. The output link represents the membership value, which specifies the degree to which an input value belongs to a Fuzzy Set. It is at this layer that the initial number and type of Membership Functions to be allocated to each of the input variables is decided. The final shapes of the Membership Functions will be fine-tuned during the network learning. The network will be tested with three (Low, Medium and High), four (Low, Medium_Low, Medium_High and High) and five (Very_Low, Low, Medium, High and Very_High) linguistic terms for both input and output linguistic variables.

iii) **LAYER-3 (RULE ANTECEDENT LAYER):** This layer is a fuzzy rule layer where each node represents one fuzzy rule. The links in this layer are used to perform antecedent matching of fuzzy rules. Normally a T-norm operator is used in this node. The output of each node represents the firing strength of the corresponding fuzzy rule.

iv) **LAYER-**4 **(RULE CONSEQUENT LAYER):** This node basically has two (2) tasks: To combine the incoming rule antecedents and determine the degree to which they belong to the output linguistic term. The number of nodes in this layer will be equal to the number of rules.

v) **LAYER-5 (COMBINATION AND DEFUZZIFICATION LAYER):** This node does the combination of all the rules consequents using a T-conorm (or S-

norm) operator and finally computes the crisp output after defuzzification [65][76].

The Mamdani System uses a supervised learning technique (e.g. Error Back Propagation) to learn the Membership Functions [73]

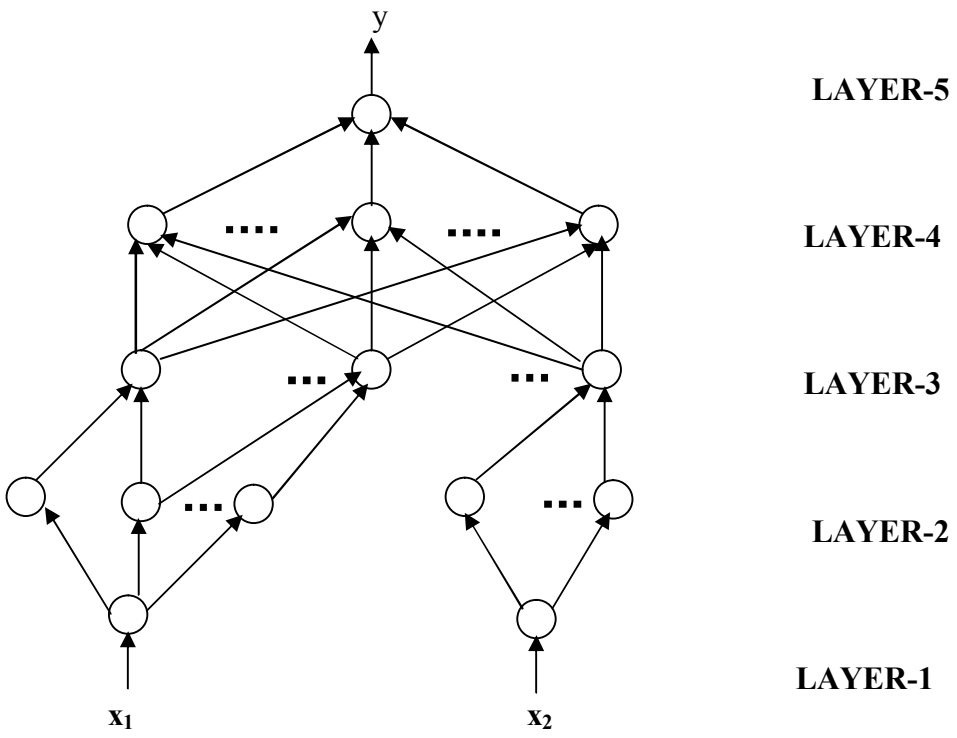


Figure 2.16: Mamdani Neuro-Fuzzy System

2:3.2.2 Takagi-Sugeno (TS) Neuro-Fuzzy System

The Takagi-Sugeno (TS) Neuro-Fuzzy System has a six (6) layer structure as shown in Figure 2.17.

i) **Layers-1, -2, and -3** function the same way as in the Mamdani Neuro-Fuzzy System [76].

ii) **LAYER-4 (RULE STRENGTH NORMALIZATION):** Every node in this layer calculates the ratio of the i^{th} rule's firing strength to the sum of all rules firing strength.

$$\overline{w}_i = \frac{w_i}{w_1 + w_2}, \text{ i} = 1, 2, \ldots \qquad 2.35$$

iii) **LAYER-5 (RULE CONSEQUENT LAYER):** Every node i in this layer is with a node function given by:

$$\overline{w}_i f_i = \overline{w}\left(p_i x_1 + q_i x_2 + r_i\right) \qquad 2.36$$

Where $\overline{\omega}_i$ = output of Layer-4 and $\{p_i, q_i, r_i\}$ = parameter set.

iv) **LAYER-6 (RULE INFERENCE LAYER):** The single node in this layer computes the overall output as the summation of all incoming signals [73][69]. The overall output is expressed as follows:

$$\sum_i \overline{w}_i f_i = \frac{\sum i w_i f_i}{\sum i w_i} \qquad 2.37$$

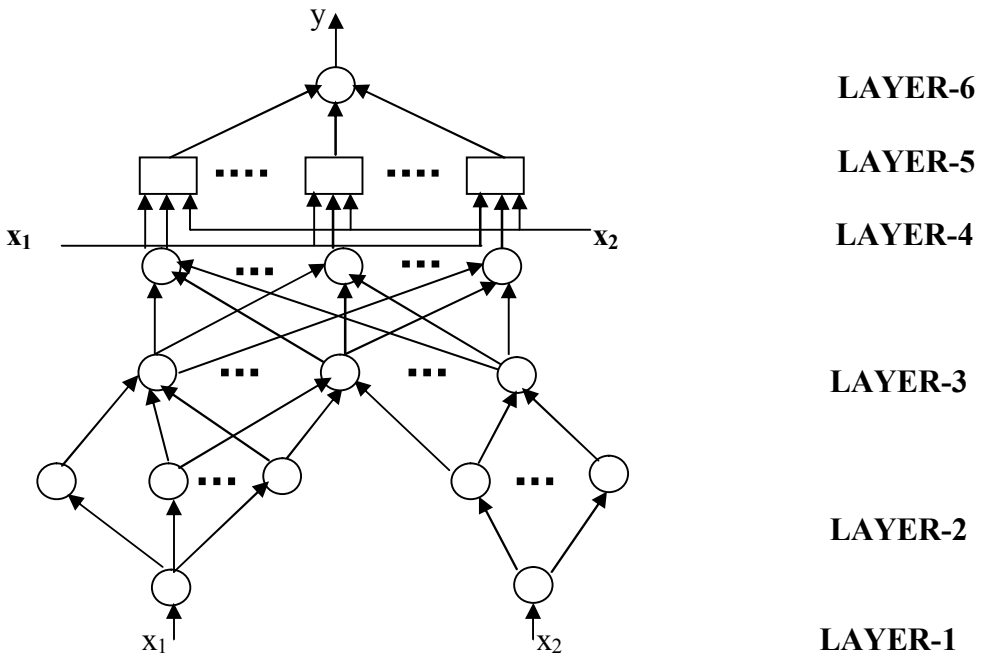


Figure 2.17: Takagi-Sugeno (TS) Neuro-Fuzzy System

The Takagi-Sugeno (TS) Neuro-Fuzzy System uses a mixture of Error Back Propagation to learn the Membership Functions and Least Mean Square estimation to determine the coefficients of the linear combinations in the rule's conclusions [76]

2:3.3 Neuro-Fuzzy Model: The Modified n*fMod* Model

Neuro-Fuzzy Systems are able to learn from examples, to generalize from learned knowledge, to explain decisions they make and to synthesize the knowledge -in the form of fuzzy rules- from the data. The Neuro-Fuzzy model to be used in this work is a modification of the *nfMod* model [19]. The modification is as a result of the fact that the training will be carried out using the fuzzyTECH Neuro-Fuzzy module.

The ***nfMod*** model implements a collection of R fuzzy rules using the Mamdani-system type form:

$$If\ (x_1\ is\ A_{1r})\ AND\ (x_2\ is\ A_{2r})\ AND...AND\ (x_n\ is\ A_{nr})\ THEN\ (y\ is\ B_r) \qquad 2.38$$

where r = 1, 2, …, R and $A_{ir} \in F(x_i, B_r \in F(y)$

The structure of the Neuro-Fuzzy System adopted for this work is shown in Figure 2.18. During the learning phase, the parameters of the input and output Fuzzy Sets are tuned in order to minimize the cost function, which is a Mean Square Error between the expected outputs y_j, of particular output learning sample and estimated outputs o_j generated by the system (j =1, 2, …b, where b = number of Fuzzy Sets for output) [19]. A modified Error Back Propagation and competitive learning (winner neurons) algorithms can be used for learning of the Neuro-Fuzzy System.

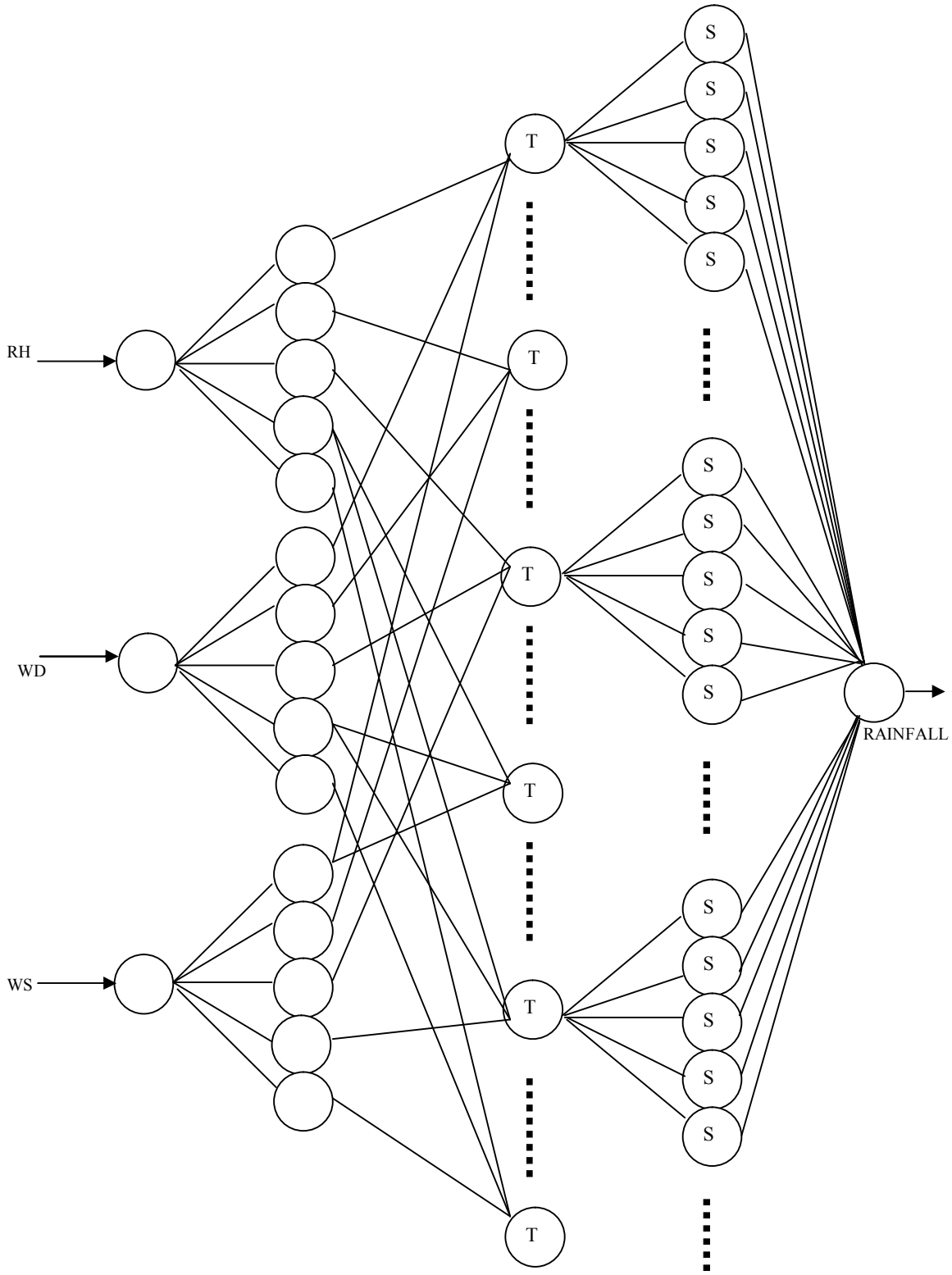


Figure 2:18: Modified ***nfMod*** Neuro-Fuzzy System Structure

The aggregation of fuzzy rule premises is realized by T-norm (symbol T) and the aggregation of the rules with the same consequents is performed with the use of the S-norm (symbol S). T-norm is implemented as MIN operator and S-norm as MAX operator [19].

After completion of the learning phase, a pure Fuzzy Logic System is generated and the system will be in a working phase. A non-fuzzy, numerical response of the system can also be obtained as the system is in the working phase.

2:3.3.1 Triangular-Norms (T-norm) And Conorms (T-conorm or S-norm)

i) **Triangular-norm:** The intersection of two Fuzzy Sets A and B is specified in general by a function: $T:[0,1]\cap[0,1]\to[0,1]$, which aggregates two membership grades as follows:

$$\mu_{A\cap B}(x)=T(\mu_A(x),\mu_B(x)) \qquad 2.39$$

A mapping $T:[0,1]\cap[0,1]\to[0,1]$ is a T-norm and it is extensively used to model the logical connective AND. The basic T-norm is the Minimum operator:

$$(\min(a,b)=\min\{a,b\}) \qquad 2.40$$

The basic T-norm must satisfy the following basic requirements [77][78]:

- **Symmetricity**, also called commutativity, implies that the operator is indifferent to the order of the Fuzzy Sets to be combined. It is defined as:

$$T(a,b)=T(b,a),\forall a,b\in[0,1] \qquad 2.41$$

- **Associativity** implies the operator can take the intersection of any number of sets in any order of pair-wise groupings. It is defined as:

$$T(a,T(b,c)) = T(T(a,b),c), \forall a,b,c \in [0,1] \qquad 2.42$$

- **Monotonicity** implies a decrease in the membership values in A or B cannot produce an increase in the membership value in $A \cap B$. It is defined as:

$$T(a,b) \le T(a',b') \; if \; a \le a' \; and \; b \le b' \qquad 2.43$$

- **Boundary** implies the operator imposes the correct generalization to crisp sets. It is defined as:

$$T(a,1) = a, \forall a \in [0,1] \qquad 2.44$$

ii) **Triangular-conorm (T-conorm or S-norm):** The union of two Fuzzy Sets is specified in general by a function: $S : [0,1] \cup [0,1] \to [0,1]$, which aggregates two membership grades as follows:

$$\mu_{A \cup B}(x) = S(\mu_A(x), \mu_B(x)) \qquad 2.45$$

A mapping $S : [0,1] \cup [0,1] \to [0,1]$ is a triangular conorm (S-norm) and they are used extensively to model the logical connective OR. The basic S-norm is the Maximum operator:

$$(\max(a,b) = \max\{a,b\}) \qquad 2.46$$

The basic S-norm must satisfy the following basic requirements [77][78]. The justification of these basic requirements is similar to that of the requirements of the T-norm operator.

- **Symmetricity** is defined as:

$$S(a,b) = S(b,a), \forall a,b \in [0,1] \qquad 2.47$$

- **Associativity** is defined as:

$$S(a,S(b,c)) = S(S(a,b),c), \forall a,b,c \in [0,1] \quad 2.48$$

- **Monotonicity** is defined as:

$$S(a,b) \leq S(a',b') \; if \; a \leq a' \; and \; b \leq b' \quad 2.49$$

- **Boundary** is defined as:

$$S(a,0) = a, \forall a \in [0,1] \quad 2.50$$

If T is a T-norm then the following equality defines an S-norm [77]:

$$S(a,b) = 1 - T(1-a, 1-b) \quad 2.51$$

2:3.4 Learning Algorithms

Neuro-Fuzzy approaches to anticipation aim at amalgamating the predictive and learning capabilities of neural computing with the clarity and coherence of articulation found in fuzzy algorithm [27]. A common way of applying Neural Net learning algorithm to a Fuzzy Set is to represent it in a special Neural Net - like structure [76]. There are several learning rules available and the common ones include the following: Hebbian, Winner Takes All (WTA), Delta and Error Back Propagation (EBP).

2:3.4.1 Hebbian Learning Rule

This is regarded as the 'father' of all learning rules. The Hebb (1949) learning rule is based on the assumption that if two (2) neighbour neurons must be activated and deactivated at the same time, then the weight connecting these neurons should increase. For neurons operating in the opposite phase, the weight between them should decrease. If there is no correlation, the weight should remain unchanged. This assumption can be described by:

$$\Delta w_{ij} = c x_i o_j \qquad 2.52$$

w_{ij} = weight from i^{th} to j^{th} neuron; c = learning constant; x_i= signal on i^{th} input; o_j = output signal [1].

The training process usually starts with values of all weights set to zero. Since desired responses of the neurons are not used in the learning procedure, the Hebbian learning rule is an unsupervised learning rule [1].

2:3.4.2 Winner Takes All (WTA) Learning Rule

The Winner Takes All (WTA) rule was developed in 1982 by Kohonen and is used for automatic clustering, and for extracting statistical properties of input data. The weights are only modified for the neuron with the highest ***net*** value. Weights of remaining neurons are left unchanged. This is also an unsupervised learning rule (as the desired output is not known). The ***net*** values for all neurons in the network are compared in each training step [1]. The ***net*** is defined as a scalar product of the weight and input vectors, mathematically expressed as:

$$net = \sum_{i=1}^{n} w_i x_i \qquad 2.53$$

2:3.4.3 Delta Rule And Error Back Propagation (EBP)

The Error Back Propagation, first proposed by Rumelhart et al (1986), was derived because the Delta rule, first proposed by Widrow and Hoff (1962) [54][55], can only be applied to networks with only a single neuron layer [10]. This means that the Error Back Propagation is essentially a Delta rule generalized for multi-layer networks [1] and non-

linear differential transfer functions [79]. It is the most often used supervised learning algorithm of such networks.

The principle behind the Error Back Propagation training algorithm is that it finds the derivatives of an error function with respect to the weights in the network [56]. The basic idea is to compare the result computed by the current network with the target output value from the sample pattern. This means that the Error Back Propagation is a gradient descent technique used to minimize the error E, for a particular training pattern. The gradient $\nabla E\left(\frac{\partial E}{\partial w_{ij}}\right)$ gives the direction of the error. For training, deviation between these values is used to update the net [7]. It is computationally efficient as it avoids quantities that are already computed. This is important as both the forward and backward pass use the same connections [80].

The Error Back Propagation can be applied to any feedforward network. Since the Fuzzy Logic System is basically a feedforward network, the Error Back Propagation can be used to train it. The idea is to use the training algorithm to adjust weights, centres and widths of the Membership Functions [56]. However, the conventional Error Back Propagation method cannot be applied directly because the functions in the inference process of the Fuzzy Logic System are usually non-differentiable [73].

2:3.5 Neuro-Fuzzy Learning Algorithm

A Neuro-Fuzzy System should be able to learn linguistic rules and/or Membership Functions or optimize existing ones. There are essentially three (3) possibilities:

i) The system starts without rules and creates new rules until the learning problem is solved. Creation of a new rule is triggered by a training pattern that is not sufficiently covered by the current database.

ii) The system starts with all the rules that can be created due to the partitioning of the variables and deletes insufficient rules from the rule base based on an evaluation of their performance.

iii) The system starts with a rule base with a fixed number of rules. During learning, rules are replaced by an optimization process [3].

There are two (2) types of learning:

i) **SUPERVISED LEARNING:** If a given input pattern, like a recognizable character, must be associated with a specified output pattern, like a set of all valid characters, each training step can be supervised by comparing the computed and the desired results [10]. This can be achieved by adjusting the network weights in an iterative manner in order to minimize their differences [9].

ii) **UNSUPERVISED LEARNING:** If the task of the training procedure is to discover regularities within the environment, like categorizing properties of given input patterns, there are usually no specified output patterns or structures to supervise the training success. The weights of the network are adjusted so that similar inputs produce similar outputs. This is the unsupervised learning [10] [9].

Using an Error Back Propagation type learning algorithm, the Neuro-Fuzzy System can identify fuzzy rules and learn Membership Functions of the fuzzy reasoning [3]. The conventional Error Back Propagation cannot be used to train the Fuzzy Logic System directly because:

i) The Fuzzy Logic System uses different mechanisms for computation:
   - Input Layer uses Fuzzification
   - Hidden Layers use Fuzzy Inference
   - Output Layer uses Defuzzification.

ii) The Error Back Propagation differentiates the transfer functions of the neurons to determine the influence of each neuron whereas the standard fuzzy inference cannot be differentiated [10].

The first problem can be solved using an expanded Fuzzy Logic inference method called Fuzzy Associative Map (FAM) (discussed in Section 2:2.4.2). The second problem is solved using a modified Error Back Propagation algorithm [10].

The modified Error Back Propagation algorithm uses a similar idea as the conventional Error Back Propagation method. First the system determines the error of the estimation, then propagates the error back, and adjusts the parameters to optimize the result. The changes of parameters are based on the rules of rewarding correct results (winner neurons) and punishing erroneous results [75].

By using an incremental/decremental learning rule base, rules can be learnt. It can add or remove rules whenever the system cannot handle the input data correctly. Whenever an

input pattern in the combination of Fuzzy Sets yields the highest degree of membership for the respective input feature, a rule node is created. A new rule node is created if this combination is not identical to the antecedents of an existing rule and the maximum permitted number of units of hidden layer is not reached [75].

The rule-learning algorithm is as follows:

i) For each input, find the Membership Function:

$$\mu_{ji} = \max\left\{\mu_j^{(i)}(x_i)\right\}, \qquad 2.54$$

where $\mu_j^{(i)}$ is the classification function.

ii) If there are still less than limited nodes in the hidden layer, and there is no rule node R with $w(x_1, R) = \mu_{j1}^{(1)}, \ldots, w(x_n, R) = \mu_{jn}^{(n)}$, then create this node and connect it.

iii) If there are still unprocessed patterns and the limited number of hidden layers is not reached, go to step (i) [75].

After the rules are established, the next step is to learn the Fuzzy Sets. The basic fuzzy function used is the triangular (Lambda) Membership Function. It can be represented as:

$$\mu : R \to [0,1]; \quad \mu_x = \begin{Bmatrix} 0 & if\ x \le a \\ \dfrac{x-a}{b-a} & if\ a \le x \le b \\ \dfrac{c-x}{c-b} & if\ b \le x \le c \\ 0 & if\ c \le x \end{Bmatrix} \qquad 2.55$$

The parameters (a, b, c) with a < b < c determine the x-coordinates of the three corners of the underlying triangular membership function (MBF) as shown in Figure 2.19 [78][75].

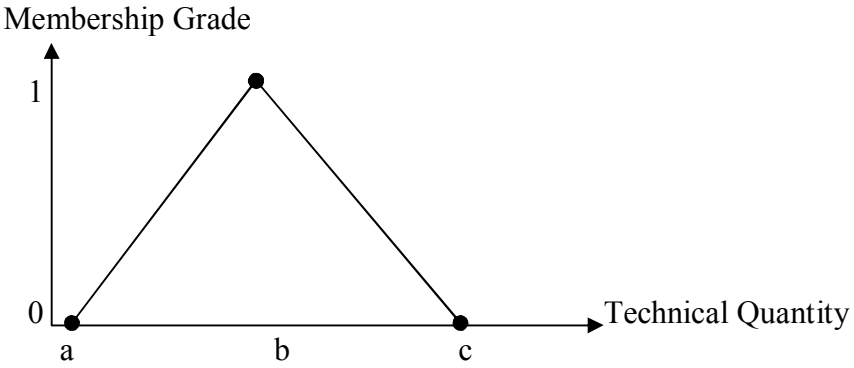


Figure 2.19: Triangular Membership Function

The Fuzzy Set learning algorithm is as follows:

i) For each output node c_i determine the output error:

$$\delta_{c_i} = t_i - o_{c_i} \qquad 2.56$$

ii) For each rule with $o_R > o$, determine the Delta value:

$$\delta_R = o_R(1 - o_R)\sum_{c \in U_3} w(R,c)\delta_c \qquad 2.57$$

iii) Find $x^{/}$ such that:

$$w(x^{/}, R)(o_{x^{/}}) = \min_{x \in U_1} \{w(x, R)(o_R)\} \qquad 2.58$$

iv) Using the learning rate $\sigma > 0$, determine the delta value for its parameters a, b, c for the Fuzzy Set w(x, R).

$$\begin{aligned} \delta_b &= \sigma\delta_R(c-a)sign(o_{x^{/}} - b) \\ \delta_a &= -\sigma\delta_R(c-a) + \delta_b \\ \delta_c &= \sigma\delta_R(c-a) + \delta_b \end{aligned} \qquad 2.59$$

v) If an epoch is completed and the END criterion is met, then STOP. Otherwise, go to step (i) [75].

NeuroFuzzy Systems use a two-step learning method to find the best match from the data. The system starts with no hidden layer and learns the rules first, and then uses these rules to train the network using the training set [75].

CHAPTER THREE

DEVELOPMENT OF FORECAST MODEL

3.1 DEVELOPMENT OF NEUROFUZZY-BASED FORECAST MODEL

The development of the NeuroFuzzy-based forecast model using the fuzzyTECH NeuroFuzzy Module involves the following steps:

i) **OBTAINING TRAINING DATA:** Neural Nets and NeuroFuzzy learning procedures depend on the number and sequence of sample data. However, as a system can be no better than the quality of the data presented, the quality is more important than the quantity of data. The weather data used in this work were obtained from the Nigeria Meteorological Agency (NiMet), Zaria and the Meteorological Unit of the Department of Soil Science, Ahmadu Bello University, Zaria for the period, 1993 – 2005. The training data (1993 to 2002) to be used by the NeuroFuzzy Module can be generated using any ASCII editor, Spreadsheet, Database or Data Acquisition software. The generated data must be of "**comma separated values**" (DHP_SC2.csv) file format type as in Figure 3.1. The file must not contain any empty rows and data record rows (starting from row 2) must start with the record names and values of input and output variables [10]. The complete data is shown in Appendix A1.

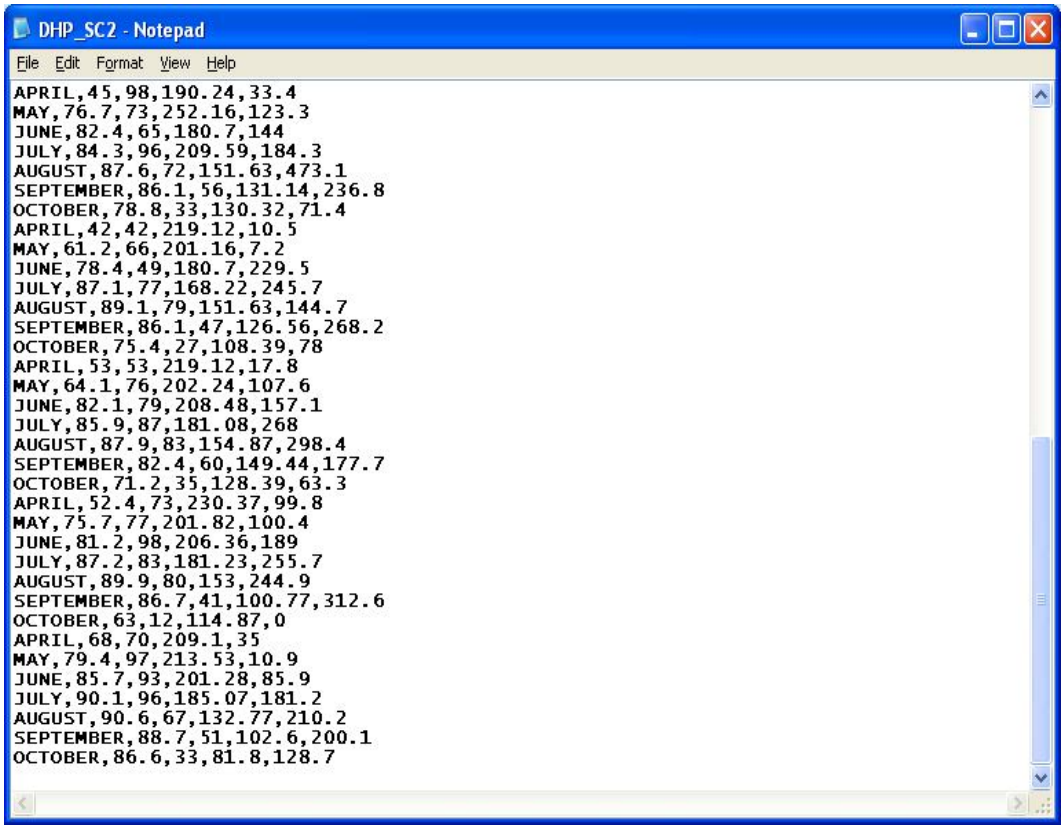


Figure 3.1: Generated Data File Using Notepad

ii) **CREATING AN 'EMPTY' FUZZY LOGIC SYSTEM:** The fuzzyTECH NeuroFuzzy Module supports the development of an 'empty' Fuzzy Logic System using the Fuzzy Design Wizard (FDW). The Fuzzy Design Wizard creates a COMPLETE but FALSE rule base with a Degree of Support (DoS) of 0 for all the rules as in Figure 3.2. Rules with Degree of Support (DoS) of 0 do not influence the computations of the Fuzzy Logic System but are required for NeuroFuzzy training. This is because only the set of rules which influence system behaviour within the given sample data are affected during the training [4].

IF			THEN	
REL_HUM	**WIND_DIR**	**WIND_SPD**	**DoS**	**RAINFALL**
Very_low	Very_low	very_low	0.00	very_low
Very_low	Very_low	very_low	0.00	Low
Very_low	Very_low	very_low	0.00	Medium
Very_low	Very_low	very_low	0.00	High
Very_low	Very_low	very_low	0.00	very_high
Very_low	Very_low	Low	0.00	very_low
Very_low	Very_low	Low	0.00	Low
Very_low	Very_low	Low	0.00	Medium
Very_low	Very_low	Low	0.00	High
Very_low	Very_low	Low	0.00	very_high
Very_low	Very_low	Medium	0.00	very_low
Very_low	Very_low	Medium	0.00	Low
Very_low	Very_low	Medium	0.00	Medium
Very_low	Very_low	Medium	0.00	High
Very_low	Very_low	Medium	0.00	very_high
Very_low	Very_low	High	0.00	very_low
Very_low	Very_low	High	0.00	Low
Very_low	Very_low	High	0.00	Medium

Figure 3.2: Sample Rule Base of Empty Fuzzy System (DoS of 0)

The Fuzzy Design Wizard reads the sample data file and extracts from it, the following:

- The number and names of variables
- The data range of each variable [10].

The Fuzzy Design Wizard developed the fuzzy training model based on the following assumptions:

- All except the last variable are input; the last variable is the output
- All inputs are implemented by five (5) linguistic terms
- The output is implemented by five (5) linguistic terms as in Figure 3.3.

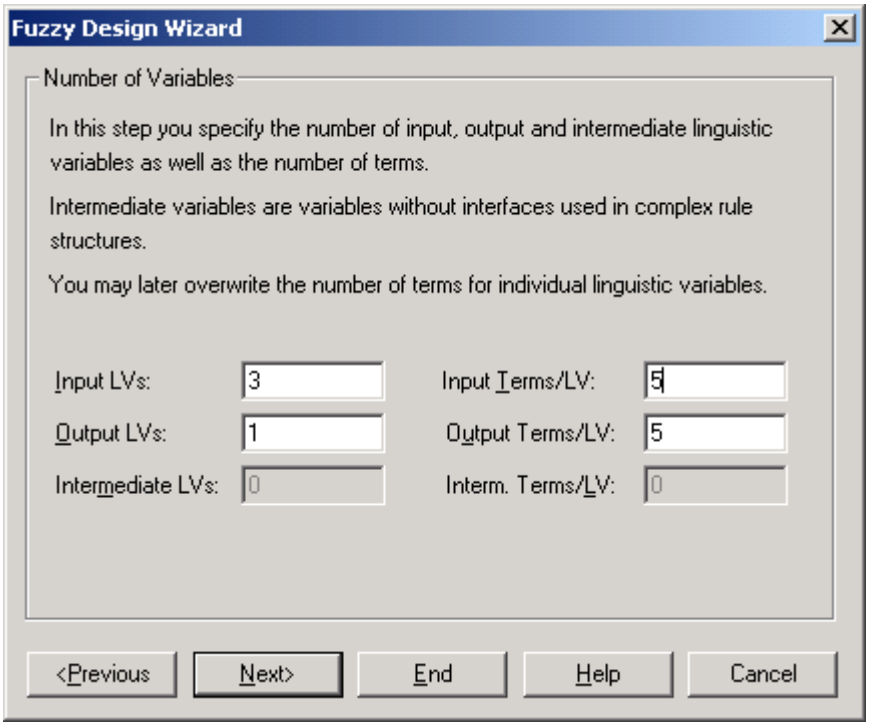


Figure 3.3: Fuzzy Design Wizard Dialog Box For Number of Variables

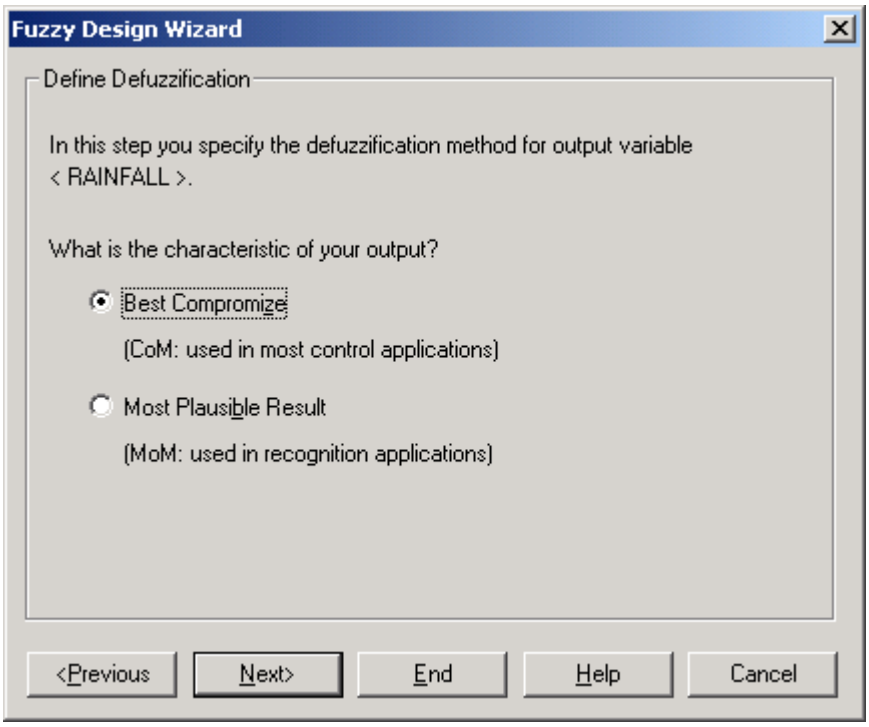


Figure 3.4 Fuzzy Design Wizard Dialog Box For Selecting Defuzzification Method

- Centre of Maximum (CoM) defuzzification method is used for the outputs as in Figure 3.4. By choosing a Centre of Maximum (CoM) defuzzification, the Fuzzy Design Wizard assumes the use of compromise defuzzification, which is compulsory for NeuroFuzzy training. This is because the NeuroFuzzy training uses error gradients to determine a direction of optimization. System response to small parameter changes is evaluated to determine the gradients and as non-compromising defuzzification methods do not react to small changes, only compromising methods can be used for training [10].

iii) **ENTERING EXISTING KNOWLEDGE IN THE SOLUTION:** The fuzzyTECH NeuroFuzzy Module uses 'a-priori' knowledge as a starting point for the training. All existing and expert knowledge can be entered into the system. The access to the 'empty' system allows a system configuration where training results can be explained to be found. The fact that existing knowledge can easily be used is a major advantage of the NeuroFuzzy approach over a Neural Net solution. The NeuroFuzzy Module extracts the necessary information from the sample data. Even if all information required in building the solution is already contained in the training data, entering existing knowledge can speed up the training. Also, if the training data is of poor quality, using existing knowledge to assist the NeuroFuzzy Module may be the only way to obtain a solution at all [10].

iv) **OPENING COMPONENTS OF THE FUZZY LOGIC SYSTEM FOR LEARNING:** All or specific components of the Fuzzy Logic System can be opened for learning. A component opened for learning using the [**LEARN MBF]**

button will have a "**L**" prefix as in Figure 3.5 and the Degree of Support (DoS) values opened for learning, if necessary, will have a gray background [10].

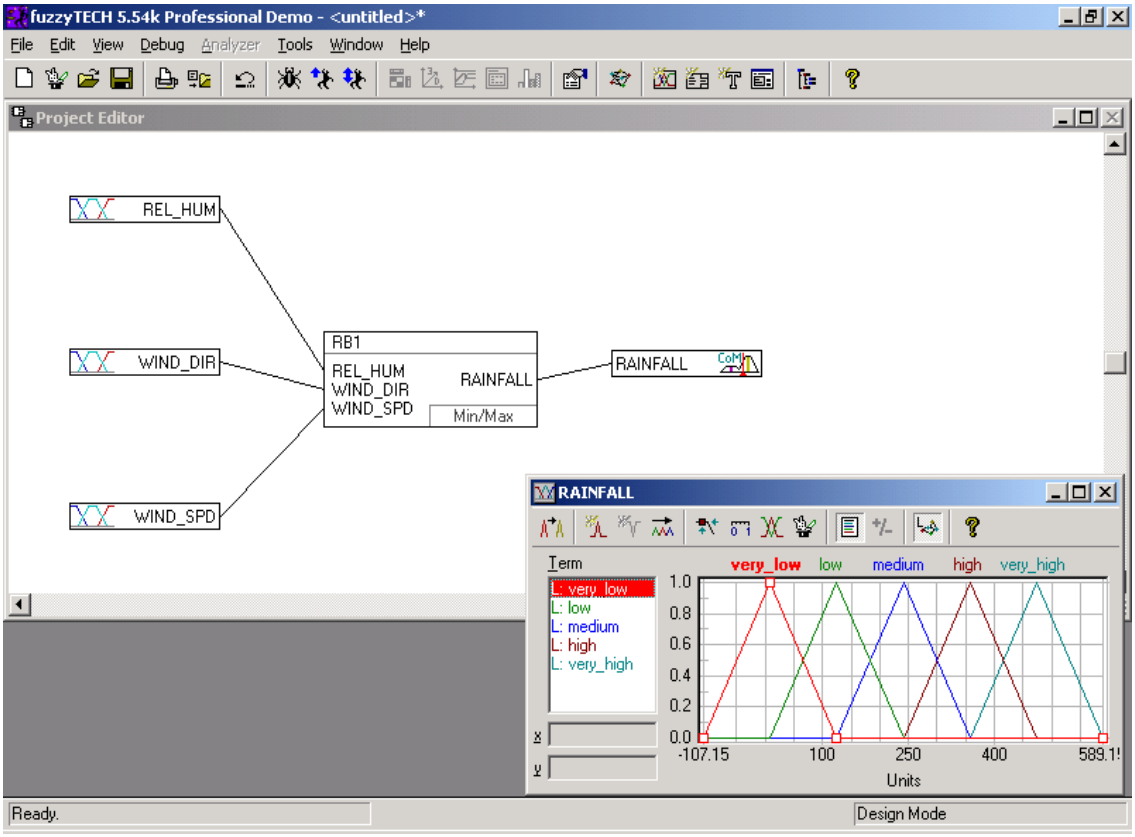


Figure 3.5:Rainfall Membership Functions Opened For Learning

v) **CONFIGURING THE NEUROFUZZY MODULE:** For NeuroFuzzy training, the Learning Method must be specified and configured by setting parameters for it. The NeuroFuzzy configuration Dialog is as shown in Figure 3.6. The fuzzyTECH NeuroFuzzy Module allows the modification of the following parameters:

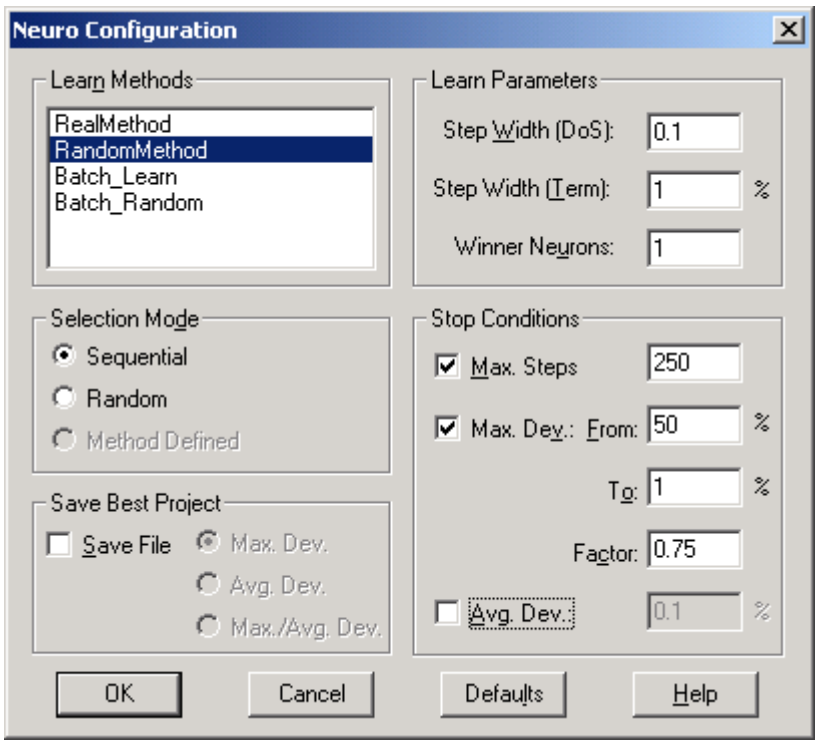


Figure 3.6: NeuroFuzzy Configuration Dialog Box

- **LEARNING RATE:** In order to converge on a solution for all sample cases, the training steps are repeated, and for each computation a small descent step (gradient descent) is performed. This step length can be adjusted with the Learning Rate. This is a global training parameter that determines the STEP WIDTH of each training step. The STEP WIDTH (DoS) determines the maximum amount (the value that is added or subtracted to a winner Degree of Support (DoS)) that a fuzzy rule is changed within a single training iteration. The value is set to **0.1**. The Learning Rate for Degree of Support (DoS) values is normalized as the values themselves in the interval **[0…1]**. The STEP WIDTH (Term) determines the changes implemented in the positions of terms. Because base variable ranges are not normalized, it is noted as a percentage of the complete base variable range [10]. The value is set to **1%**.

- **LEARNING METHOD:** The NeuroFuzzy Module allows standard methods (REAL_METHOD and RANDOM_METHOD) and batch training procedures (BATCH_LEARN and BATCH_RANDOM). The first of each group uses the Learning Rate entered as constant training steps, while the second uses randomly defined steps from the equally partitioned interval **[0…Learning Rate]**. The RANDOM_METHOD is to be used in this investigation as the Learning Method as it incorporates the Error Back Propagation and Competitive Learning methods with a randomly defined Learning Rate [10].
- **WINNER NEURONS:** This determines the basic training strategy. Only the winner neurons are updated in order to keep a rule base transparent. The number of winners determines how many neurons are updated within each iteration step. Iteration is completed each time all samples are used once for computation of an output. A winner neuron equal to one **(1)** means that in every training iteration only one rule – the winner in competition to all other rules- is changed. Using this configuration is known as Competitive Learning. The more the winner neurons specified, the less is the training results protected from former samples [10].
- **AVERAGE DEVIATION:** The Average Deviation criterion is fulfilled if the average of the errors occurring during a complete iteration is less than a defined error threshold. The average error is computed by adding the errors for each sample divided by the number of samples. The last computed error is taken into account for each sample. In order to enhance the efficiency of the system, the error is only computed if the sample is trained. Even when all samples except one are perfectly trained, the failed sample may cause problems that cannot be compensated. This

means that the error of a single sample is of more interest than the average error and as such, in most cases, the Average Deviation criterion is not used as a Stop Condition [10].

- **MAXIMUM DEVIATION:** The Maximum Deviation criterion compares the error of the worst sample with an error threshold. Samples under the threshold are skipped within the training procedure. Thus, only the samples still "bad enough" are used for further training of the system. Training results are automatically evaluated after any complete iteration. The more samples found that produce errors below the threshold, the faster the training computations are executed. Maximum Deviation criterion can only be used as a Dynamic Break Condition [10].
- **DYNAMIC BREAK CONDITION:** The Maximum Deviation criteria, as a dedicated Stop Condition, can implicitly be used to improve the training performance of the NeuroFuzzy Module optimization. When using Maximum Deviation as the STOP criterion, only the samples that continue to produce an error larger than the defined threshold are trained. The threshold starts at the **[FROM]** value and is reduced by the **[FACTOR]** value each time the threshold parameter is fulfilled. The training is stopped when the threshold has reached the entered **[TO]** value and all samples meet the designated criteria or the maximum number of iterations is reached, whichever comes first. The value in the **[FACTOR]** field is used to determine the next threshold when the current threshold is fulfilled by all samples**.** The factor is multiplied with the current threshold; thus, the factor chosen must be between 0 and 1. A factor of 1 fixes the threshold to the **[FROM]** value; a factor of 0 completes the training in two steps: one using the **[FROM]** threshold and the second using the **[TO]**

threshold. Values close to 1 result in training only the worst samples and shorter computation times per iteration [10]. As shown in Figure 3.6, the initial **[FROM]** value is **50**, the **[FACTOR]** value is **0.75** and the **[STOP]** value is **1.** Training runs of 200, 250, 300, 350, 370, 400, 500 and 1000 iterations were carried out and each model tested with the validation data of 2003. Based on the outcome of the tests, as in Figure's 3.7 – 3.14, the maximum number of iterations equal to **250** was decided upon. This is because the test plot obtained from using **250** iterations shows a better trend between actual and predicted rainfall when compared with the other number of iterations.

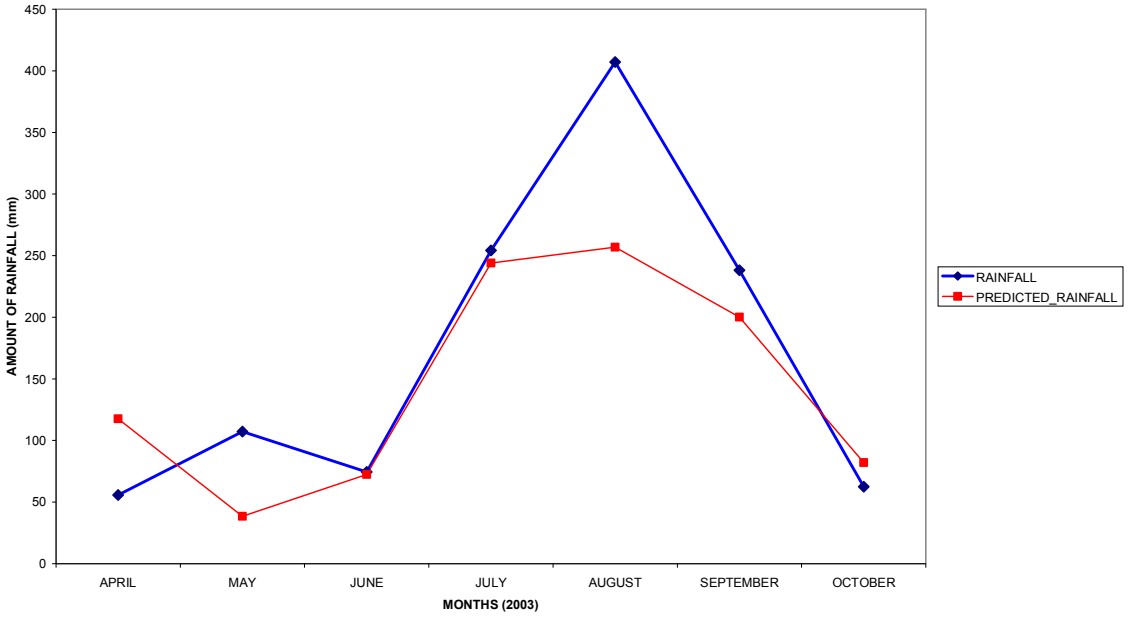


Figure 3.7: Test Plot for 200 Iterations

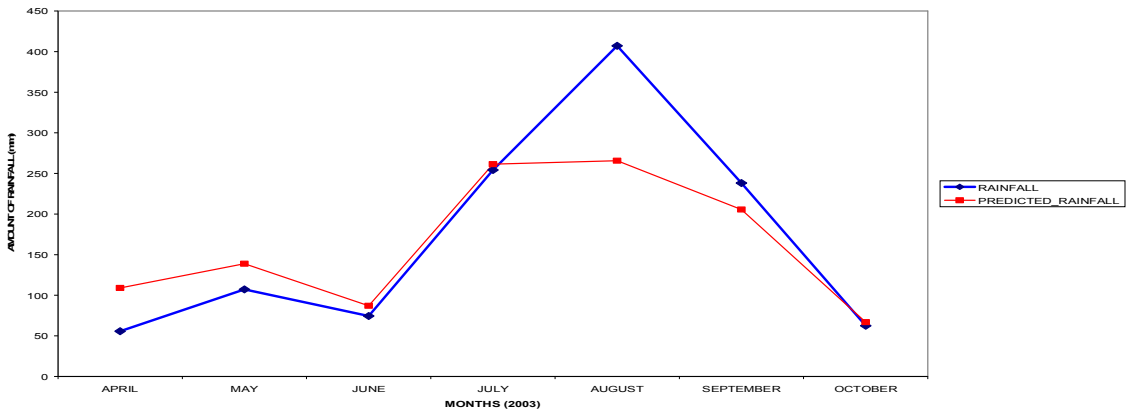


Figure 3.8: Test Plot for 250 Iterations

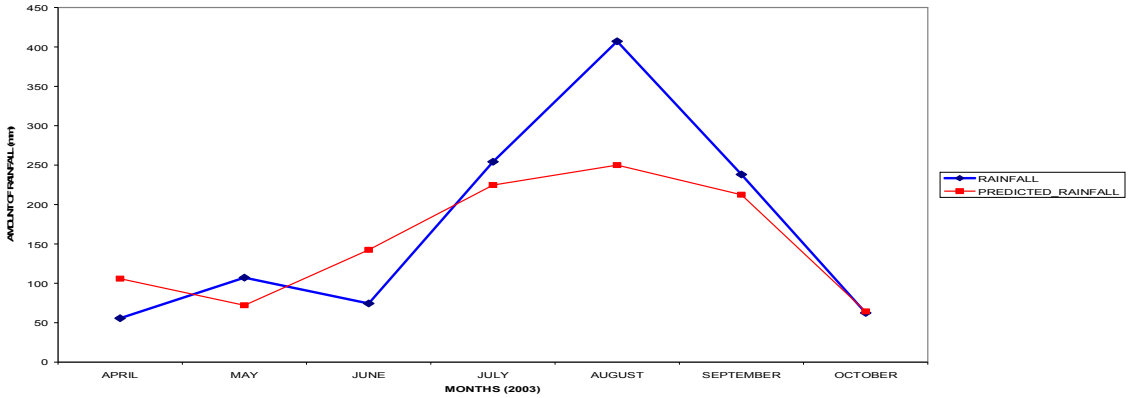


Figure 3.9: Test Plot for 300 Iterations

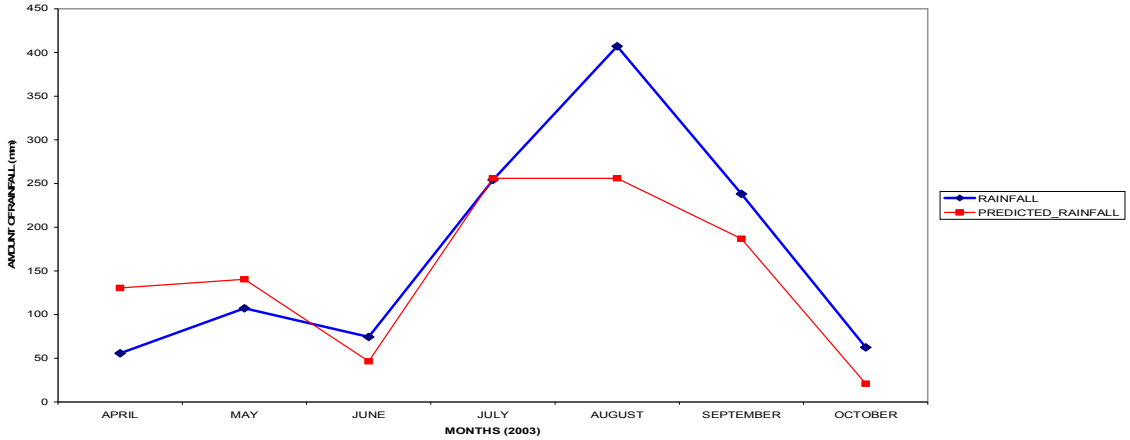


Figure 3.10: Test Plot for 350 Iterations

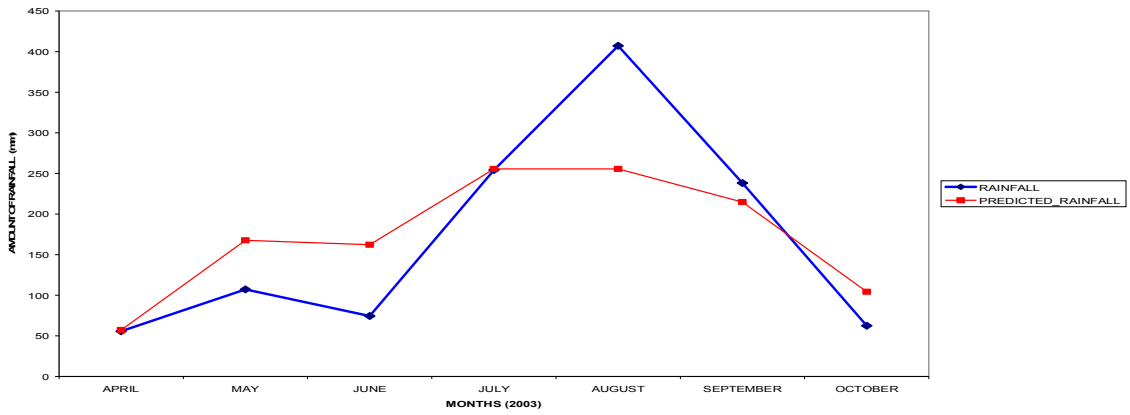


Figure 3.11: Test Plot for 370 Iterations

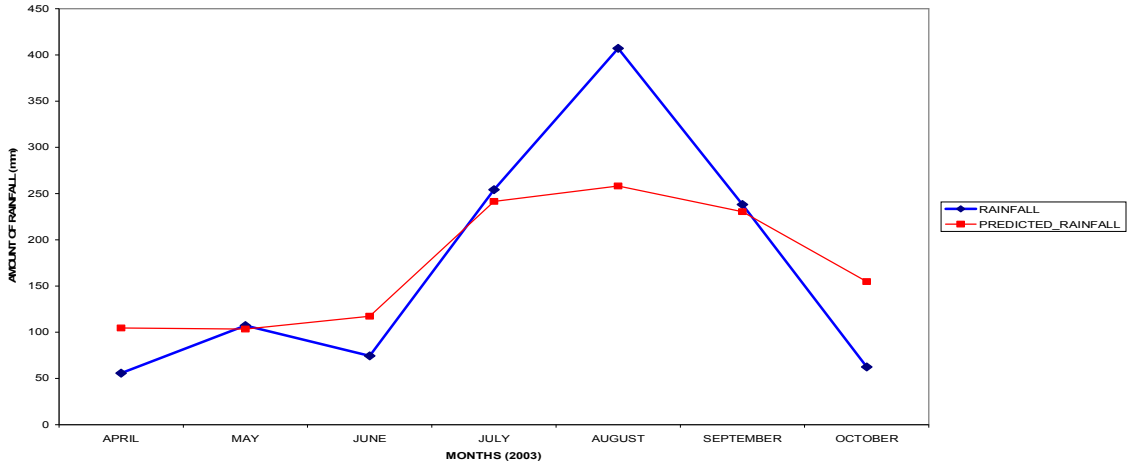


Figure 3.12: Test Plot for 400 Iterations

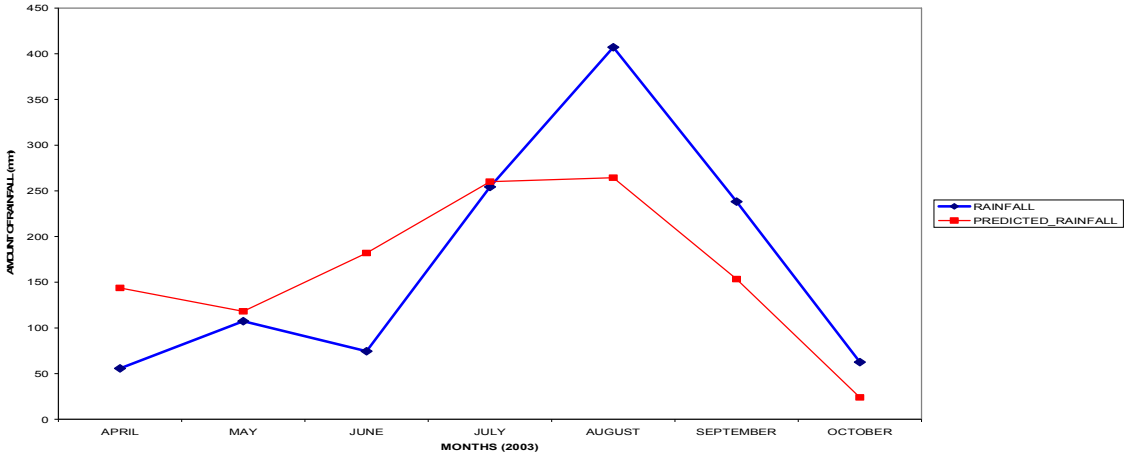


Figure 3.13: Test Plot for 500 Iterations

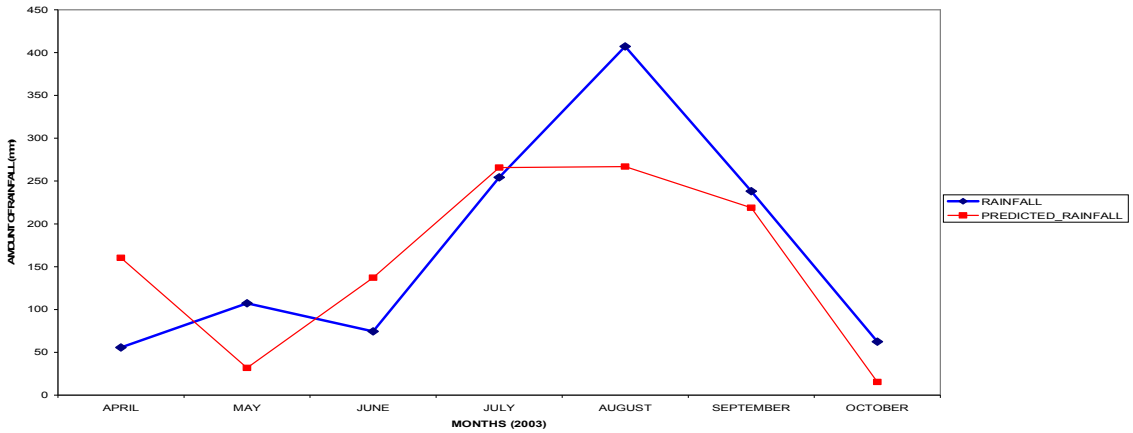


Figure 3.14: Test Plot for 1000 Iterations

- **SAMPLE SEQUENCING:** The sample data may be provided in Sequential or Random order. Random sequencing takes the next sample from the equally partitioned set of all samples while Sequential sequencing takes the next sample following the sequence in the sample data file [10]. The Sequential method is adopted for this work.

Before the NeuroFuzzy Module training is started, all the fuzzyTECH Editors and Analyzers are to be configured and arranged. By the entering the Interactive Debug mode, the Watch Window (Figure 3.15) and other analyzers (3D Plot, Time Plot, etc) are activated [10].

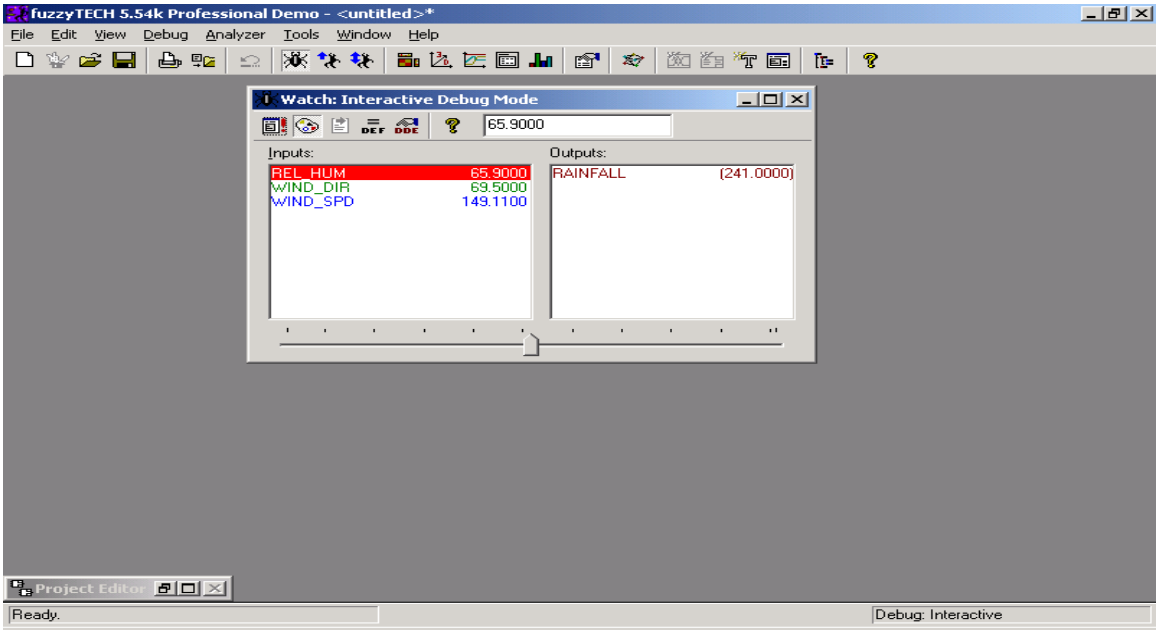


Figure 3.15: Interactive Debug Mode With The Watch Window Activated

- **NEUROFUZZY MODULE TRAINING WITH THE SAMPLE DATA:** Training is initiated using the **[NF Training]** button. The Learn Control Dialog before and after training, as in Figure's 3.16 and 3.17, is used to supervise and guide the training progress. The Plot Area enables the display of a NeuroFuzzy Error Plot, a NeuroFuzzy Error Statistic or both. The Error Plot shows the training history of the Maximum Deviation and the Average Deviation while the Statistic Plot classifies the deviations of all samples. The number of samples fitting into each Deviation class is displayed [10].

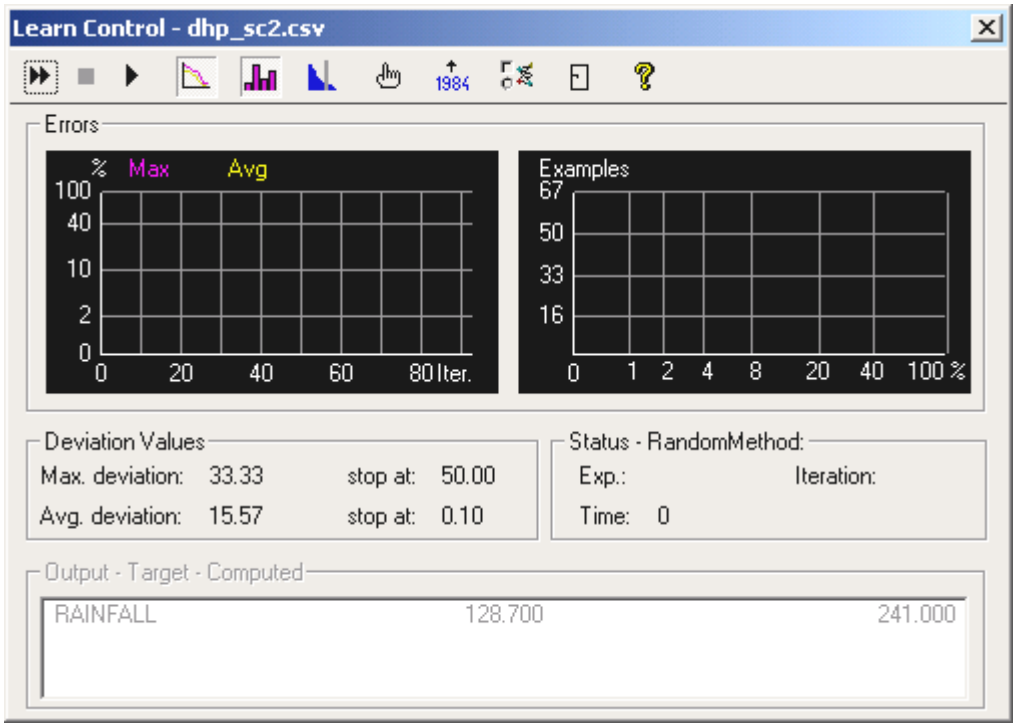


Figure 3.16: NeuroFuzzy Learning Control Dialog Before Start of Training

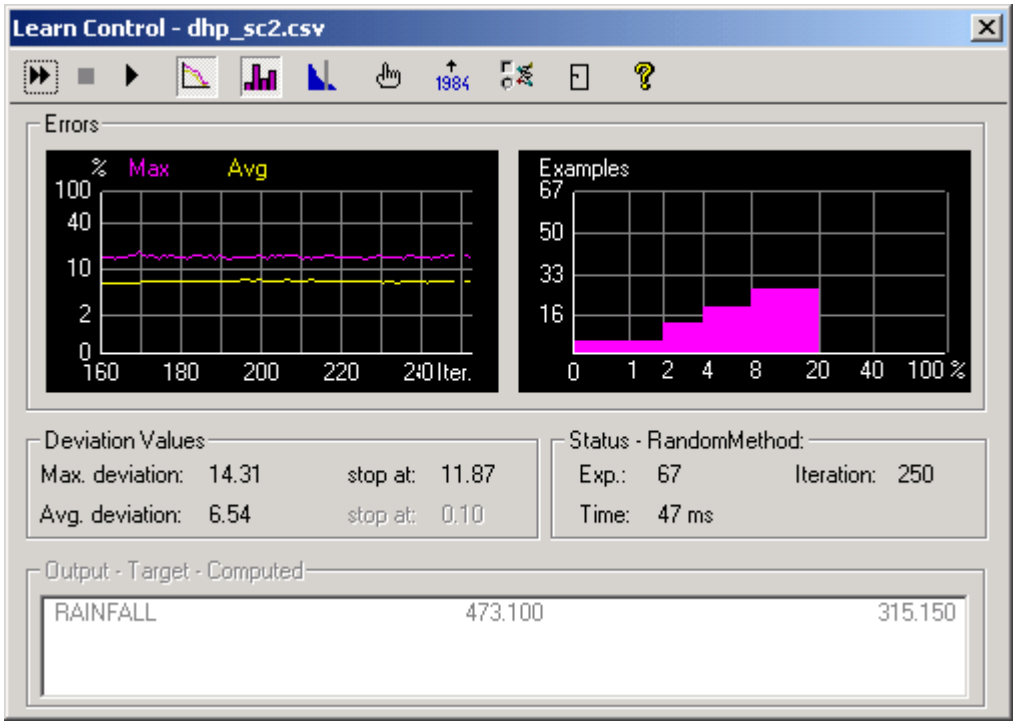


Figure 3.17: NeuroFuzzy Learning Control Dialog After Training

The group **"Deviation Values"** displays the values of the current errors relative to the selected error condition thresholds. The group **"Status"** displays various types of information: number of current sample, number of training cycles and iteration timer value. Enabling the **[Update All Windows]** button causes the NeuroFuzzy Module to update contents of all opened editors and analyzers for each training example. This allows the training progress to be watched interactively. The NeuroFuzzy Module modifies the Degree of Support (DoS) of the rules and moves the Membership Functions of the output variable [10].

The List Box "**Output - Target - Computed"** displays a row for each output variable under training. "**Target"** displays the desired output value (from the sample) while **"Computed"** displays the value computed by the current Fuzzy Logic System under training. This function helps to determine which samples are causing any significant remaining error [10].

vi) **EVALUATING SYSTEM PERFORMANCE:** In order to monitor the system behaviour relative to the results produced by all samples, the PERFORM COMPUTATION command is executed after any complete iteration. At the end of each iteration, the perform run is computed to determine the real errors and the average error of all samples is also computed and displayed. The error plot and statistics and the iteration counter are all updated as well. The perform run also checks the break criteria of the training procedure and if the system fulfills all valid criteria, the training is stopped. After completion of the actual training, the result is a 'pure' Fuzzy Logic System, which, can be tested, in the working phase

with the Watch Window (Figure 3.15) and analyzers (e.g. 3D plot, as in Figure 3.18) of the fuzzyTECH NeuroFuzzy Module. If the result is not satisfactory, some development steps can be repeated [10].

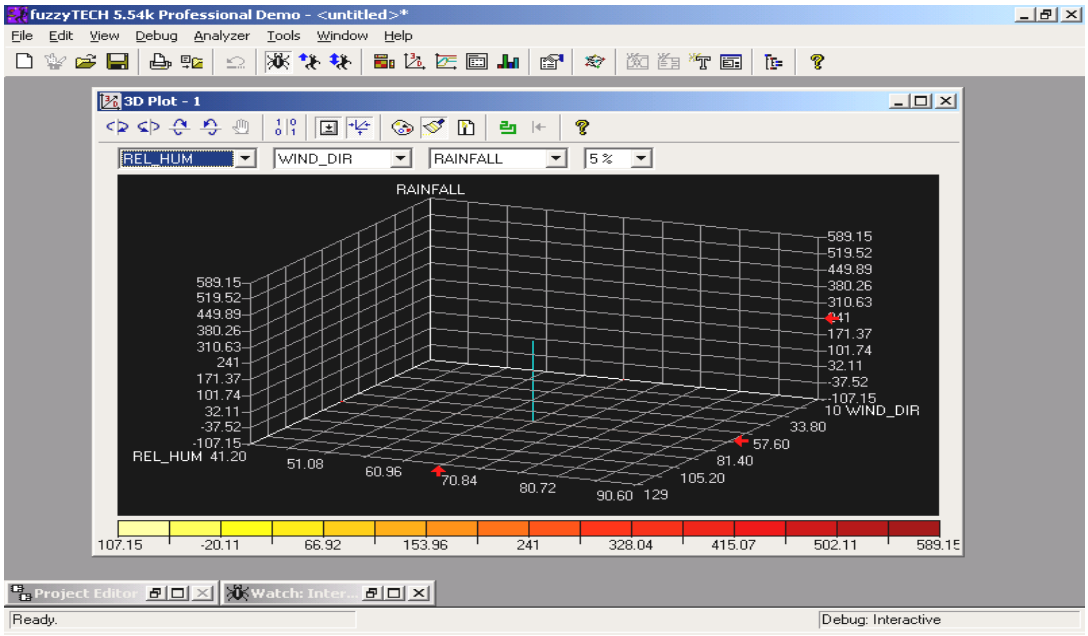


Figure 3.18: 3D Plot Showing Transfer Surface Before Training

vii) **MANUAL OPTIMIZATION:** The resultant Fuzzy Logic System can further be manually optimized using an interactive approach with the aid of the Watch Window. Usually the rule set subjected to the NeuroFuzzy training includes more rules than required in the final solution, and using either expert or common knowledge, the functionally redundant or unnecessary rules can be deleted using the "**Alpha Cut (α - cut)**" function of the Rule Block utility [10]. The α - cut of a fuzzy set A is given by the Membership Function (MBF):

$$\mu_{A^\alpha}(x) = \begin{cases} \mu_A(x) & when\ \mu_A(x) \geq \alpha \\ 0 & otherwise \end{cases} \qquad 3.1$$

where $\alpha \in [0,1]$

In this case (model formed by three input variables and one output variable, each having five linguistic terms, the size of the initial rule base is 5^{3+1}), a total of **six hundred and twenty five (625)** rules**,** as in Appendix A2, were generated. Upon application of α - cut value of 0.0 as in Figure 3.19, the rule base was reduced to **fifty seven (57)** as in Appendix A3**.**

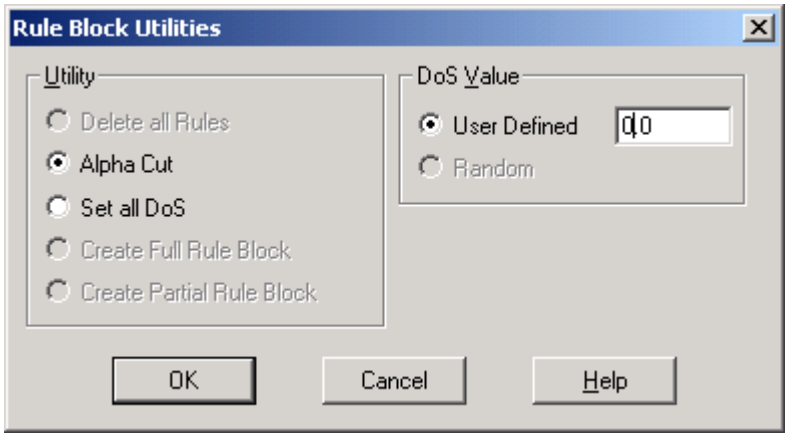


Figure 3.19: Alpha-Cut Dialog Box

viii) **IMPLEMENTATION:** The result of the NeuroFuzzy Module training is a 'pure' Fuzzy Logic System, which, can be, implemented on computers, microcontrollers or industrial controllers [10]. The 'pure' Fuzzy Logic System, named the Rain Forecast Model (RFM), is implemented using the Fuzzy Technology Language (FTL).

3.2 RAIN FORECAST MODEL

The Fuzzy Technology Language (FTL) is a hardware and vendor independent description language for Fuzzy Logic Systems with definition formats and no loops and branches. Such chip manufacturers as Intel, SGS-Thomson, Siemens, Texas Instruments, Microchip, etc support the Fuzzy Technology Language (FTL). The Fuzzy Technology Language (FTL) consists basically of two entities:

i) **Objects:** Each object consists of an object name and an object body in **"{ }".** Within an object body, other objects and slots can be defined.

ii) **Slots:** A slot consists of a slot name to the left of an **"="** and a value for the slot to the right.

Comments are put into **"/* … */"** marks [81].

The general flow of the Rainfall Forecast Model using the Fuzzy Technology Language (FTL) is described as follows:

```
PROJECT{
/*Main section for project data*/
SHELLOPTIONS{
        /*Section containing terminal configurations for the project*/
        }/* SHELLOPTIONS*/
        MODEL{
        /*Section containing the entire model definitions (Variable and Object Section)*/
                VARIABLE_SECTION{
                /*Section containing all definitions of linguistic variables*/
                        LVAR{
                                TERM{
                                }/*TERM*/
                        }/*LVAR*/
                }/*VARIABLE SECTION*/
                OBJECT_SECTION{
                /*Section containing all structural elements of a Fuzzy Logic System*/
                        RULEBLOCK{
                                RULES{
                                }/*RULES*/
                        }/*RULEBLOCK*/
                        }/*OBJECT SECTION*/
```

```
        }/*MODEL*/
}/*PROJECT*/
```

The complete Fuzzy Technology Language (FTL) program listing for the Rain Forecast Model is shown in Appendix A4. The Rain Forecast Model is as shown in Figure 3.20 and the Membership Functions for the inputs (Relative Humidity, Wind Speed and Wind Direction) and the output (Rainfall) are shown in Figure's 3.21 – 3.24.

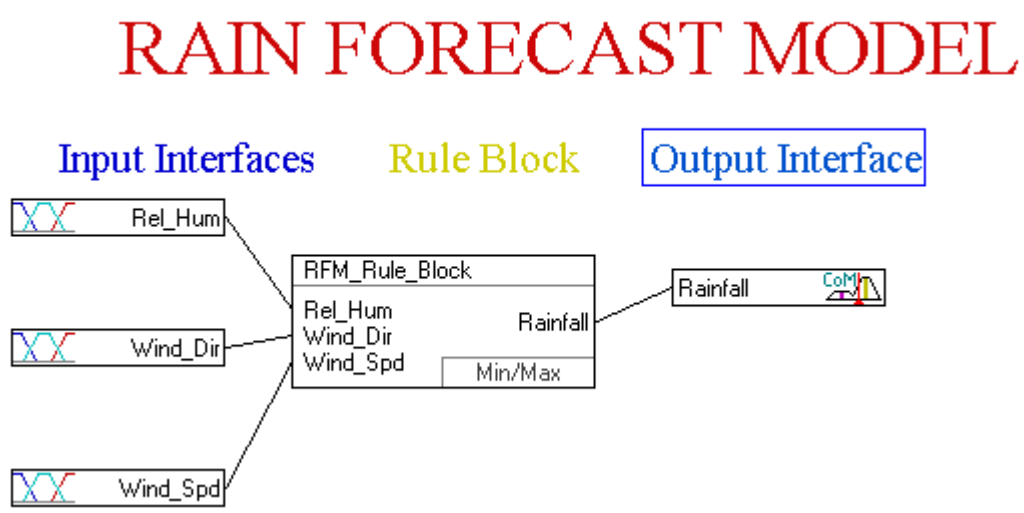


Figure 3.20: Structure of the Rain Forecast Model Fuzzy Logic System

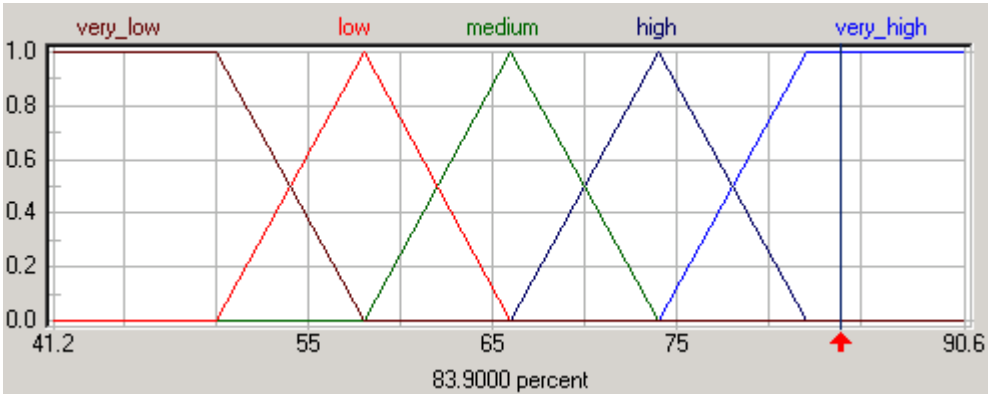


Figure 3.21: Membership Function (MBF) of Relative Humidity (Rel_Hum)

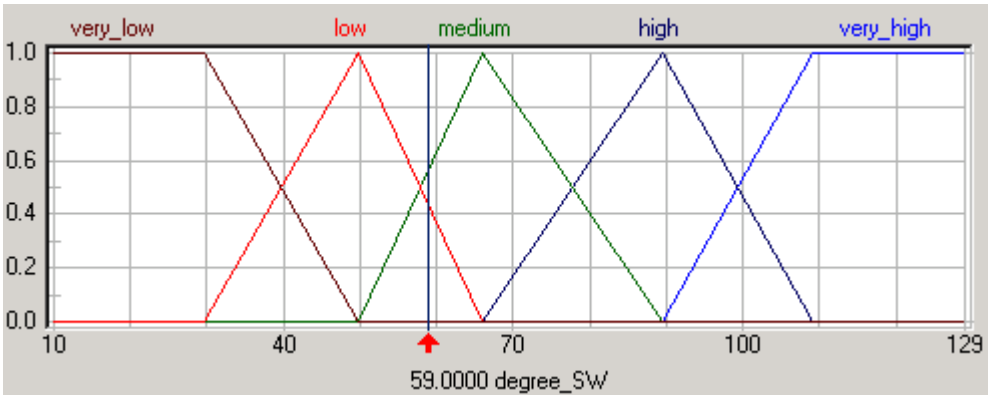


Figure 3.22: Membership Function (MBF) of Wind Direction (Wind_Dir)

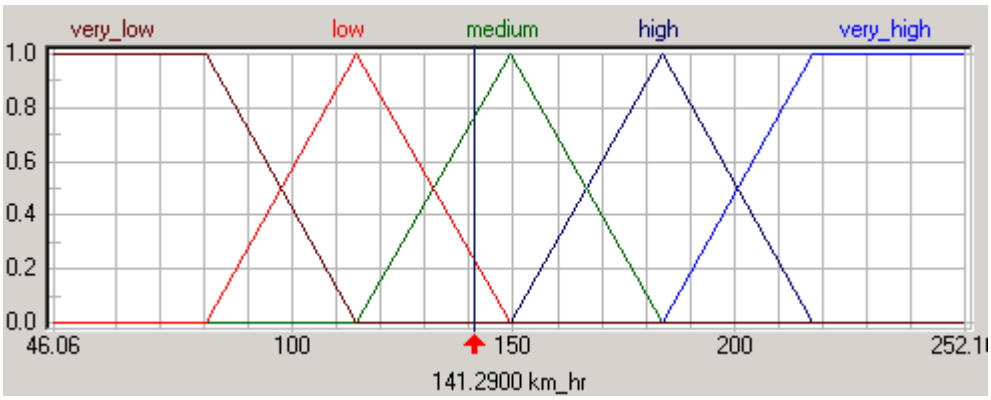


Figure 3.23: Membership Function (MBF) of Wind Speed (Wind_Spd)

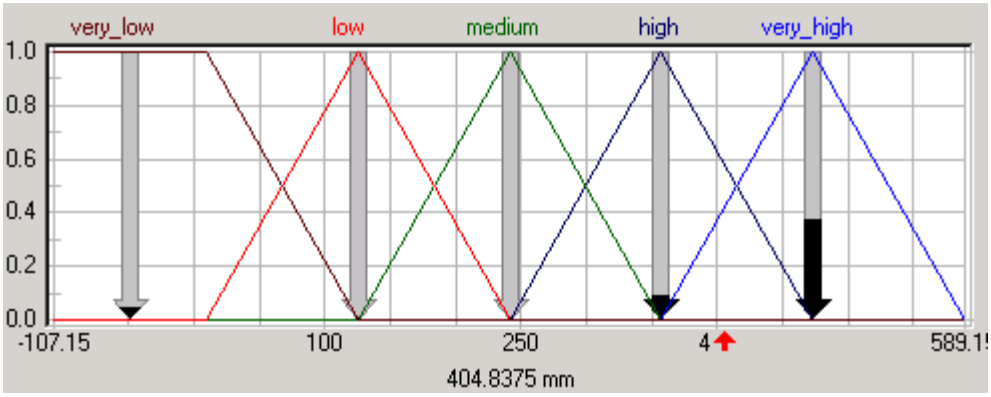


Figure 3.24: Membership Function (MBF) of Rainfall (Rainfall)

The Rain Forecast Model, after optimization, has fifty-seven (57) rules and the parameters of the Rule Block (RFM_Rule_Block) are as shown in Table 3.1. The complete rule base is as shown in Appendix A3

Table 3.1: Parameters of the Rain Forecast Model Rule Block (RFM_Rule_Block)

Aggregation:	MINMAX
Parameter:	0.00
Result Aggregation:	MAX
Number of Inputs:	3
Number of Outputs:	1
Number of Rules:	57

The Rain Forecast Model is then tested for its ability to generalize with previously unseen data, which is the validation data of 2003 to 2005, using the interactive debug Watch Window. The Watch Windows of Figure's 3.25 – 3.27 show the outcomes for August 2003, July 2004 and September 2005 as examples.

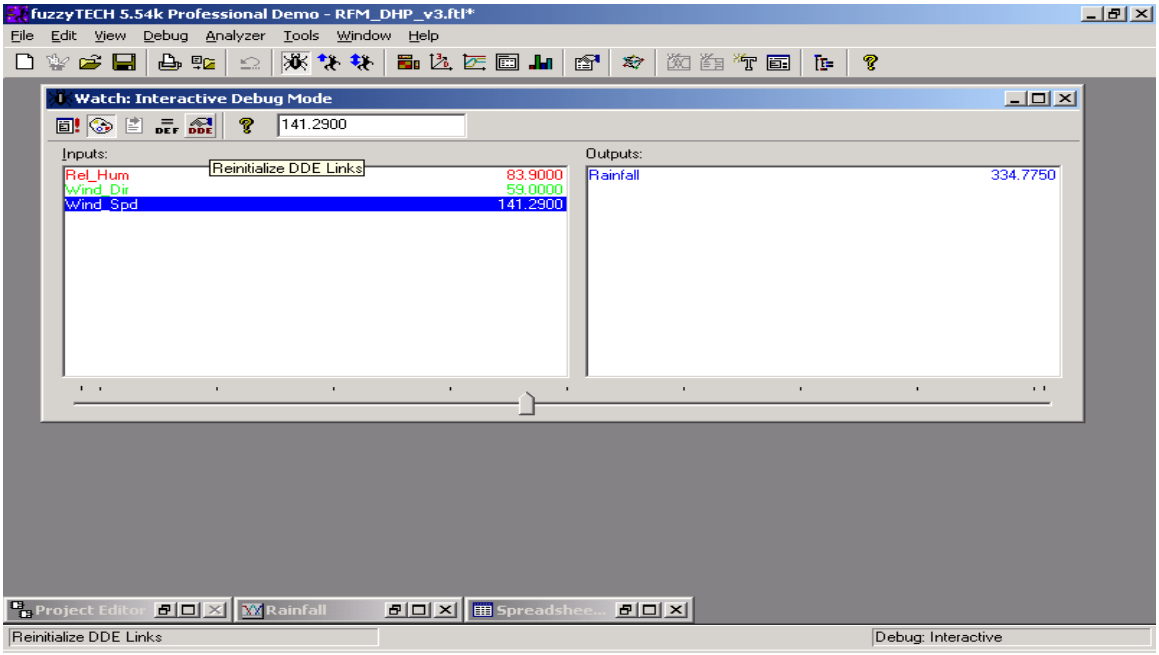


Figure 3.25: Interactive Watch Window With Example Data of August 2003.

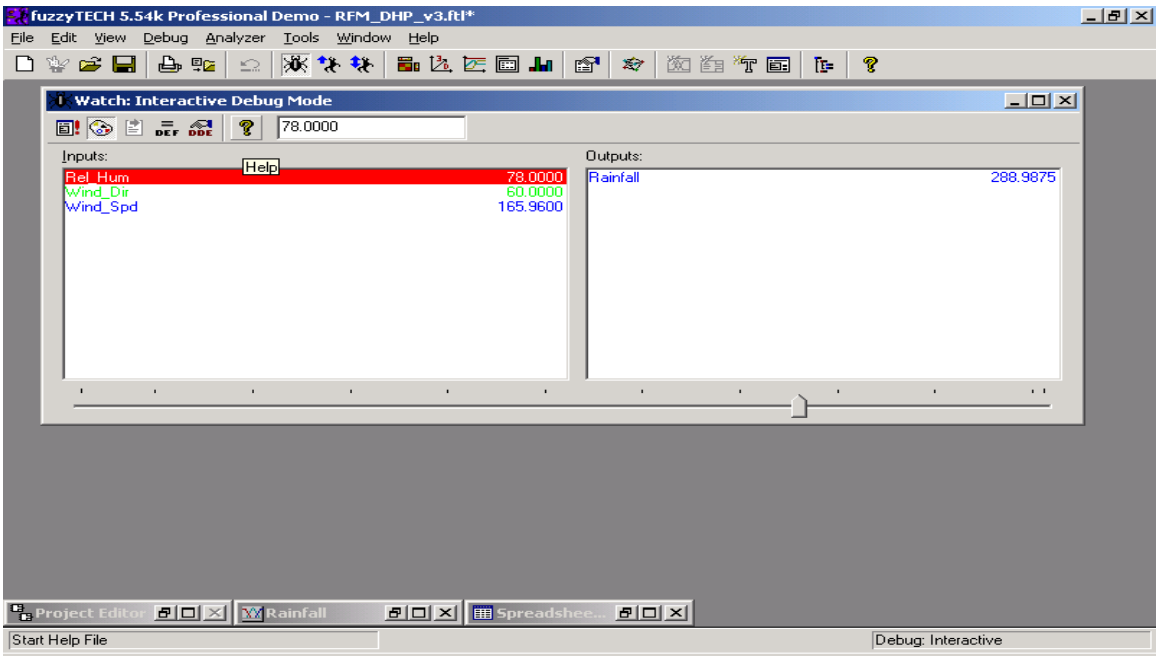


Figure 3.26: Interactive Watch Window With Example Data of July 2004.

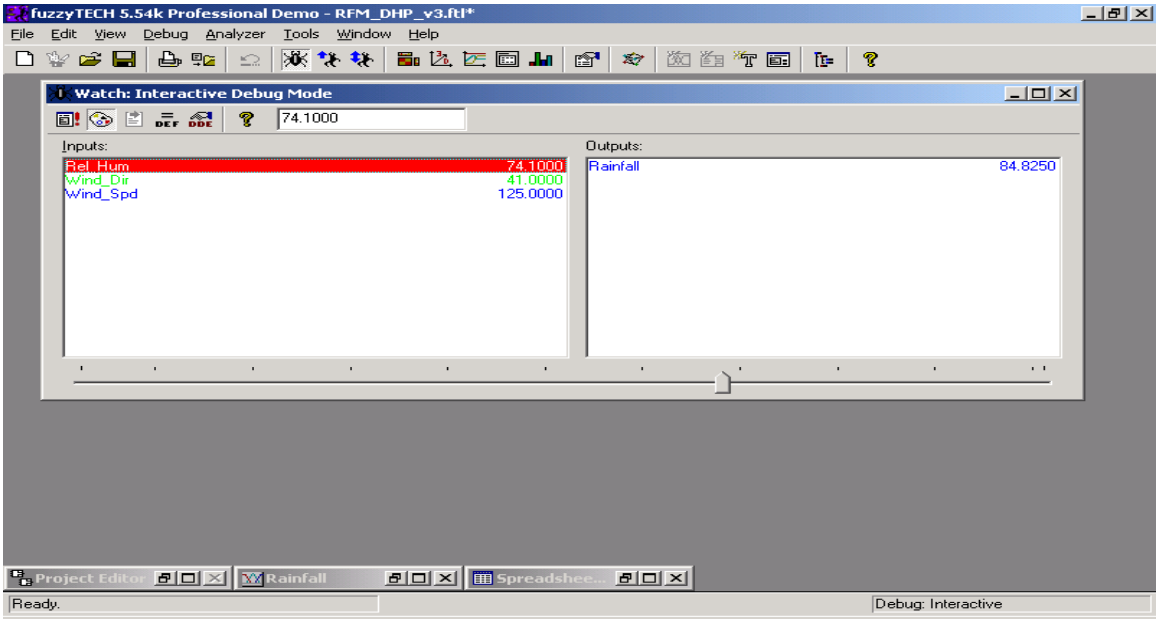
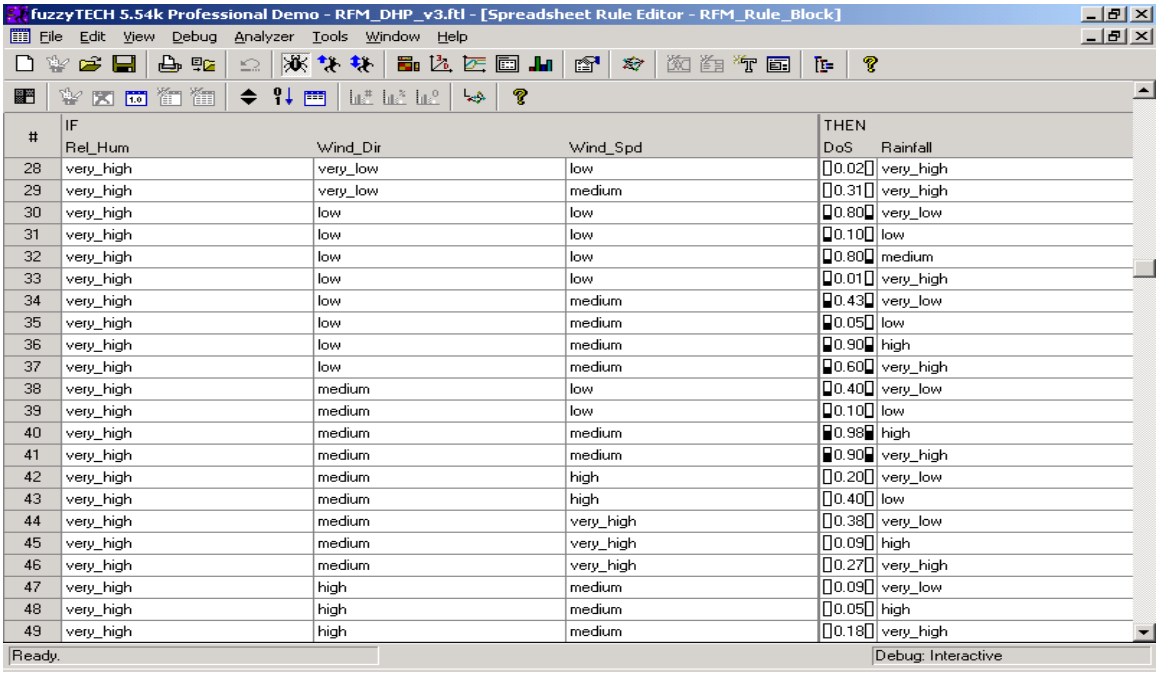


Figure 3.27: Interactive Watch Window With Example Data of September 2005.

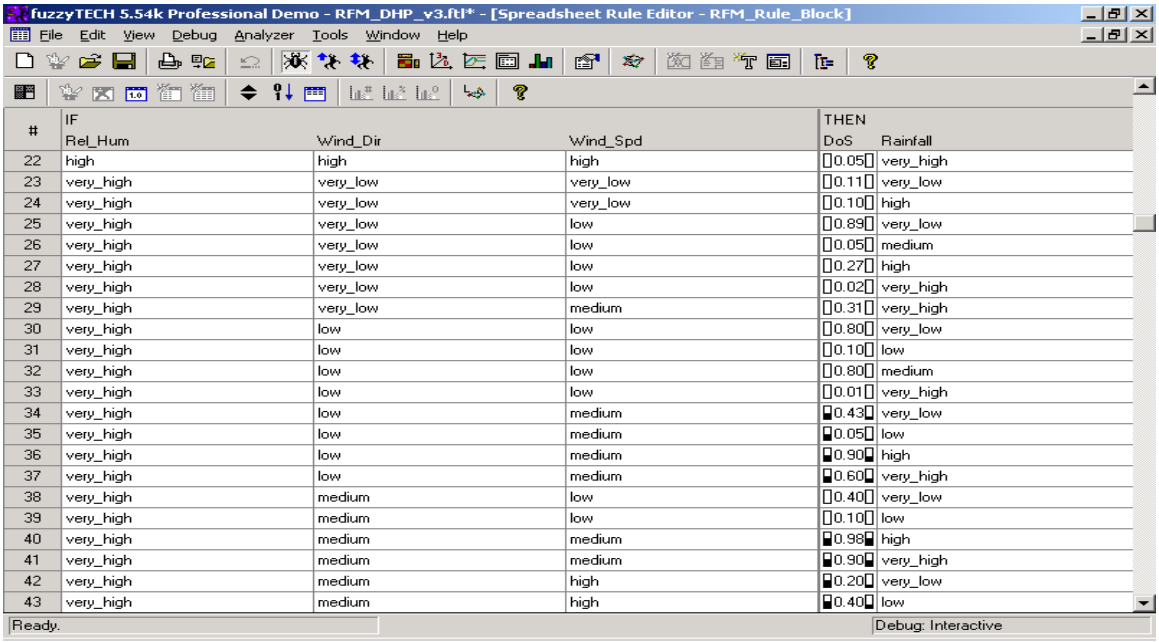
For each of the entries in the Watch Window, certain rules are fired, as in Figure's 3.28 to 3.30 for August 2003, July 2004 and September 2005 respectively and these modify the Membership Functions of the output (Rainfall) Variable. The 3D plots, as in Figure's

3.31 to 3.33 (using the example data of July 2004) show how this affects the transfer surface.



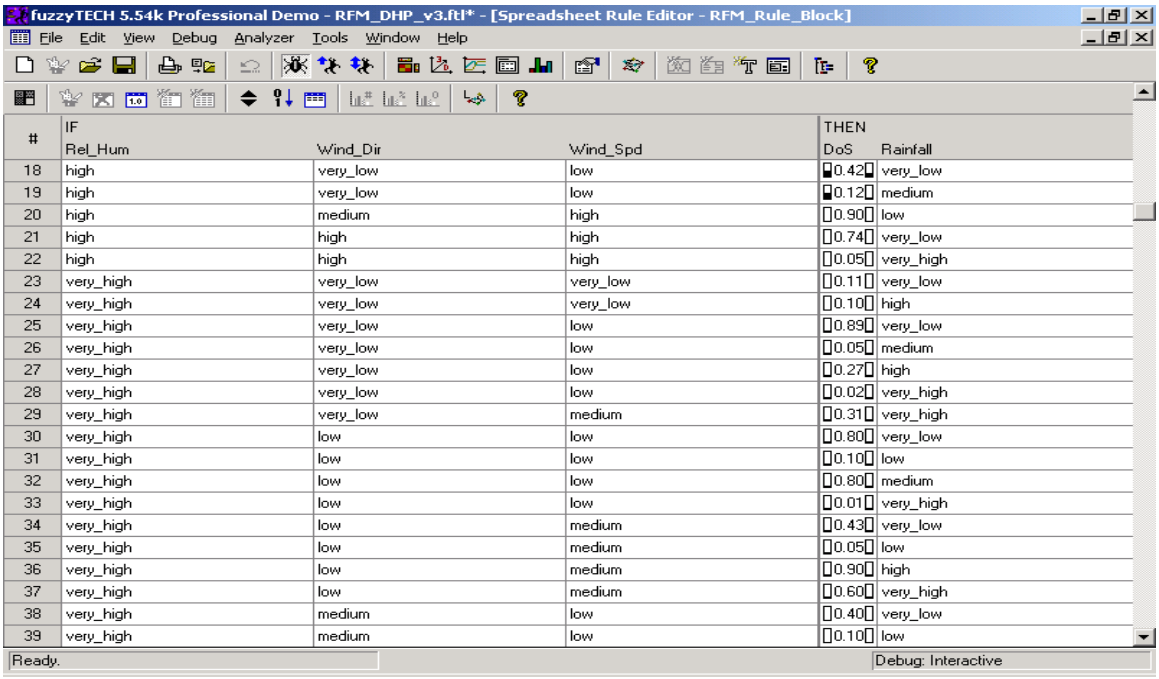
#	IF			THEN	
	Rel_Hum	Wind_Dir	Wind_Spd	DoS	Rainfall
28	very_high	very_low	low	0.02	very_high
29	very_high	very_low	medium	0.31	very_high
30	very_high	low	low	0.80	very_low
31	very_high	low	low	0.10	low
32	very_high	low	low	0.80	medium
33	very_high	low	low	0.01	very_high
34	very_high	low	medium	0.43	very_low
35	very_high	low	medium	0.05	low
36	very_high	low	medium	0.90	high
37	very_high	low	medium	0.60	very_high
38	very_high	medium	low	0.40	very_low
39	very_high	medium	low	0.10	low
40	very_high	medium	medium	0.98	high
41	very_high	medium	medium	0.90	very_high
42	very_high	medium	high	0.20	very_low
43	very_high	medium	high	0.40	low
44	very_high	medium	very_high	0.38	very_low
45	very_high	medium	very_high	0.09	high
46	very_high	medium	very_high	0.27	very_high
47	very_high	high	medium	0.09	very_low
48	very_high	high	medium	0.05	high
49	very_high	high	medium	0.18	very_high

Figure 3.28: Rules Fired For Example Data of August 2003.



#	IF			THEN	
	Rel_Hum	Wind_Dir	Wind_Spd	DoS	Rainfall
22	high	high	high	0.05	very_high
23	very_high	very_low	very_low	0.11	very_low
24	very_high	very_low	very_low	0.10	high
25	very_high	very_low	low	0.89	very_low
26	very_high	very_low	low	0.05	medium
27	very_high	very_low	low	0.27	high
28	very_high	very_low	low	0.02	very_high
29	very_high	very_low	medium	0.31	very_high
30	very_high	low	low	0.80	very_low
31	very_high	low	low	0.10	low
32	very_high	low	low	0.80	medium
33	very_high	low	low	0.01	very_high
34	very_high	low	medium	0.43	very_low
35	very_high	low	medium	0.05	low
36	very_high	low	medium	0.90	high
37	very_high	low	medium	0.60	very_high
38	very_high	medium	low	0.40	very_low
39	very_high	medium	low	0.10	low
40	very_high	medium	medium	0.98	high
41	very_high	medium	medium	0.90	very_high
42	very_high	medium	high	0.20	very_low
43	very_high	medium	high	0.40	low

Figure 3.29: Rules Fired For Example Data of July 2004.



#	IF			THEN	
	Rel_Hum	Wind_Dir	Wind_Spd	DoS	Rainfall
18	high	very_low	low	0.42	very_low
19	high	very_low	low	0.12	medium
20	high	medium	high	0.90	low
21	high	high	high	0.74	very_low
22	high	high	high	0.05	very_high
23	very_high	very_low	very_low	0.11	very_low
24	very_high	very_low	very_low	0.10	high
25	very_high	very_low	low	0.89	very_low
26	very_high	very_low	low	0.05	medium
27	very_high	very_low	low	0.27	high
28	very_high	very_low	low	0.02	very_high
29	very_high	very_low	medium	0.31	very_high
30	very_high	low	low	0.80	very_low
31	very_high	low	low	0.10	low
32	very_high	low	low	0.80	medium
33	very_high	low	low	0.01	very_high
34	very_high	low	medium	0.43	very_low
35	very_high	low	medium	0.05	low
36	very_high	low	medium	0.90	high
37	very_high	low	medium	0.60	very_high
38	very_high	medium	low	0.40	very_low
39	very_high	medium	low	0.10	low

Figure 3.30: Rules Fired For Example Data of September 2005.

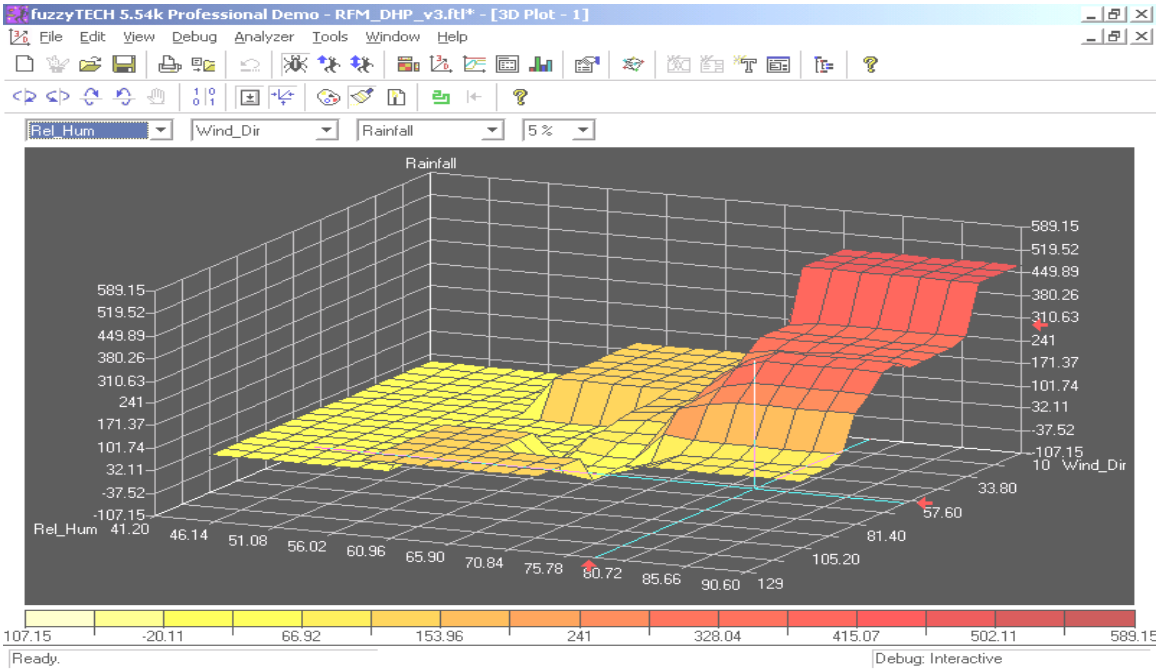


Figure 3.31: 3D Plot of Rel_Hum, Wind_Dir and Rainfall (July 2004)

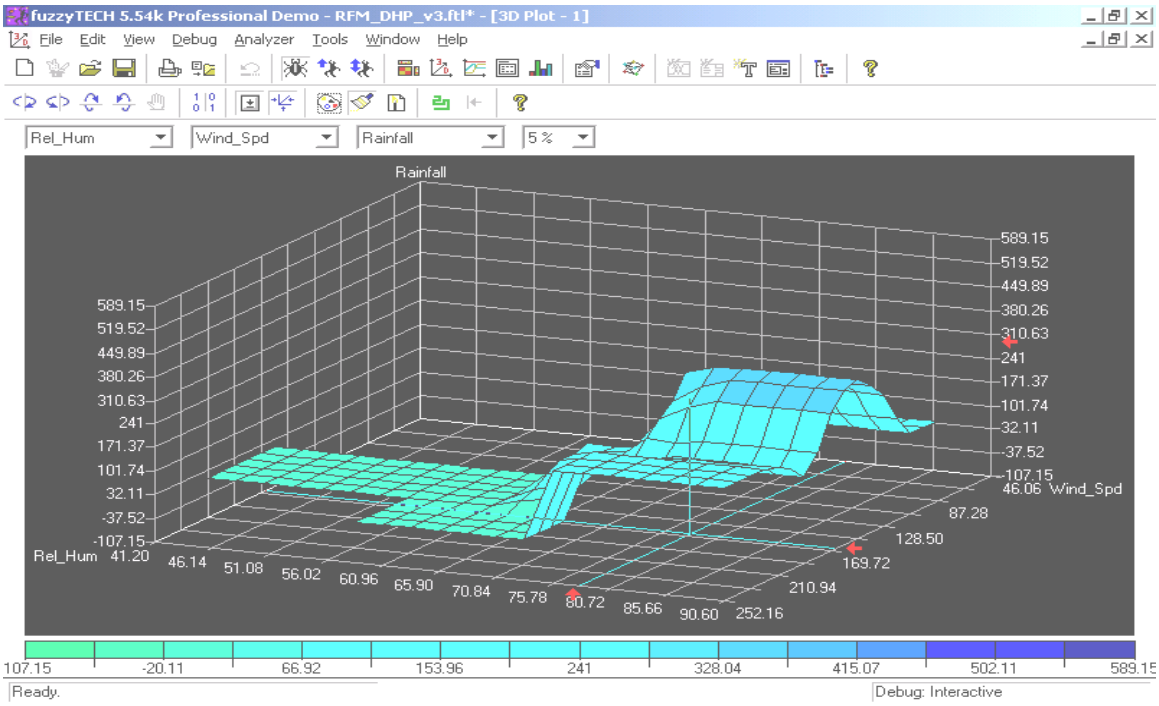


Figure 3.32: 3D Plot of Rel_Hum, Wind_Spd and Rainfall (July 2004)

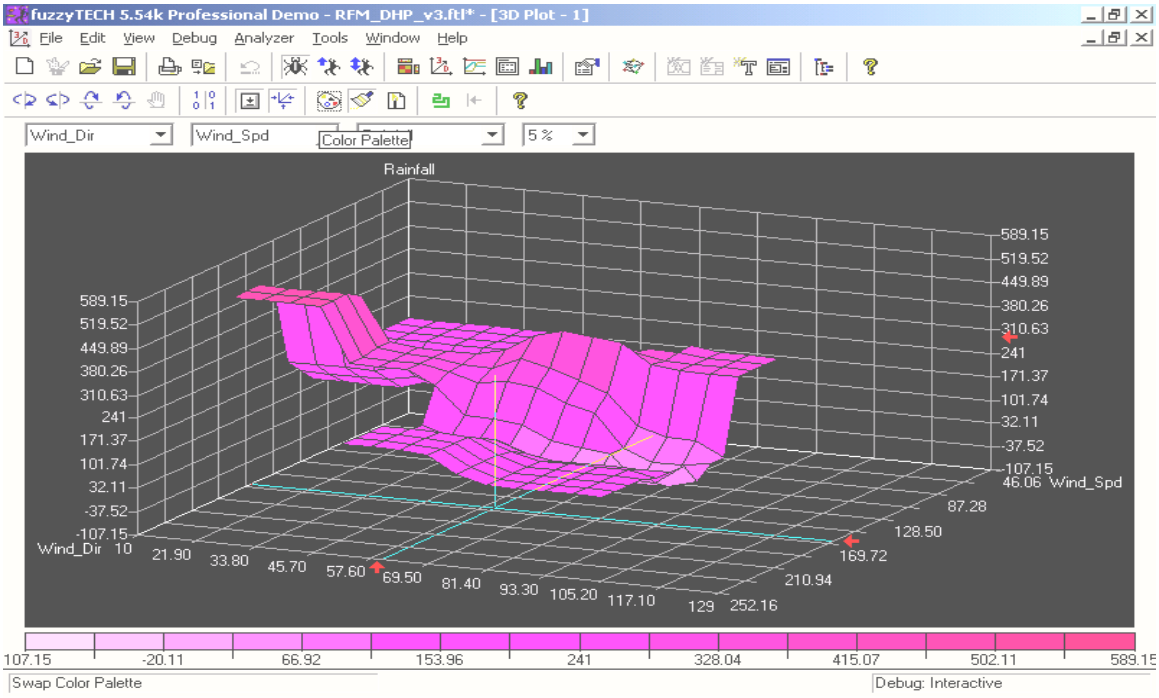


Figure 3.33: 3D Plot of Wind_Dir, Wind_Spd and Rainfall (July 2004)

CHAPTER FOUR

TIME SERIES PREDICTION OF WEATHER VARIABLES

4.1 INTRODUCTION

The Rain Forecast Model developed for forecasting monthly amount of rainfall depends on weather variable inputs, namely Relative Humidity, Wind Direction, and Wind Speed. These are basically time series data. A time series is a set of observations of a variable at regular intervals over time. A basic requirement for time series is that the data be displayed in the order in which they occurred, since it is possible that successive observations may probably be dependent [82][83]. According to **Rojas [82],** the challenge of predicting future values of a time series spans a variety of disciplines and as such time series techniques find application in such diverse data sets as equity market prices, disease control, meteorological measurements, astronomic observations, and others. The time series plots for Relative Humidity, Wind Direction, and Wind Speed are as shown in Figure's 4.1 to 4.3.

Time series analysis relies, at least in part, on understanding or exploiting the dependence among the observations. Because of this reliance, the goal of time series prediction can, therefore, be stated thus: **Given a sequence up to time t,** $x(1), x(2), ..., x(t)$, **find the continuation** $x(t+1), x(t+2), ...$ [82]. The method to be employed in finding this continuation is a hybrid of the Simple Moving Average method and that of Exponential Smoothing.

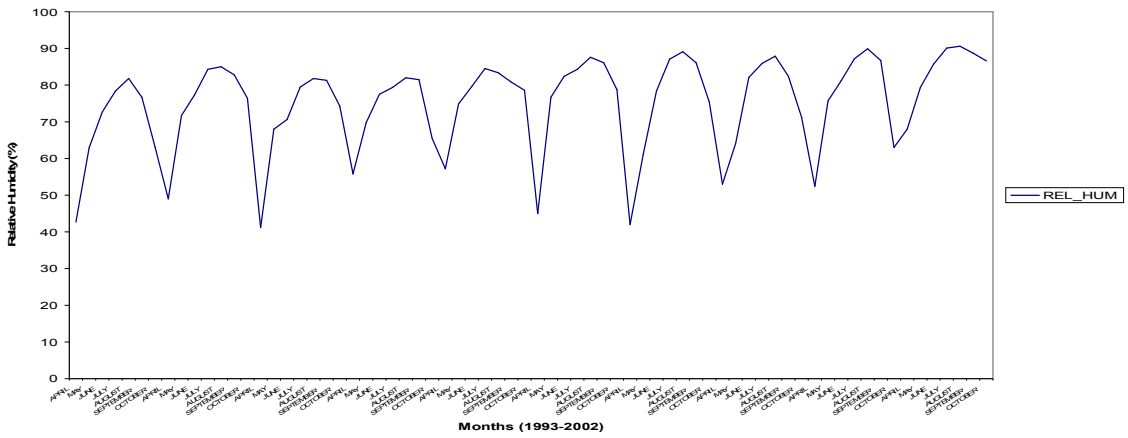


Figure 4.1: Time Series Plot of Relative Humidity.

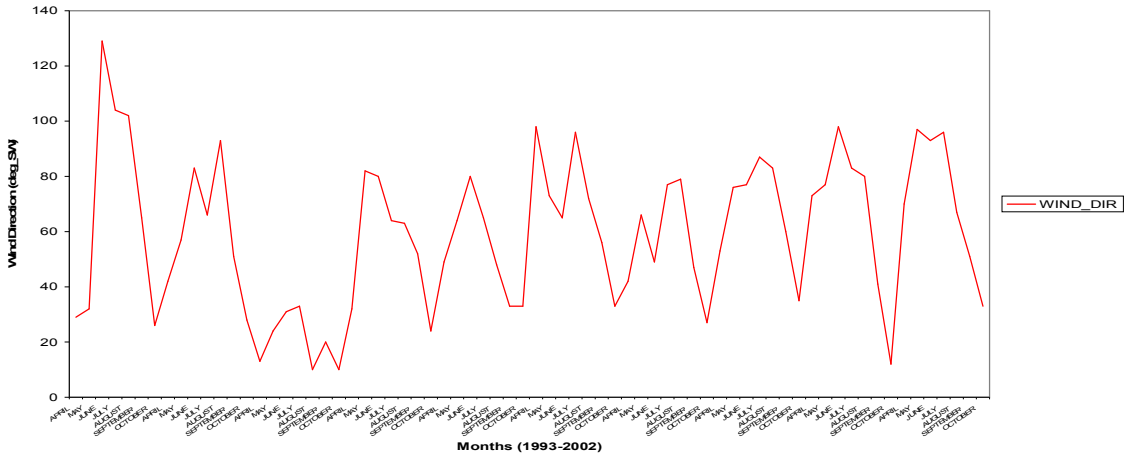


Figure 4.2: Time Series Plot of Wind Direction

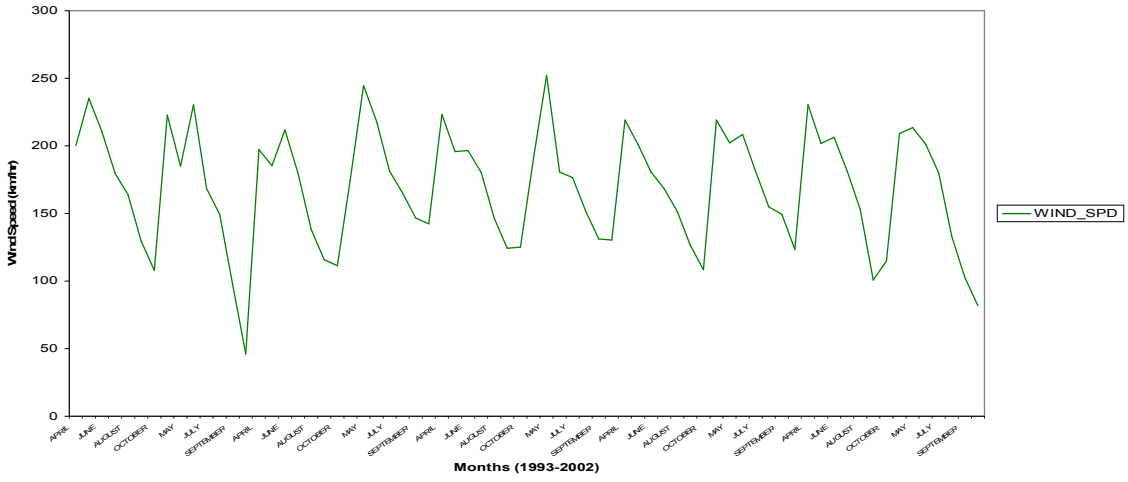


Figure 4.3: Time Series Plot of Wind Speed

4.2 SIMPLE MOVING AVERAGES

The basic components of a time series are:

i) **TREND**, which is the underlying direction (upward or downward tendency) and rate of change in a time series, when allowance has been made for the other components.

ii) **SEASONAL EFFECTS**, which describe similar variations occurring during corresponding periods.

iii) **CYCLICAL FACTORS**, which describe any regular fluctuations.

The simple moving averages method is a smoothing technique that uses a type of average that is adjusted to allow for seasonal or cyclical components of a time series and which helps make the long term trends clearer [83][22].

In the simple moving average method, only the **n** most recent periods of data points need to be maintained. At the end of each period, the oldest period's data is discarded and the newest period's data is added to the database. The database is then divided by **n** and used as a forecast for the next period [83][84].

In order to determine the moving average $x(t+1)$ for a period $(t+1)$, the following expression is used:

$$x(t+1) = \frac{\left[D_t + D_{t-1} + \ldots + D_{t-n-1}\right]}{n}, \quad n \le t \qquad 4.1$$

Where, n = number of observations used in the calculation

D = time series data

A critical issue in using the simple moving average method is in the choice of **n**. It can be determined by selecting a value that minimizes the Mean Squared Error (MSE) of the forecasting, which is given by [84]

$$MSE = \frac{\sum_{t=1}^{n} (D_t - x_t)^2}{n} \qquad 4.2$$

The simple moving average method was used for each of the weather variables on a monthly basis, that is, say for Relative Humidity, April 1993 to April 2010 and so on. A moving 'window' of three-year, five-year, seven-year and nine-year width (n=3, n=5, n=7 and n=9) respectively was applied on the validation data of 2003 to 2005 in order to determine the choice of n to minimize the Mean Squared Error (MSE). The result is as shown in Table 4.1.

Table 4.1: MSE Values For n=3, n=5, n=7 and n=9

	2003				**2004**				**2005**			
	n=3	n=5	n=7	n=9	n=3	n=5	n=7	n=9	n=3	n=5	n=7	n=9
REL_HUM	111.1	73.9	59.96	**53.08**	285.4	87.68	94.4	**60.09**	150.1	43.94	30.97	**15.65**
WIND_DIR	685.4	116.33	63.34	**38.76**	928.6	222.4	179.58	**103.2**	1342	401.78	292.44	**160.6**
WIND_SPD	568.6	185.1	251.33	**152.5**	285.4	191.6	222.89	**125.5**	150.1	81.97	40.395	**79.66**

On the basis of Table 4.1, the choice of **n=9** was decided upon since, on the overall, it exhibits the least Mean Squared Error (MSE) values. The Mean Squared Error (MSE) values obtained, as in Table 4.1, also indicate that the higher the value of **n** the lower the value of the Mean Squared Error (MSE). It should also be noted that in this case the highest possible value of n is 9. The selected value is then used to determine preliminary forecast values for the weather variables from 2003 to 2010. The Exponential Smoothing

method is then applied to obtain the forecast values to be used with the developed Rain Forecast Model (RFM).

4.3 SIMPLE EXPONENTIAL SMOOTHING METHOD

Exponential Smoothing is a forecasting technique that weighs past data in an exponential manner so that the most recent data carries more weight. This means that an exponentially smoothed moving average is based not on a sequential average of individual time periods but on the most recent data and the average prior to it adjusted by a smoothing constant, α. The smoothing constant, α, is a number in the range [0, 1] with smaller values resulting in less weight given to the current time period. It can be determined on a trial-and-error basis but **Roberts [22]** has determined an empirical expression for the smoothing constant as:

$$\alpha = \frac{1}{\frac{n}{2} - 1} \qquad 4.3$$

Where n is the moving window width [22][85].

The smoothed series is recursively updated as new observations are recorded using the following expression:

$$S(t) = \alpha D(t) + (1 - \alpha) S(t - 1) \qquad 4.4$$

Where S(t) is the value of the smoothed series at period t.

The current smoothed value is an interpolation between the previous smoothed value (S(t-1)) and the current observation, where α controls the closeness of the interpolated value to the most recent observation [85].

Based on the choice of n=9 from Table 4.1, and from equation 4.3, a value of α=0.286 is to be used to determine the forecast values of the weather variables using Microsoft Excel®. A sample Microsoft Excel® spreadsheet showing the procedure is as shown in Figure 4.4.

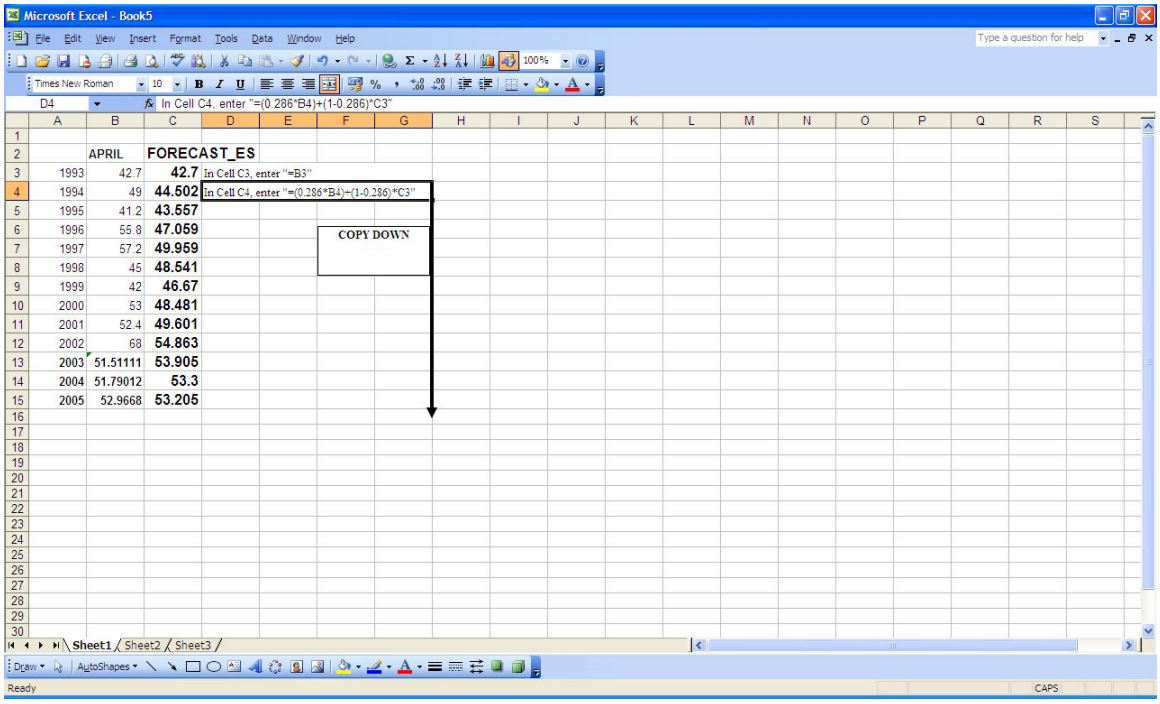


Figure 4.4: Sample Microsoft Excel® Spreadsheet Showing Exponential Smoothing Procedure

The complete forecasted values of the weather variables using the hybrid Moving Averages and Exponential Smoothing methods is shown in Appendix 5. These values are then used to plot the time series plot for the weather variables from 1993 to 2010. The time series plots for Relative Humidity, Wind Direction, and Wind Speed are as shown in Figure's 4.5 to 4.7.

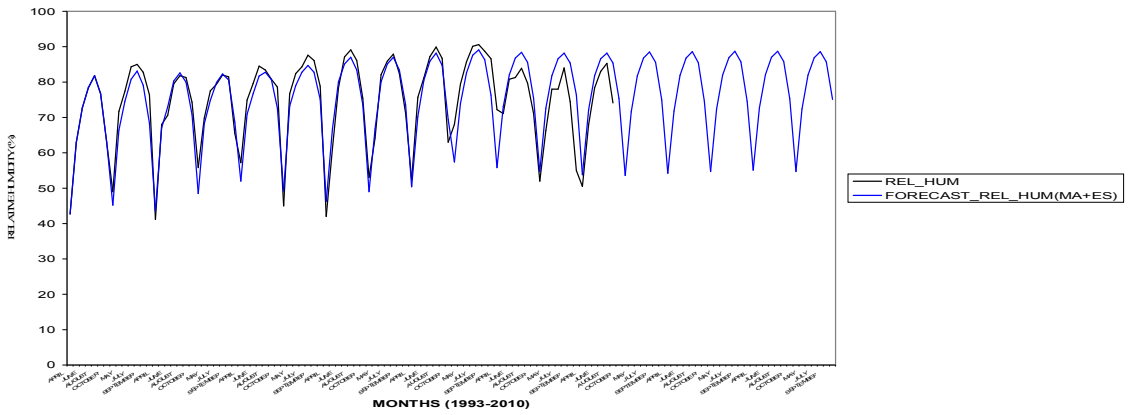


Figure 4.5: Time Series Plot of Relative Humidity (1993-2010).

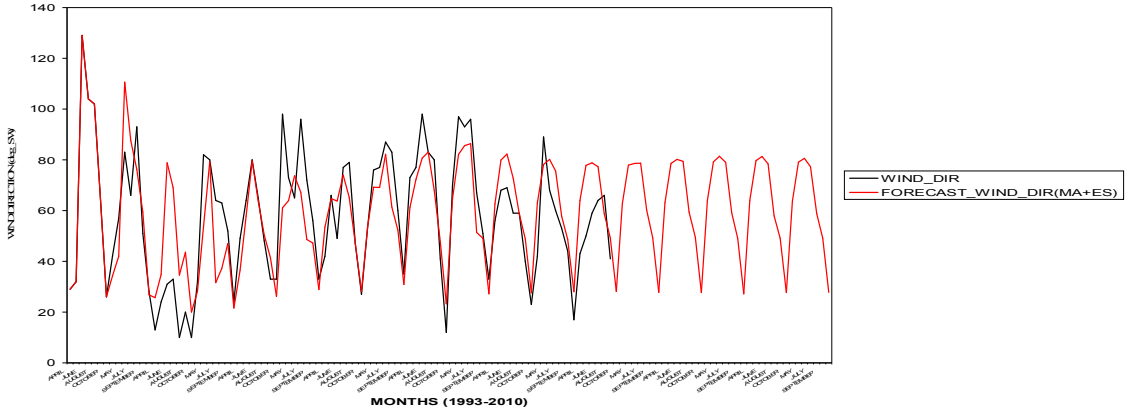


Figure 4.6: Time Series Plot of Wind Direction (1993-2010)

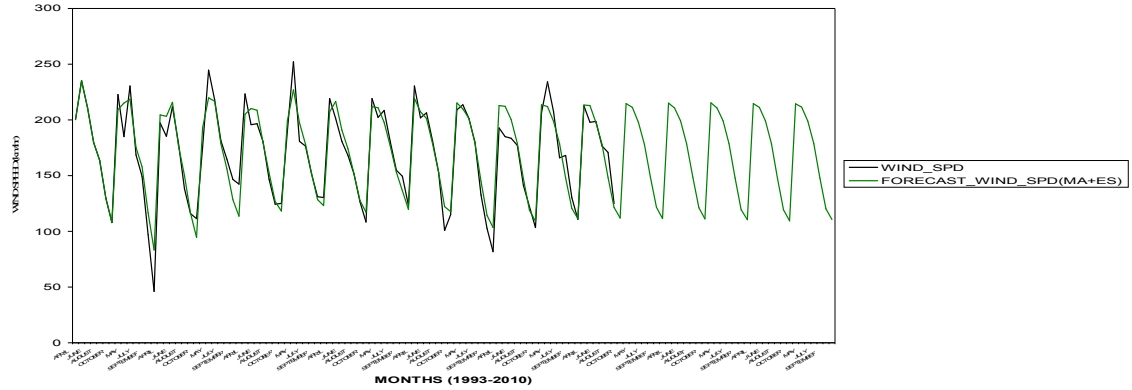


Figure 4.7: Time Series Plot of Wind Speed (1993-2010)

4.4 TESTING WITH THE RAIN FORECAST MODEL (RFM)

The forecasted values of the weather variables (from Appendix A5) are then used to forecast the monthly rainfall amounts for 2003 to 2010 using the developed Rain Forecast Model (RFM). Sample watch window outputs for October 2005, April 2006, August 2008and July 2010 are as shown in Figure's 4.8 – 4.11.

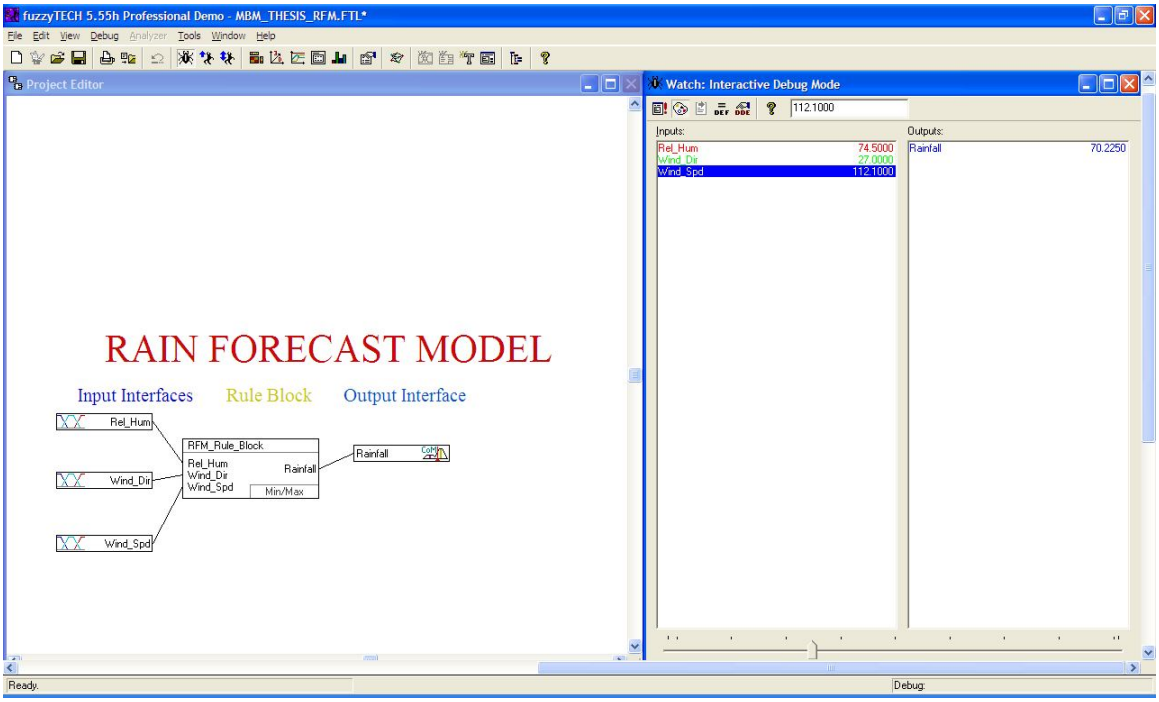


Figure 4.8: Watch Window Output For October 2005

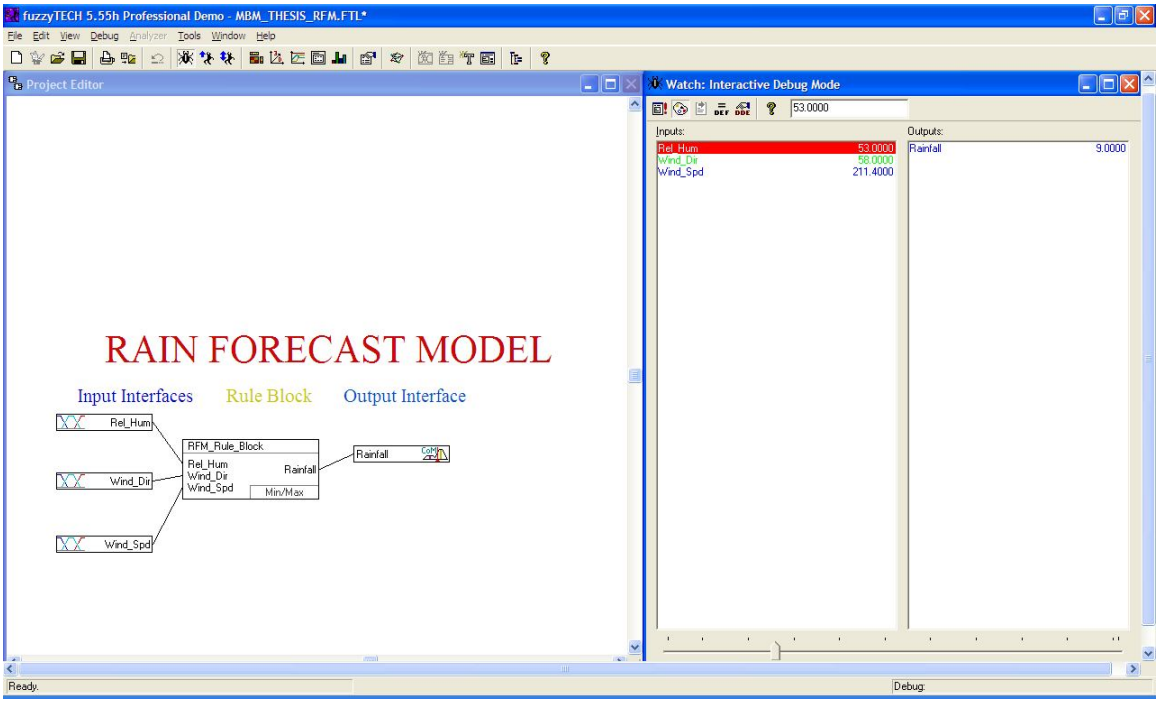


Figure 4.9: Watch Window Output For April 2006

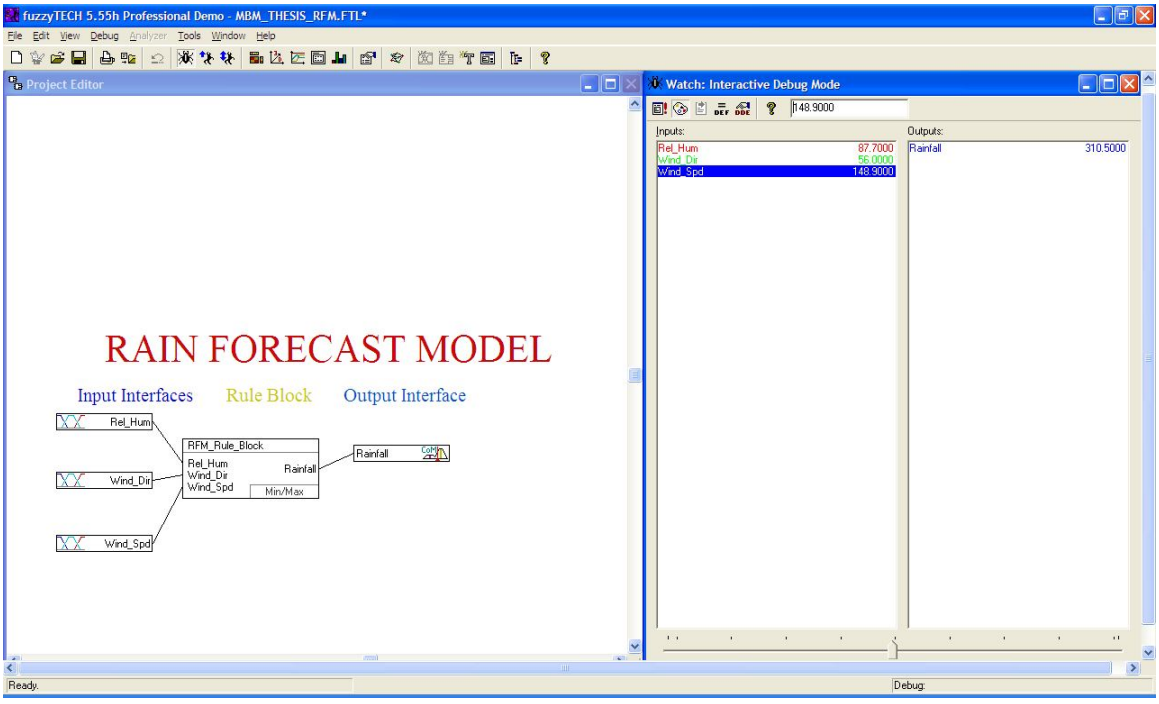


Figure 4.10: Watch Window Output For August 2008

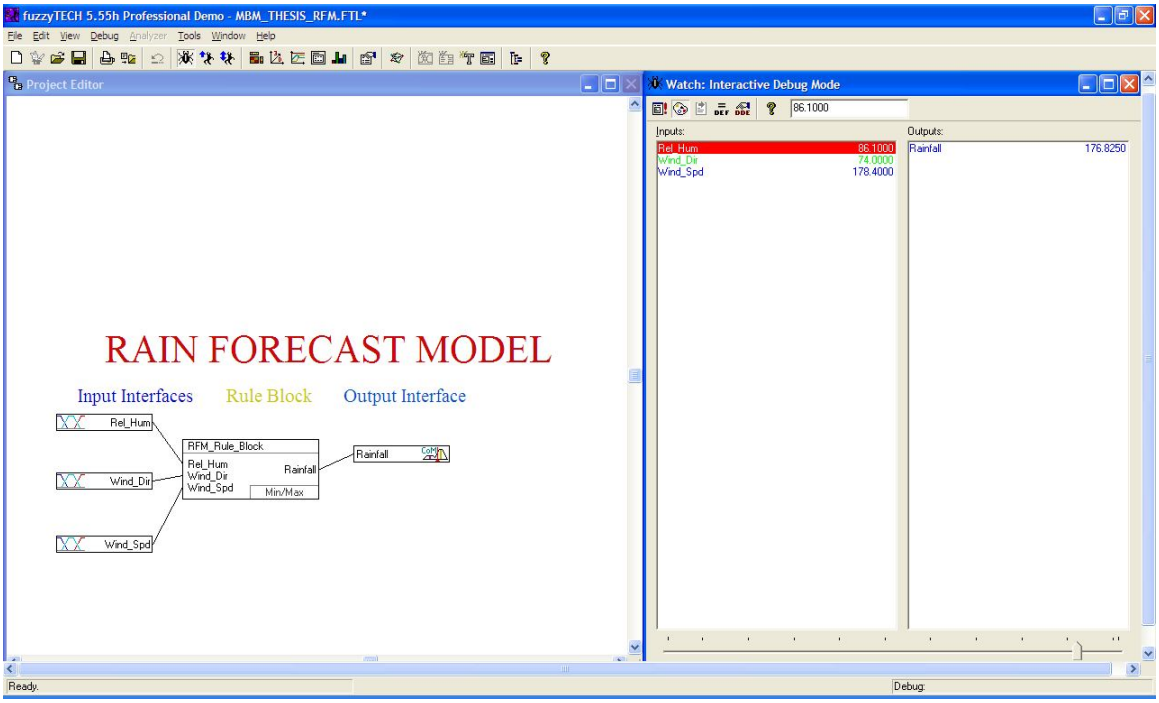


Figure 4.11: Watch Window Output For July 2010

The complete forecasted rainfall data obtained from the Rain Forecast Model (RFM) is shown in Appendix A6. A plot of the rainfall data, forecasted (2003 to 2010) and actual (2003 to 2005) is shown in Figure 4.12. Figure 4.13 shows a plot of the rainfall data, actual (2003 to 2005), forecasted (2003 to 2010) and that obtained from subjecting the actual weather variables of 2003 to 2005 to the Rain Forecast Model (RFM).

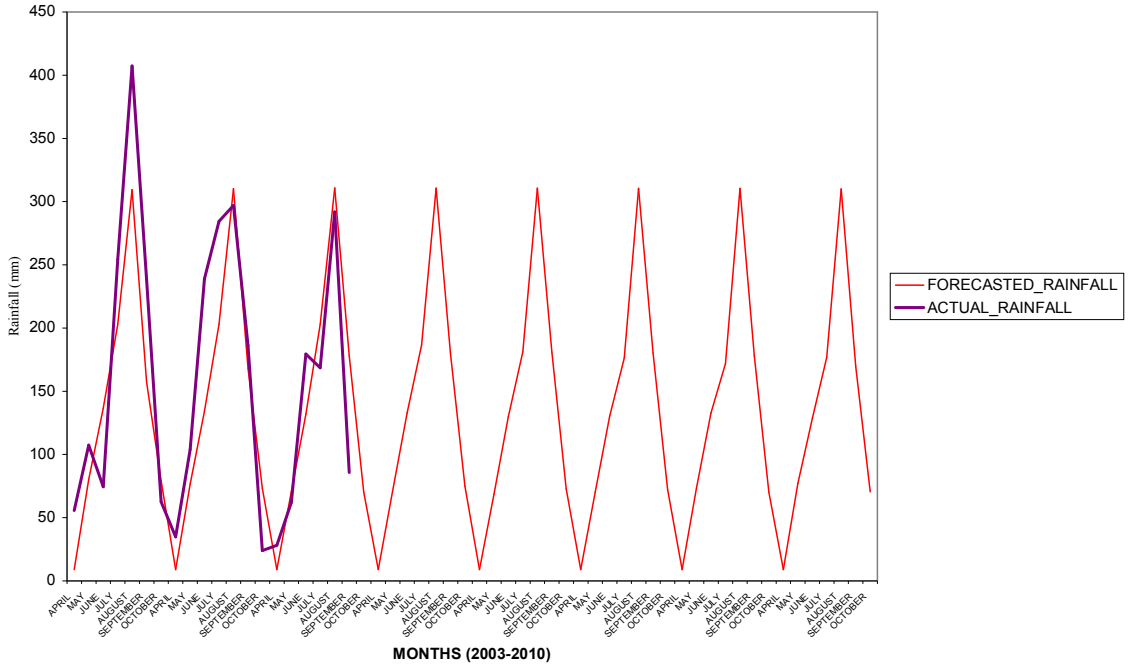


Figure 4.12: Plot of Actual (2003-2005) And Forecasted (2003-2010) Rainfall Data

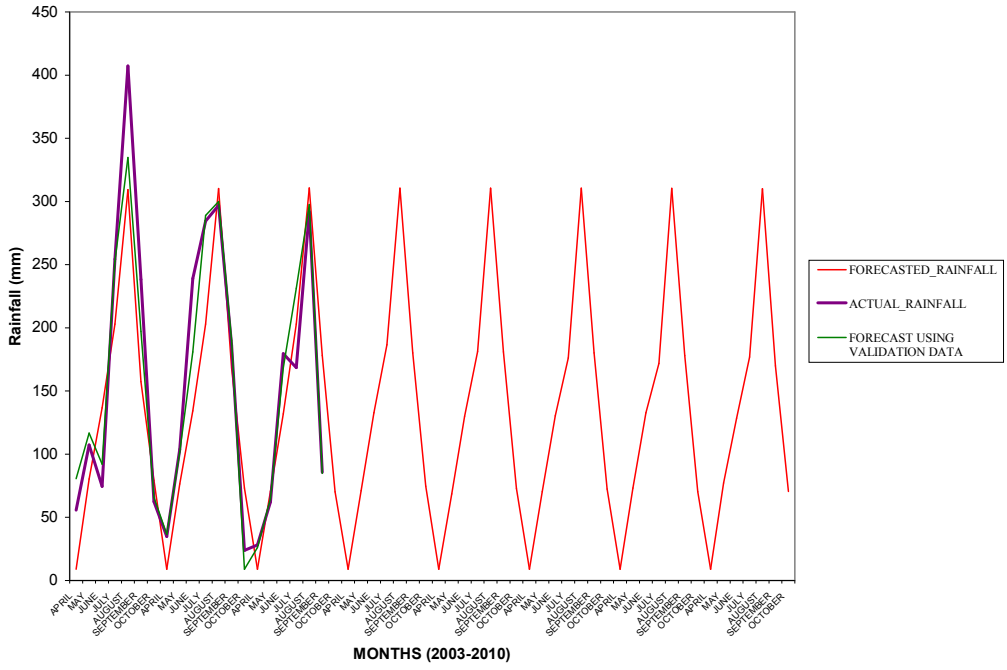


Figure 4.13: Plot of Actual (2003-2005), Validation (2003-2005) And Forecasted (2003-2010) Rainfall Data

CHAPTER FIVE

FUZZY TIME SERIES FORECASTING

5.1 INTRODUCTION

Fuzzy Set Theory and Fuzzy Logic were introduced as a means of representing, manipulating, and utilizing data and information that possess non-statistical uncertainty (also known as lexical or linguistic uncertainty [78]. The motivation, therefore, for fuzzy set theory is mainly to provide a formal, powerful and quantitative framework to cope with the vagueness of human knowledge as it is expressed by means of natural languages [24].

A fuzzy set A in the universe of discourse U is characterized by a membership function (MBF):

$$\mu_A : U \to [0,1] \qquad 5.1$$

This associates with each element u of U a number μ(u) in the interval [0, 1] that represents the grade of membership of u in A. The fuzzy set A of $U = u_1, u_2, \ldots, u_n$ will be denoted by [78][86]:

$$A = \sum_{i=1}^{n} \mu(u_i)/u_i \qquad 5.2$$

The construction of a fuzzy set, therefore, depends on two things:

i) The identification of a suitable universe of discourse.

ii) The specification of an appropriate membership function (MBF).

The specification of the membership function (MBF) may vary, for the same concept, from person to person but they cannot be assigned arbitrarily.

5.2 FUZZY TIME SERIES

According to **Abbasov et al [23]**, time series represents a consecutive series of observations taken over equal time intervals and lies at the root of exploring such real processes as in economics, meteorology, demography and natural sciences among others. The application of Fuzzy Logic and fuzzy sets to time series analysis gave rise to Fuzzy Time Series. The method to be applied here is the method adopted by **Abbasov et al [23]**, in forecasting population in Azerbaijan.

Similar fuzzy time series technique has been used by **Lee et al [24]** to forecast the unemployment rates in Taipei and by **Chen et al [87]** to forecast enrollments of the University of Alabama. **Wu [88]** also used the fuzzy time series technique to develop a web-based fuzzy forecasting system to model wind characteristics in Taiwan. This technique can, therefore, be used for forecasting where time series data exist such as population growth, statutory financial allocation to States and/or Local Governments, student enrollment into Universities, telephone growth, and electric power demand among others. These issues are very important for planning in developing countries like Nigeria.

Let U be the universe of discourse, where $U = \{u_1, u_2, ..., u_n\}$ and let A be a fuzzy set in the universe of discourse defined as:

$$A = f_A(u_1)/u_1 + f_A(u_2)/u_2 + ... + f_A(u_n)/u_n \qquad 5.3$$

Where: f_A is the membership function of A, $f_A : U \to [0,1]$,

$f_A(u_i)$ indicates the grade of membership of u_i in the fuzzy set A, $f_A(u_i) \in [0,1]$ and $1 \le i \le n$.

Let $X(t)(t = ...,0,1,2,...)$ be the universe of discourse and be a subset of R, and let fuzzy set $f_i(t)(i = 1,2,...)$ be defined in X(t). Let F(t) be a collection of $f_i(t)(i = 1,2,...)$. Then, F(t) is called a fuzzy time series of $X(t)(t = ...,0,1,2,...)$.

If $F(t)$ is caused by $F(t-1)$, denoted by $F(t-1) \to F(t)$, then this relationship can be represented by:

$$F(t) = F(t-1) \circ R(t,t-1) \qquad 5.4$$

Where: The symbol "$\circ$" denotes the Max-Min composition operator;

$R(t,t-1)$ is a fuzzy relation between $F(t)$ and $F(t-1)$ and is called **the first-order model** of $F(t)$.

Let $F(t)$ be a fuzzy time series and let $R(t,t-1)$ be a first-order model of $F(t)$. If $R(t,t-1) = R(t-1,t-2)$ for any time t, then $F(t)$ is called **time-invariant fuzzy time series**. If $R(t,t-1)$ is dependent on time t, that is $R(t,t-1)$ may be different from $R(t-1,t-2)$ for any time t, then $F(t)$ is called a **time-variant fuzzy time series** [20][26].

5.3 FORECASTING METHODOLOGY

The fuzzy time series technique will be applied to the historical rainfall data on a monthly basis. The methodology will be described with the historical rainfall data of the month of

September, as an example. The data required to forecast for the other months are listed in Appendix A7.

The development of the forecast model using the fuzzy time series technique to forecast for the rainfall involves the following steps:

i) **Definition of the universal set U** containing interval between the minimum and maximum variations, as obtained from Table 5.1. Variation for, say September 1996, is the difference between the rainfall value of September 1996 and that of September 1995.

$$U = [V_{\min}, V_{\max}] \qquad 5.5$$

$$U = [-112.5, 134.9]$$

ii) Division of the universal set U into several equal-length intervals containing the variation values. There is no empirical method of determining the size of the interval as such it is more predicated on the size of the universal set, **U. Abbasov et al [23]** used seven equal-length intervals whilst **Lee et al [24]** used five equal-length intervals in their respective works**.** In this work, the interval is partitioned into five equal-length intervals as follows:

$$u_1 = [-112.5, -63.02]$$
$$u_2 = [-63.02, -13.54]$$
$$u_3 = [-13.54, 35.94]$$
$$u_4 = [35.94, 85.52]$$
$$u_5 = [85.42, 134.9]$$

The following linguistic variables are defined in order to describe the variations of the monthly rainfall amounts:

- Significant Decrease (SD)
- Moderate Decrease (MD)
- Minimal Change (MC)
- Moderate Increase (MI)
- Significant Increase (SI)

In order to account for the fact that forecasting with fuzzy time series exhibits the least average error [26], it is necessary to find the middle points of the intervals:

$$u_m^1 = -87.76$$
$$u_m^2 = -38.28$$
$$u_m^3 = 11.2$$
$$u_m^4 = 60.68$$
$$u_m^5 = 110.16$$

iii) **Determining the fuzzy sets**, that is, the qualitative description of the variation values as a linguistic variable.

$U = Variation\ in\ Ra\inf\ all\ Amount$ is a linguistic variable that assumes the following linguistic value:

$$\begin{aligned} A_1 &= Significant\ Decrease\ in\ Ra\inf all\ Amount \\ A_2 &= Moderate\ Decrease\ in\ Ra\inf all\ Amount \\ A_3 &= Minimal\ Change\ in\ Ra\inf all\ Amount \\ A_4 &= Moderate\ Increase\ in\ Ra\inf all\ Amount \\ A_5 &= Significant\ Increase\ in\ Ra\inf all\ Amount \end{aligned} \qquad 5.6$$

The fuzzy sets A1, … , A5 are defined on U by the following relationship, according to **Abbasov et al [23]**:

$$\mu_{A_i}(u_i) = \frac{1}{1 + \left[C\left(U - u_m^i\right)\right]^2} \qquad 5.7$$

Where: U = Variation (In the case of defining the fuzzy sets A_1, … , A_5, this is the middle point of the interval)

u_m^i = Middle point of the corresponding interval

C = Constant. There is no empirical method of determining this constant but according to **Abbasov et al [23],** it must be chosen in such a way that it ensures the conversion of definite quantitative values into fuzzy values. A value of **C = 0.01** is chosen for this work.

Consequently,

$$\begin{aligned} A_1 &= 1/u_1 + 0.80/u_2 + 0.51/u_3 + 0.31/u_4 + 0.20/u_5 \\ A_2 &= 0.80/u_1 + 1/u_2 + 0.80/u_3 + 0.51/u_4 + 0.31/u_5 \\ A_3 &= 0.51/u_1 + 0.80/u_2 + 1/u_3 + 0.80/u_4 + 0.51/u_5 \\ A_4 &= 0.31/u_1 + 0.51/u_2 + 0.80/u_3 + 1/u_4 + 0.80/u_5 \\ A_5 &= 0.20/u_1 + 0.31/u_2 + 0.51/u_3 + 0.80/u_4 + 1/u_5 \end{aligned} \qquad 5.8$$

The continuous membership functions of fuzzy sets A_i depicting the values of the linguistic values of the variation of rainfall amount are shown in Figure 5.1

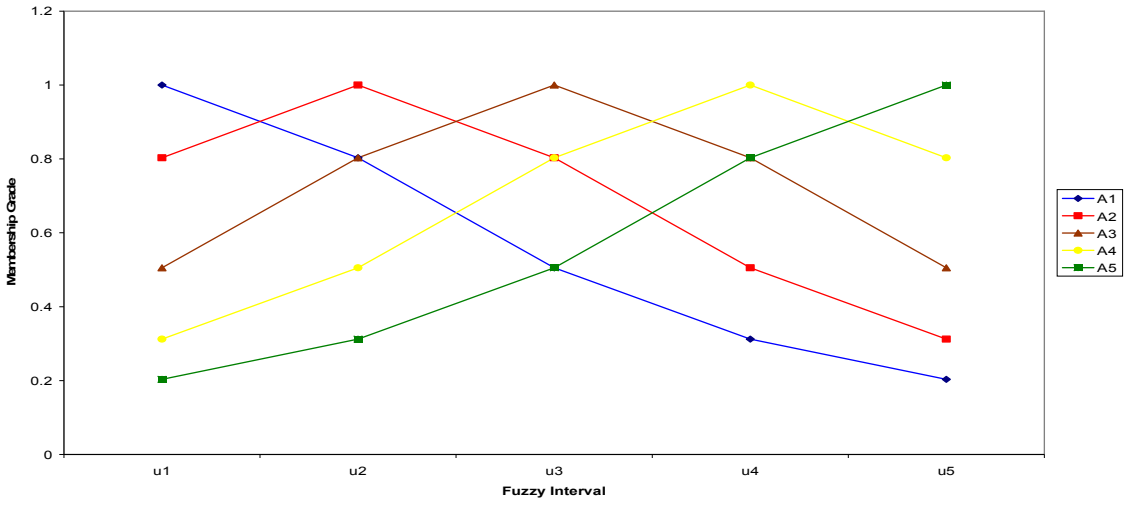


Figure 5.1: Membership Function (MBF) of Set of Fuzzy Intervals

i) **Fuzzifying the input data** that is, converting numerical values into fuzzy values. This is also achieved using equation 5.7 and is as shown in Table 5.1.

Table 5.1: Variation and Fuzzification of Variations

September	RAINFALL	VARIATION		CONSTANT(k)	u1	u2	u3	u4	u5
1993	199.7								
1994	203.1	3.4	A3	0.01	0.55	0.85	0.99	0.75	0.47
1995	112	-91.1	A1	0.01	1.00	0.78	0.49	0.30	0.20
1996	173.5	61.5	A4	0.01	0.31	0.50	0.80	1.00	0.81
1997	192.3	18.8	A3	0.01	0.47	0.75	0.99	0.85	0.55
1998	236.8	44.5	A4	0.01	0.36	0.59	0.90	0.97	0.70
1999	268.2	31.4	A3	0.01	0.41	0.67	0.96	0.92	0.62
2000	177.7	-90.5	A1	0.01	1.00	0.79	0.49	0.30	0.20
2001	312.6	134.9	A5	0.01	0.17	0.25	0.40	0.64	0.94
2002	200.1	-112.5	A1	0.01	0.94	0.64	0.40	0.25	0.17

ii) Selection of parameter (w > 1) corresponding to the time period prior to the concerned month and calculation of the relationships matrix. It is based on the choice of **w** (the model basis) that the fuzzy relationship matrix $\mathbf{R^w(t)}$ is calculated and from which the forecast is made.

The choice of **w** is very critical to the fuzzy time series analysis and as such a sensitivity analysis is carried out for the fuzzy times series model with varying values of **w** in order to determine which one gives the least forecasting error. This is carried out using values of **w** = 4, 5 and 6 and the outcomes are as shown in Table 5.2.

Table 5.2: Forecasting Errors With Various Values of **w**

		w=4		w=5		w=6	
SEPTEMBER	**ACTUAL RAINFALL**	F_RAIN	**% ERROR**	F_RAIN	**% ERROR**	F_RAIN	**% ERROR**
2003	**238.2**	171.5	**28.00**	171.5	**28.00**	171.5	**28.00**
2004	**186.3**	165.3	**11.27**	167.2	**10.25**	167.2	**10.25**
2005	**85.7**	167.4	**-95.33**	171.2	**-99.77**	171.2	**-99.77**

The percentage errors for **w=5** and for **w=6** are similar considering the procedure as subsequently explained. On the basis of Table 5.2, the model basis, **w = 6** was selected. The relationship matrix $\mathbf{R^w(t)}$ is then determined from the expression:

$$R^w(t) = O^w(t) \otimes C^w(t) \qquad 5.9$$

Where $O^w(t)$ is the operation matrix (i x j), $C^w(t)$ is the criterion matrix (1 x j) and $\otimes$ is the operation Min $(\cap)$.

As an example, for September 2003, a 5 x 5 operation matrix will be formed as follows:

$$O^6(sep2003) = \begin{vmatrix} fuzzy\ \text{var}iation\ for\ september\ 1997 \\ fuzzy\ \text{var}iation\ for\ september\ 1998 \\ fuzzy\ \text{var}iation\ for\ september\ 1999 \\ fuzzy\ \text{var}iation\ for\ september\ 2000 \\ fuzzy\ \text{var}iation\ for\ september\ 2001 \end{vmatrix} \qquad 5.10$$

$$O^6(sep2003) = \begin{vmatrix} 0.47 & 0.75 & 0.99 & 0.85 & 0.55 \\ 0.36 & 0.59 & 0.90 & 0.97 & 0.70 \\ 0.41 & 0.67 & 0.96 & 0.92 & 0.62 \\ 1.00 & 0.79 & 0.49 & 0.30 & 0.20 \\ 0.17 & 0.25 & 0.40 & 0.64 & 0.94 \end{vmatrix}$$

A 1 x 5 criterion matrix will be formed as follows:

$$C^6(sep2003) = \left| fuzzy\ \text{var}iation\ for\ september\ 2002 \right| \qquad 5.11$$

$$C^6(sep2003) = \left| 0.94 \quad 0.64 \quad 0.40 \quad 0.25 \quad 0.17 \right|$$

The relationship matrix, $\mathbf{R^w(t)}$ is then determined from: equation 5.9 resulting in:

$$R^{w}(sep2003) = \begin{vmatrix} R_{11} & R_{12} & R_{13} & R_{14} & R_{15} \\ R_{21} & R_{22} & R_{23} & R_{24} & R_{25} \\ R_{31} & R_{32} & R_{33} & R_{34} & R_{35} \\ R_{41} & R_{42} & R_{43} & R_{44} & R_{45} \\ R_{51} & R_{52} & R_{53} & R_{54} & R_{55} \end{vmatrix} \qquad 5.12$$

$$= \begin{vmatrix} 0.47 & 0.64 & 0.40 & 0.25 & 0.17 \\ 0.36 & 0.59 & 0.40 & 0.25 & 0.17 \\ 0.41 & 0.64 & 0.40 & 0.25 & 0.17 \\ 0.94 & 0.64 & 0.40 & 0.25 & 0.17 \\ 0.17 & 0.25 & 0.40 & 0.25 & 0.17 \end{vmatrix}$$

The final fuzzified predicted variation **$F^{w}(t)$** is given as:

$$F^{6}(sep2003) = \begin{vmatrix} Max(R_{11}, R_{21}, R_{31}, R_{41}, R_{51}) \\ Max(R_{12}, R_{22}, R_{32}, R_{42}, R_{52}) \\ Max(R_{13}, R_{23}, R_{33}, R_{43}, R_{53}) \\ Max(R_{14}, R_{24}, R_{34}, R_{44}, R_{45}) \\ Max(R_{15}, R_{25}, R_{35}, R_{45}, R_{55}) \end{vmatrix}^{T} \qquad 5.13$$

$$F^{6}(sep2003) = \begin{vmatrix} 0.94 & 0.64 & 0.40 & 0.25 & 0.17 \end{vmatrix}$$

iii) **Defuzzifying obtained results, that is, conversion into quantitative values.**

In order to achieve this, **Abbasov et al [23]** proposed the following formula:

$$V(t) = \frac{\sum_{i=1}^{5} \mu_t(u_i) \bullet u_m^i}{\sum_{i=1}^{5} \mu_t(u_i)} \qquad 5.14$$

Where V(t) is the expected variation for the forecast year t, $\mu_t(u_i)$ is the calculated membership function (MBF) for the forecast year t and u^i_m are the middle points of the intervals (the size of the intervals is 5).

As an illustration, for September 2003,

$$V(sep2003) = \frac{(0.94 \times -87.76) + (0.64 \times -38.28) + (0.40 \times 11.2) + (0.25 \times 60.68) + (0.17 \times 110.16)}{0.94 + 0.64 + 0.40 + 0.25 + 0.17}$$

$$V(sep2003) = \frac{-68.6}{2.4} = -28.6$$

This implies that there will be a moderate decrease in rainfall by the value (V(sep2003)) from the amount of rainfall in September 2002. Therefore:

$$R(sep2003) = R(sep2002) + V(sep2003) = 200.1 - 28.6 = 171.5$$

Similarly, rainfall values for September 2004 to September 2010 are determined and the results are as in Table 5.3

Table 5.3: Forecasted Rainfall For September 2003 to 2010

SEPTEMBER	RAINFALL	VARIATION		CONSTANT (C)	u1	u2	u3	u4	u5
1993	199.7								
1994	203.1	3.4	A3	0.01	0.55	0.85	0.99	0.75	0.47
1995	112	-91.1	A1	0.01	1.00	0.78	0.49	0.30	0.20
1996	173.5	61.5	A2	0.01	0.31	0.50	0.80	1.00	0.81
1997	192.3	18.8	A3	0.01	0.47	0.75	0.99	0.85	0.55
1998	236.8	44.5	A4	0.01	0.36	0.59	0.90	0.97	0.70
1999	268.2	31.4	A3	0.01	0.41	0.67	0.96	0.92	0.62
2000	177.7	-90.5	A1	0.01	1.00	0.79	0.49	0.30	0.20
2001	312.6	134.9	A5	0.01	0.17	0.25	0.40	0.64	0.94
2002	200.1	-112.5	A1	0.01	0.94	0.64	0.40	0.25	0.17
2003	**171.5**	**-28.6**	**A2**	**0.01**	**0.74**	**0.99**	**0.86**	**0.56**	**0.34**
2004	**167.2**	**-4.3**	**A3**	**0.01**	**0.59**	**0.90**	**0.98**	**0.70**	**0.43**
2005	**171.2**	**4**	**A3**	**0.01**	**0.54**	**0.85**	**0.99**	**0.76**	**0.47**
2006	**178.4**	**7.2**	**A3**	**0.01**	**0.53**	**0.83**	**1.00**	**0.78**	**0.49**
2007	**187.5**	**9.1**	**A3**	**0.01**	**0.52**	**0.82**	**1.00**	**0.79**	**0.49**
2008	**197.3**	**9.8**	**A3**	**0.01**	**0.51**	**0.81**	**1.00**	**0.79**	**0.50**
2009	**207.7**	**10.4**	**A3**	**0.01**	**0.51**	**0.81**	**1.00**	**0.80**	**0.50**
2010	**218.1**	**10.4**	**A3**	**0.01**	**0.51**	**0.81**	**1.00**	**0.80**	**0.50**

The same approach is used to determine the forecasted rainfall for the other months and the plot of the forecasted rainfall using the fuzzy time series technique and actual rainfall for the forecast period is shown in Figure 5.2 while that of the forecasted rainfall using

the fuzzy time series technique and that using the Rain Forecast Model (RFM) is shown in Figure 5.3. It can be seen that the forecast plot using the Rain Forecast Model (RFM) is closer to the actual rainfall plot than that using the fuzzy time series method.

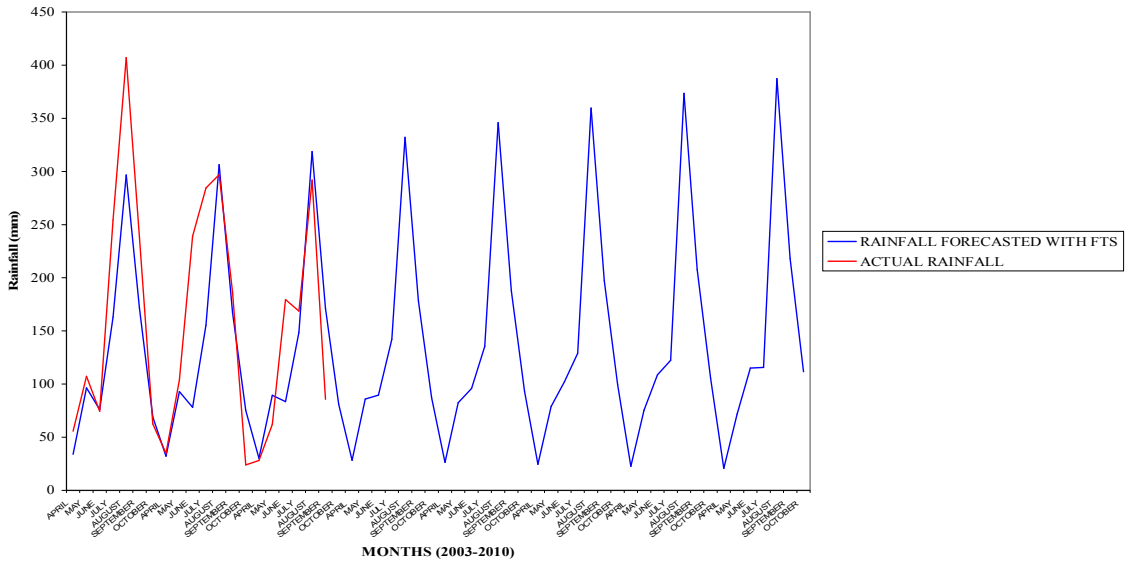


Figure 5.2: Plot of Actual Rainfall And Forecasted Rainfall Using Fuzzy Time Series

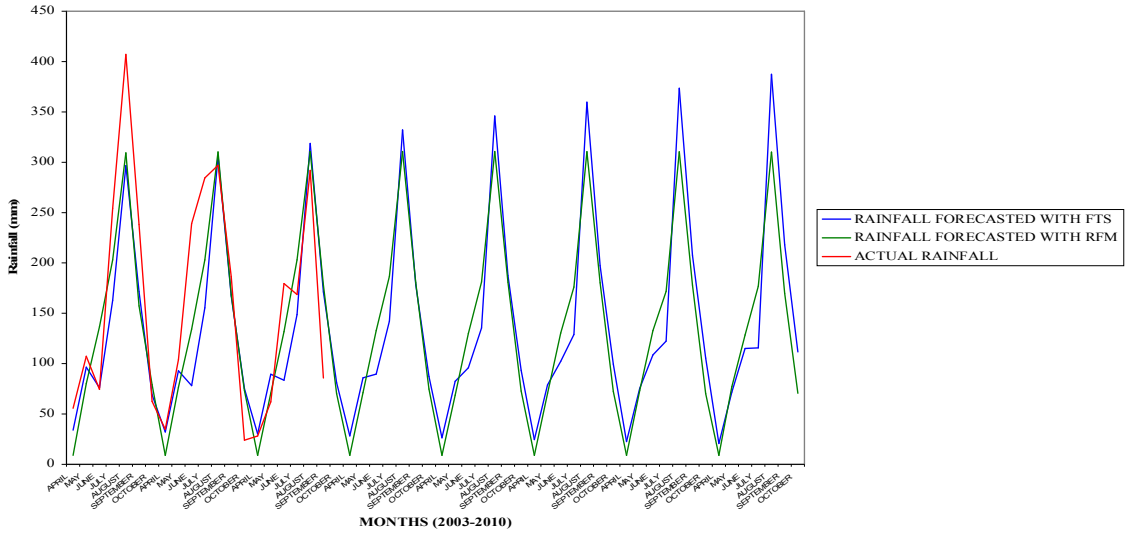


Figure 5.3: Plot of Actual Rainfall, Forecasted Rainfall Using Fuzzy Time And Forecasted Rainfall Using The Rain Forecast Model (RFM)

CHAPTER SIX

RESULTS AND ANALYSIS

6.1 INTRODUCTION

This investigation involves the development of a Neuro-Fuzzy based Rain Forecast Model (RFM) using test data consisting of weather variables Relative Humidity, Wind Direction and Wind Speed from 1993 to 2002 and validation data from 2003 to 2005. The choice of the weather variables used in this work is at the discretion of the author and not necessarily based on any existing work. In order to forecast into the future, the weather variables are forecasted using a hybrid statistical method consisting of moving averages and exponential smoothing. This hybrid method has been adopted in this work based on the fact that the respective methods of Moving Averages and Exponential Smoothing can be used in time series forecasting. The forecasted weather variables are then used to forecast the rainfall using the Rain Forecast Model (RFM). A fuzzy time series technique is also used to forecast the rainfall in order to compare with results obtained from the Rain Forecast Model (RFM). The results obtained are analyzed in this chapter. The analysis consists of forecast plots and statistical error measures in the estimation period like the Root Mean Square Error (RMSE) and the Correlation Factor [89].

Forecast plots are qualitative indicators of the performance of a model. The performance of each developed model is presented in the form of a plot which allows for visual comparison of the models.

The error measure in the validation period is very important as the model's performance in the validation period is a vital guide to its ability to predict the future. It must, however, be noted that the validation period is usually a much smaller sample of data than the estimation period. It is always possible to get good performance on a training set, but the important thing is to have the model perform well on new data, especially the validation data as well as in the forecast period. Therefore, the error measure in the forecast period will indicate how well the model could predict the future by comparing it with the results obtained in the validation period.

6.2 RAIN FORECAST MODEL (RFM)

The Rain Forecast Model (RFM), as in Figure 6.1, is developed using the Fuzzy Technology Language (FTL) as the pure Fuzzy Logic implementation of the Neuro-Fuzzy System developed using the fuzzyTECH 5.5h Neuro-Fuzzy module (described in Section 3:1).The Neuro-Fuzzy System developed consists of fifty-seven rules as in Appendix A3.

The Rain Forecast Model is used to forecast for the rainfall with the previously unseen data, which is the validation data of 2003 to 2005, using the interactive debug Watch Window. The Watch Windows of Figure's 3.25 – 3.27 of Section 3:2 show the outcomes for August 2003, July 2004 and September 2005 as examples.

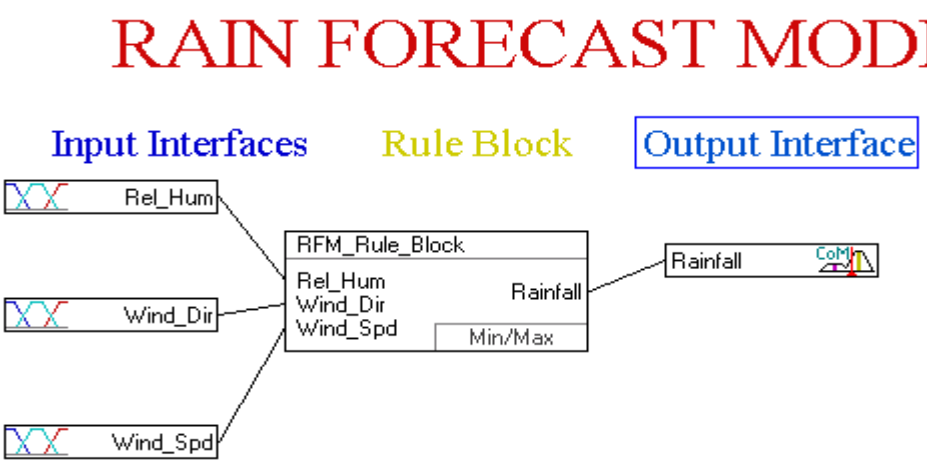


Figure 6.1: Structure of the Rain Forecast Model Fuzzy Logic System

The plot of the actual rainfall data for the validation period and that forecasted using the Rain Forecast Model (RFM) is shown in Figure 6.2. The Root Mean Square Error (RMSE) between the actual rainfall and forecasted rainfall generated by the Rain Forecast Model (RFM) is used as the performance function and the outcome is as tabulated in Table 6.1. The Root Mean Square Error (RMSE) is determined from:

$$RMSE = \sqrt{MSE} \qquad 6.1$$

Where,

$$MSE = \frac{\sum_{t=1}^{n} \left(Actual_Rainfall_t - Predicted_Rainfall_t \right)^2}{n} \qquad 6.2$$

In addition to this, the Pearson's correlation factor, R, between the actual rainfall and forecasted rainfall generated by the Rain Forecast Model (RFM) has also been determined. The correlation coefficient, a number between 0 and 1, is a measure of how well trends in the validation values using the developed Rain Forecast Model (RFM) follow trends in the actual data. What constitutes a good value of R is dependent on the problem domain.

In this case, R > 0.8 is an indication of a fairly good model considering the volatility of weather variables in general and rainfall in particular.

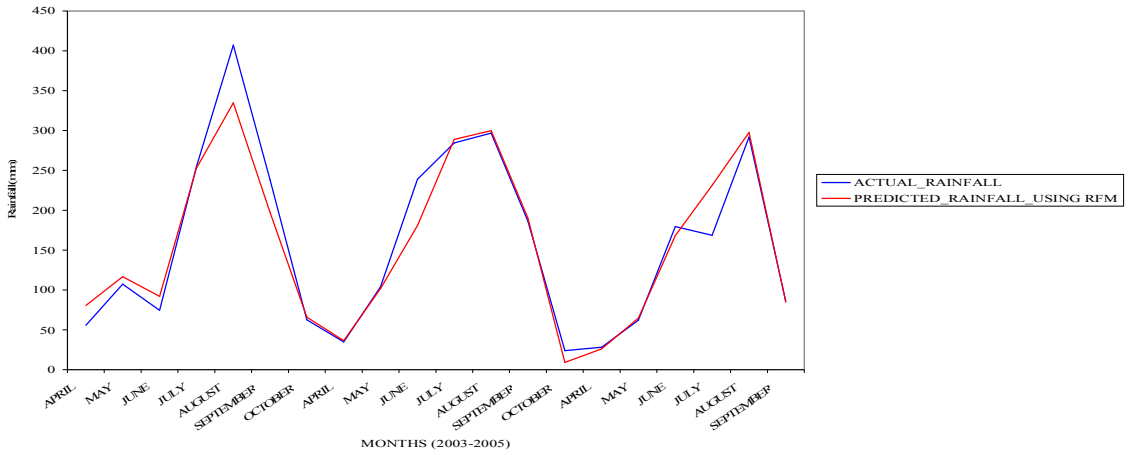


Figure 6.2: Plot of Actual Rainfall And Forecasted Rainfall Using Validation Data of 2003 – 2005

Table 6.1: Root Mean Square Error (RMSE) Between Actual And Forecasted Rainfall

MONTHS	REL_HUM	WIND_DIR	WIND_SPD	ACTUAL_RAINFALL	PREDICTED_RAINFALL _USING RFM	SE
APRIL	72.2	56	192.8	55.7	**80.6**	**620.0**
MAY	71.1	68	185.02	107.3	**116.6**	**86.5**
JUNE	80.8	69	183.5	74.5	**92.025**	**307.1**
JULY	81.3	59	177.11	254.3	**252.6**	**2.9**
AUGUST	83.9	59	141.29	407.2	**334.8**	**5241.8**
SEPTEMBER	79.7	40	122.2	238.2	**197.2**	**1681.0**
OCTOBER	71.1	23	103.37	62.5	**65.9**	**11.6**
APRIL	52	42	206.48	34.8	**36.5**	**2.9**
MAY	66.3	89	234.22	103.9	**101.7**	**4.8**
JUNE	78	68	207.3	239	**180.9**	**3375.6**
JULY	78	60	165.96	284.4	**288.99**	**21.1**
AUGUST	84	53	168.14	296.9	**299.9**	**9.0**
SEPTEMBER	74.5	44	130.21	186.3	**190.2**	**15.2**
OCTOBER	55	17	110.78	23.9	**9**	**222.0**
APRIL	50.5	43	212.6	28.1	**26.05**	**4.2**
MAY	68	50	197.9	62.1	**64.6**	**6.2**
JUNE	78.4	59	198.6	179.5	**168.3**	**125.4**
JULY	83	64	176.6	168.6	**231.3**	**3931.3**
AUGUST	85.3	66	170.9	291.9	**297.6**	**32.5**
SEPTEMBER	74.1	41	125	85.7	**84.8**	**0.8**
						MSE=785.1
						RMSE=28.02
						R=0.97

6.3 FORECASTING RAINFALL USING FORECASTED WEATHER VARIABLES

The developed Rain Forecast Model (RFM) uses as its inputs the weather variables: Relative Humidity, Wind Direction and Wind Speed. A hybrid statistical approach using the time series techniques of Moving Averages and Exponential Smoothing (described in Section's 4:2 and 4:3) was adopted in predicting the weather variables for the period 2003 to 2010. The forecasted values of the weather variables are shown in Appendix A5.

The complete forecasted rainfall data obtained from the Rain Forecast Model (RFM) using the forecasted weather variables is shown in Appendix A6. A plot of the rainfall data, forecasted (2003 to 2010) and actual (2003 to 2005) is shown in Figure 6.3.

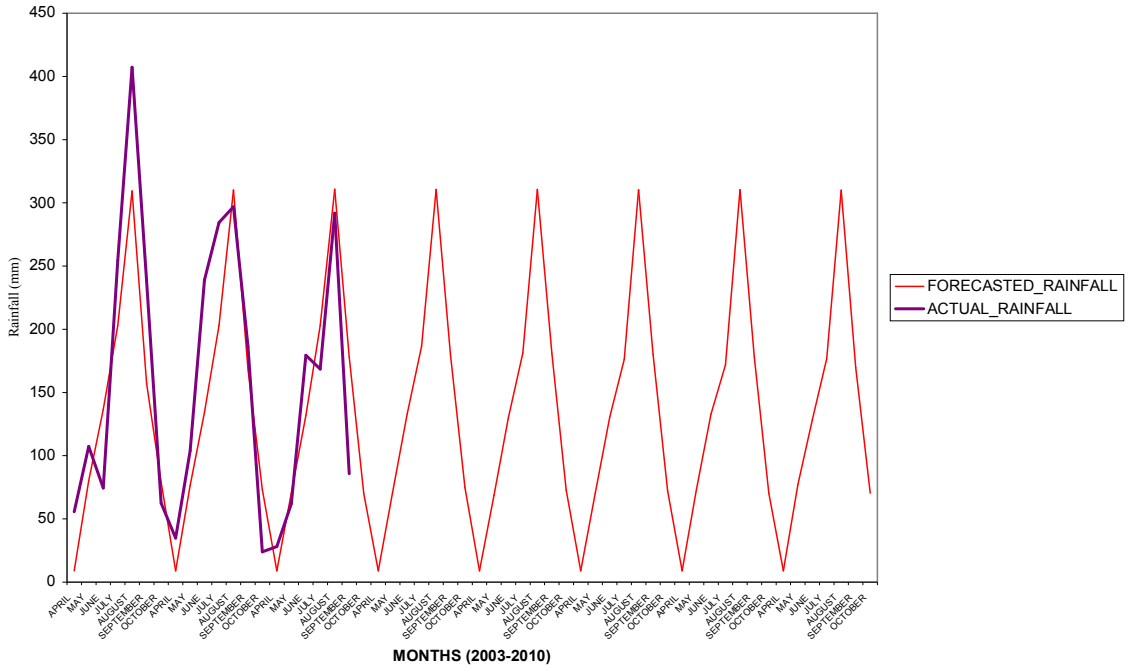


Figure 6.3: Plot of Actual (2003-2005) And Forecasted (2003-2010) Rainfall Data

The Root Mean Square Error (RMSE) between the actual rainfall and forecasted rainfall generated by the Rain Forecast Model (RFM) based on the forecasted weather variables is used as the performance function and the outcome is as tabulated in Table 6.2 In this case also, the model can only be tested for the validation period (2003 to 2005). In addition to this, the Pearson's correlation factor, R, between the actual rainfall and forecasted rainfall generated by the Rain Forecast Model (RFM) has also been determined. The correlation factor, here, is a measure of how well trends in the forecasted values using the developed Rain Forecast Model (RFM) follow trends in the actual data. The Root Mean Square Error (RMSE) is **54.99** while the correlation factor, R is **0.88**. This is a good indication of the efficiency of the model

Table 6.2: Root Mean Square Error (RMSE) Between Actual And Forecasted Rainfall Using Forecasted Weather Variables

		ACTUAL_RAINFALL	FORECASTED_RAINFALL	SE
2003	APRIL	55.7	9	2180.89
	MAY	107.3	80.3	729
	JUNE	74.5	136.6	3856.41
	JULY	254.3	203	2631.69
	AUGUST	407.2	309.4	9564.84
	SEPTEMBER	238.2	157.3	6544.81
	OCTOBER	62.5	79.2	278.89
2004	APRIL	34.8	9	665.64
	MAY	103.9	76.5	750.76
	JUNE	239	134.3	10962.09
	JULY	284.4	203	6625.96
	AUGUST	296.9	310.2	176.89
	SEPTEMBER	186.3	168.3	324
	OCTOBER	23.9	73.2	2430.49
2005	APRIL	28.1	9	364.81
	MAY	62.1	70.7	73.96
	JUNE	179.5	131.8	2275.29
	JULY	168.6	203	1183.36
	AUGUST	291.9	310.7	353.44
	SEPTEMBER	85.7	178	8519.29
				MSE=3024.6255
				RMSE=54.99
				R=0.88

6:4 FORECASTING RAINFALL USING FUZZY TIME SERIES

The application of Fuzzy Logic and fuzzy sets to time series analysis gave rise to Fuzzy Time Series. This means that fuzzy time series methods can be applied where time series data exists. The method applied in this work, however, is the one adopted by **Abbasov et al [23]**, in forecasting population in Azerbaijan. It is used to forecast for rainfall, as described in Section 5:3, in order to compare the results computed with the results obtained from the Rain Forecast Model (RFM).

A plot of the rainfall data, forecasted (2003 to 2010) using the fuzzy time series method and actual (2003 to 2005) is shown in Figure 6.4

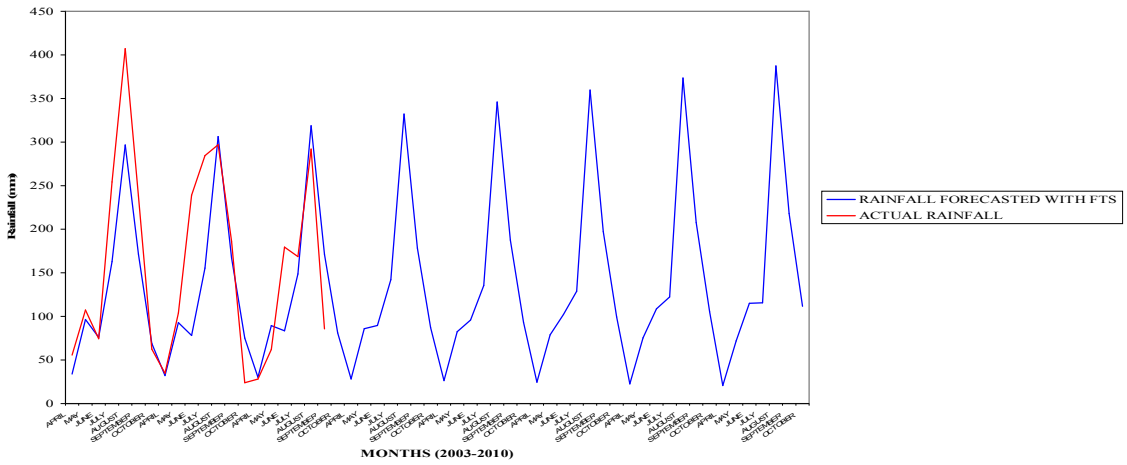


Figure 6.4: Plot of Actual Rainfall And Forecasted Rainfall Using Fuzzy Time Series

The Root Mean Square Error (RMSE) between the actual rainfall and forecasted rainfall using the fuzzy time series method is used as the performance function and the outcome is as tabulated in Table 6.3 In this case also, the model can only be tested for the validation period (2003 to 2005) in order to have a basis for comparison with the actual

data.. In addition to this, the Pearson's correlation factor, R, between the actual data rainfall and forecasted rainfall obtained from using the fuzzy time series has also been determined. The correlation factor, here, is a measure of how well trends in the forecasted values using the adopted fuzzy time series method follow trends in the actual data. The Root Mean Square Error (RMSE) is **66.67** while the correlation factor, R, is **0.82**. Although both measures are good, they are, however, inferior to the measures obtained for the **Rain Forecast Model (RFM).**

Table 6.3: Root Mean Square Error (RMSE) Between Actual And Forecasted Rainfall Using Fuzzy Time Series

MONTHS	ACTUAL RAINFALL	RAINFALL FORECASTED WITH FTS	SE
APRIL	55.7	33.9	475.24
MAY	107.3	96.4	118.81
JUNE	74.5	76.1	2.56
JULY	254.3	162.9	8353.96
AUGUST	407.2	296.7	12210.25
SEPTEMBER	238.2	171.5	4448.89
OCTOBER	62.5	69	42.25
APRIL	34.8	32.1	7.29
MAY	103.9	92.9	121
JUNE	239	78.2	25856.64
JULY	284.4	155.4	16641
AUGUST	296.9	306.3	88.36
SEPTEMBER	186.3	167.2	364.81
OCTOBER	23.9	75.1	2621.44
APRIL	28.1	30.2	4.41
MAY	62.1	89.4	745.29
JUNE	179.5	83.5	9216
JULY	168.6	148.7	396.01
AUGUST	291.9	318.7	718.24
SEPTEMBER	85.7	171.2	7310.25
			MSE=4487.135
			RMSE=66.97
			R=0.82

6:5 RAINFALL FORECAST RESULTS

The plot of the rainfall forecast results obtained from using the Rain Forecast Model (RFM) and that obtained from using fuzzy time series is as shown in Figure 6.5. The forecast period is from 2003 to 2010.

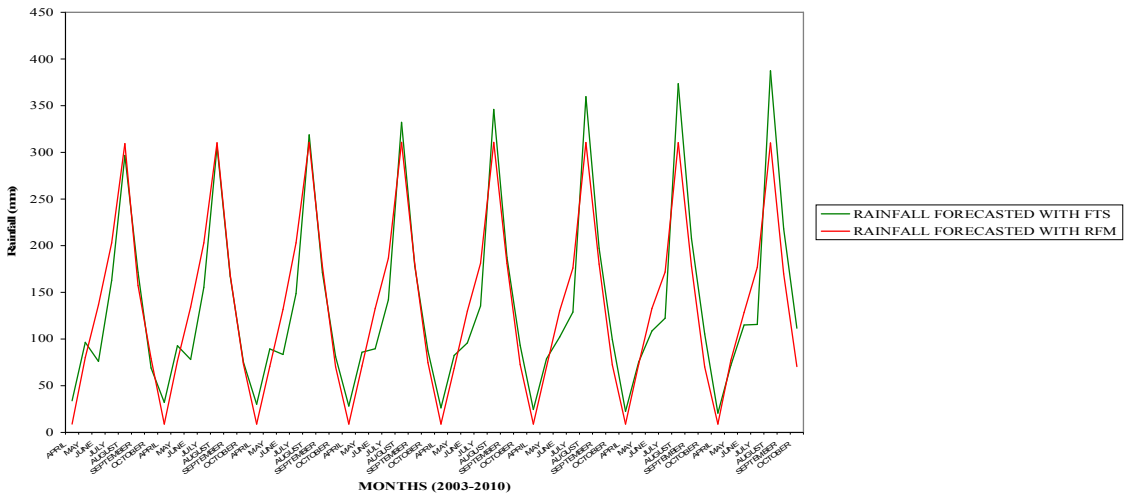


Figure 6.5: Plot of Forecasted Rainfall Using Rain Forecast Model (RFM) And Fuzzy Time Series

The Root Mean Square Error (RMSE) between the forecasted rainfall using the Rain Forecast Model (RFM) and that using the fuzzy time series method is used as the performance function and the outcome is as tabulated in Table 6.4. In addition to this, the Pearson's correlation factor, R, between these data has also been determined. The correlation coefficient is determined here as a measure of how well trends in the forecasted values using the developed Rain Forecast Model (RFM) follow the trends in the forecasted values using the adopted fuzzy time series method. In this case, it is safe to assume the fact that $R > 0.8$ is an indication of a fairly good correlation between the

results obtained using the Rain Forecast Model (RFM) and using the fuzzy time series method.

Table 6.4: Root Mean Square Error (RMSE) Between Forecasted Rainfall Using The Rain Forecast Model (RFM) And Forecasted Rainfall Using Fuzzy Time Series

MONTHS	RAINFALL FORECASTED WITH RFM	RAINFALL FORECASTED WITH FTS	SE
APRIL	9	33.9	620.01
MAY	80.3	96.4	259.21
JUNE	136.6	76.1	3660.3
JULY	203	162.9	1608
AUGUST	309.4	296.7	161.29
SEPTEMBER	157.3	171.5	201.64
OCTOBER	79.2	69	104.04
APRIL	9	32.1	533.61
MAY	76.5	92.9	268.96
JUNE	134.3	78.2	3147.2
JULY	203	155.4	2265.8
AUGUST	310.2	306.3	15.21
SEPTEMBER	168.3	167.2	1.21
OCTOBER	73.2	75.1	3.61
APRIL	9	30.2	449.44
MAY	70.7	89.4	349.69
JUNE	131.8	83.5	2332.9
JULY	203	148.7	2948.5
AUGUST	310.7	318.7	64
SEPTEMBER	178	171.2	46.24
OCTOBER	70.2	81.2	121
APRIL	9	28.3	372.49
MAY	71.1	85.9	219.04
JUNE	132.8	89.6	1866.2
JULY	186.6	142.1	1980.3
AUGUST	310.6	332.1	462.25
SEPTEMBER	180.2	178.4	3.24
OCTOBER	74.7	87.3	158.76
APRIL	9	26.4	302.76
MAY	68.6	82.4	190.44
JUNE	129.9	95.9	1156
JULY	181.2	135.5	2088.5
AUGUST	310.6	345.9	1246.1
SEPTEMBER	180.9	187.5	43.56
OCTOBER	72.8	93.4	424.36
APRIL	9	24.5	240.25
MAY	70.3	78.9	73.96
JUNE	130.4	102.3	789.61
JULY	176.1	128.9	2227.8

MONTHS	RAINFALL FORECASTED WITH RFM	RAINFALL FORECASTED WITH FTS	SE
AUGUST	**310.5**	**359.7**	2420.6
SEPTEMBER	**180.2**	**197.3**	292.41
OCTOBER	**72.4**	**99.5**	734.41
APRIL	**9**	**22.6**	184.96
MAY	**73.2**	**75.4**	4.84
JUNE	**132.8**	**108.7**	580.81
JULY	**171.8**	**122.3**	2450.3
AUGUST	**310.4**	**373.5**	3981.6
SEPTEMBER	**178**	**207.7**	882.09
OCTOBER	**70.3**	**105.6**	1246.1
APRIL	**9**	**20.7**	136.89
MAY	**77.2**	**71.9**	28.09
JUNE	**128.5**	**115.1**	179.56
JULY	**176.8**	**115.7**	3733.2
AUGUST	**310.1**	**387.3**	5959.8
SEPTEMBER	**169.8**	**218.1**	2332.9
OCTOBER	**70.5**	**111.7**	1697.4
		MSE=	**1068.81**
		RMSE	**32.69**
		R=	**0.94**

6.6 SIGNIFICANCE OF RESULTS OBTAINED FROM INVESTIGATION

Weather forecast in general and rainfall forecast in particular, especially on the long term, is a difficult task considering its extreme variability. A number of possible interactions within the atmosphere and between the atmosphere and other influences make it impossible to make long-range forecasts with the same degree of accuracy as short-range forecasts. The root of this problem is referred to as **Chaos**, also called the "**butterfly effect**" [90].

The objective of this investigation was to develop a data driven method using the Soft Computing technique of Neuro-Fuzzy to forecast the rainfall using the following weather variables: Relative Humidity, Wind Direction and Wind Speed. The Neuro-Fuzzy System developed was implemented as a pure Fuzzy Logic System as the Rain Forecast Model

(RFM). The fuzzy time series method proposed by **Abbasov et al [23]** was used to predict for the rainfall in order to have a basis for comparison.

Forecast verification is the process of determining the quality of forecasts. Various procedures exist but all involve measures of the relationship between a forecast and corresponding observations [90]. Statistical performance functions of Root Mean Square Error (RMSE) and Correlation Factor (R) are used to measure the performance of the developed model and the results are summarized as in Table 6.5. In Addition to these, forecast plots have also been used as qualitative performance functions.

Table 6.5: Summary of Results

Validation Data		RFM Using Forecasted Weather Variables		Fuzzy Time Series	
RMSE	**R**	**RMSE**	**R**	**RMSE**	**R**
28.02	0.97	54.99	0.88	66.97	0.82

Considering the nature of the problem, that is the unpredictability of rainfall, these are fairly reasonable results. These results cover the validation period of 2003 to 2005. The results for the validation period are obtained using the actual weather variables with the developed Rain Forecast Model (RFM). The forecasted rainfall is then compared with the actual rainfall. This now forms the basis for comparing the performance of the developed model when subjected to the forecasted weather variables for the same period. The result obtained from using the fuzzy time series method for the same period is then used to compare the performance of the two models using that of the validation period as a reference. It can be seen from Table 6.5 that the developed Rain Forecast Model (RFM) performed better than the fuzzy time series method of **Abbasov et al [23].**

The performances of the two models for the entire forecast period of 2003 to 2010 are then compared using the statistical performance functions of Root Mean Square Error (RMSE) and Correlation Factor (R). This is meant to determine how the results obtained from the developed Rain Forecast Model (RFM) compare with those obtained from an already existing fuzzy time series method. The result is as summarized in Table 6.6.

Table 6.6: Summary of Result of Comparison of Models

Rain Forecast Model (RFM) v. Fuzzy Time Series	
RMSE	**R**
32.69	0.94

This shows a degree of consistency with respect to the results obtained using the two methods. The fuzzy time series forecasting method [23] adopted in this work, however, has the following drawbacks:

i) It requires a large amount of computations to derive the fuzzy relationship matrix, $\mathbf{R^w(t)}$.

ii) The max-min composition operations will take a large amount of computation time when the fuzzy relationship matrix $\mathbf{R^w(t)}$ is very big.

It is worth noting that high amount of computation time means higher cost.

Since the model developed in this work is data driven, it means that the quality of the forecast is dependent on the quality of the input data. It also invariably means that an unforeseen event, or an event that was not necessarily taken into account, such as an unexpected level of rainfall (high or low), drought, would not be reflected accurately in the computation

CHAPTER SEVEN

CONCLUSIONS AND SUGGESTIONS FOR FURTHER WORK

7.1 INTRODUCTION

Zaria is located within the Sudan Savannah zone in a region in which the people's main pre-occupation is farming. As such, rainfall determines the zonal pattern of crops and the seasonal activities of the farmers. The duration and intensity of rainfall also plays a vital role in power generation in Nigeria with three hydro stations: Kainji, Jebba and Shiroro dependent on it. These issues make the development of a forecast model for rainfall very important.

The Soft Computing technique of Neuro-Fuzzy was applied in this work to develop a forecast model, using a modified ***nfMod*** Neuro-Fuzzy System, which can be implemented as a pure Fuzzy Logic System using the Fuzzy Technology Language (FTL). With available past data, the same technique can be used to develop forecast models for various situations like:

i. Load forecasting using inputs like temperature, relative humidity, days of the week, etc or
ii. Fault prediction of electronic systems using voltage and currents as inputs and component temperature and leakage current as outputs or
iii. Water release into a river using inputs like precipitation, water level and reservoir-release and so on.

In order to forecast into the future, a hybrid statistical method consisting of Moving Averages and Exponential Smoothing was used to forecast for the inputs to be used by the developed model.

Implementation of the developed model as a Fuzzy Logic System means that every output can be explained by a set of rules. This is very important in the development of not only forecast models but also intelligent control systems, such as intelligent electronic thermostats [92] among others.

7.2 MAJOR HIGHLIGHTS

What this work has set out to achieve as a contribution to knowledge is the development of a forecast model as a program written in the Fuzzy Technology Language (FTL). The model was implemented as a 'pure' Fuzzy Logic System. As a case study, a Rain Forecast Model (RFM) was developed using weather variables: Relative Humidity, Wind Direction and Wind Speed as inputs in order to forecast for rainfall in Zaria. It was found that these variables can be used to provide reliable short term and long term rainfall forecasts.

The fuzzy time series method used by **Abbasov et al [23]** in forecasting population growth in Azerbaijan was modified and used to forecast rainfall. The results obtained were compared with those obtained using the Rain Forecast Model (RFM). The Rain Forecast Model (RFM) (**RMSE=54.99** and **R=0.88**) has been shown to be more efficient than the fuzzy time series (**RMSE=66.97** and **R=0.82**).

The investigation has further highlighted the fact that short term forecasting is more accurate and reliable than long term forecasting.

7.3 CONSTRAINTS

The following are some of the constraints encountered in carrying out this investigation:

i) Soft Computing techniques, such as Neural Nets, Fuzzy Logic Systems and Neuro-Fuzzy Systems, are mostly data driven techniques. This means that the quality of data is very important to the working of these techniques. In carrying out this work, therefore, one of the main issues is ascertaining the quality of the data used.

ii) Another constraint is the method used in obtaining the data used in developing the model. Using satellite based or real-time, on-line data acquisition methods can enhance the accuracy and authenticity of the data used in developing the forecast model.

iii) Another constraint on the model developed is the possibility of the occurrence of unforeseen events such as drought, an unexpectedly high or low level of rainfall (for example torrential rainfall leading to floods), wind speed, wind direction or relative humidity, among others. These can lead the model to produce less reliable outputs.

7.4 CONCLUSIONS

The Neuro-Fuzzy System developed in this investigation has fifty seven (57) rules. These are the rules that are fired accordingly depending on what weather variables: Relative Humidity, Wind Direction and Wind Speed) were used as inputs in order to produce the rainfall output. The Neuro-Fuzzy System was developed using the fuzzyTECH Neuro-Fuzzy module using the data set of 1993 to 2002 as the training set. Some of the critical issues in the training are determining the number of linguistic terms and the number of iterations. The number of linguistic terms for both input and output variables was set at five while the number of iterations was two hundred and fifty (250). After the training, the system is tested for its ability to generalize, that is its response to previously unseen data, using the interactive watch window with the validation data of 2003 to 2005. The developed system was then implemented as a pure Fuzzy Logic System using the Fuzzy Technology Language (FTL) as the Rain Forecast Model (RFM).

The Rain Forecast Model (RFM) used as its inputs the weather variables and in order to forecast into the future, these weather variables were forecasted using a hybrid statistical technique comprising of Moving Averages and Exponential Averages. The forecast period is between 2003 and 2010. A fuzzy time series technique was also applied to forecast for the rainfall so as to have a basis for comparison with the developed Rain Forecast Model (RFM).

Two statistical tests were carried out in order to measure the performance of the developed model and that of the fuzzy time series method namely Root Mean Squared

Error (RMSE) and Correlation Factor (R). The developed model was tested with the validation data of 2003 to 2005 and the following performance function results were obtained: **RMSE=28.02** and **R=0.97.** This formed the basis for comparing the performance of the developed model when subjected to the forecasted weather variables for the same period. The performance function results obtained were: **RMSE=54.99** and **R=0.88.** The fuzzy time series method was also applied for the same period and the results obtained used to compare the performance of the two models. The performance function results obtained were: **RMSE=66.97** and **R=0.82.** It can then be said that the developed Rain Forecast Model (RFM) performed better than the fuzzy time series method of **Abbasov et al [23]** in the validation period**.**

In order to determine how the results obtained from the developed Rain Forecast Model (RFM) compare with those obtained from an already existing fuzzy time series method, the performances of the two models for the entire forecast period of 2003 to 2010 are then compared using the statistical performance functions of Root Mean Square Error (RMSE) and Correlation Factor (R). The results obtained were: **RMSE=32.69** and **R=0.94.** This shows a degree of consistency with respect to the results obtained using the two methods but the developed model has the added advantage of requiring lesser computation time.

7.5 SUGGESTIONS FOR FURTHER WORK

The following are suggested as possible ways of improving this work:

i) Since the developed model depends on the weather variables as its inputs, it is recommended that other techniques apart from the hybrid statistical method of Moving Averages and Exponential Smoothing used in this work be looked into as a way of improving on the quality of the forecast. One of such techniques is the Fuzzy Nearest Neighbour Method (FNNM). This method, which is based on the identification of nearest neighbours in the past data, depends on identifying local variation in the time series data rather than modeling the global nature of the time series behaviour [93][94].

ii) The possibility of extending this work in order to produce a rain forecast model for Kaduna State in particular and Northern Nigeria in general should be explored. This is because of the critical role that rainfall plays in the lives and economies of its people.

iii) The developed model can further be enhanced by incorporating such linguistic inputs as occurrence of drought, floods, among others. One of the advantages of the Neuro-Fuzzy technique is that human intuition still has value in the final edit of forecasting models.

iv) The methodology adopted in developing the forecast model can also be adopted for such areas as Telecommunications and Power Systems in order to produce forecast models for telephone growth or power system demand among others. These sectors have been deregulated in Nigeria and it is expected that with forecast models that can project growth in both the short term and long term, efficiency and proper planning can be enhanced. This is

more so for the power sector where so much has been expended in the last few years without much tangible returns.

v) It is recommended that the developed forecast model properly packaged and marketed to organizations that rely on such models for their operations.

vi) Further investigation is required in developing a forecast model that will be more reliable and efficient in carrying out long term forecast.

REFERENCES

1) **WILAMOWSKI, B.M. (2003),** "Neural Network Architectures And Learning", Proceedings of ICIT 2003, Maribor- Slovenia, ppTU1 - ppTU12.

2) **RINGWOOD, J.V. (2002),** "Intelligent Forecasting of Electricity Demand", Proceedings of European Symposium on Intelligent Technologies, Abufeira-Portugal. **www.eeng.may.ie/~jringwood/publist.htm**

3) **MITRA, S., and HAYASHI, Y. (2000),** "Neuro-Fuzzy Rule Generation: Survey In Soft Computing Framework", IEEE Transactions on Neural Networks, Vol. 11, No. 3, pp748 - 768.

4) **SEE, l., ABRAHAT, R. J., and OPENSHAW, S. (1998),** "An Integrated Neuro-Fuzzy-Statistical Approach To Hydrological Modeling", Proceedings of 3rd International Conference on Geo-Computation, University of Bristol.

5) **SEE, l., and OPENSHAW, S. (1998), "**Using Soft Computing Techniques To Enhance Flood Forecasting On The River Ouse", Proceedings of Hydromatics '98, Copenhagen-Denmark.

6) **ABRAHAM, A. and PHILIP, N. S. (2001),** "Soft Computing Models For Weather Forecasting"

http://meghnad.iucaa.ernet.in/~nspp/scs.pdf

7) **ABRAHAM, A., STEINBERG, D., and PHILIP, N. S. (2001)** "Rainfall Forecasting Using Soft Computing Models And Multivariate Adaptive Regression Splines"

http://meghnad.iucaa.ernet.in/~nspp/ieee.pdf

8) **NGUYEN, T. D. (2002),** "AI Techniques For Prediction of Short Data Series With Unreliable Data Quality".

www.cig.ensmp.fr/~iahs/PUBs/Brasilia-Papers/Nguyen.pdf

9) **KUTSURELIS, J.E. (1998), "**Forecasting Financial Markets Using Neural Networks: An Analysis of Methods and Accuracy", M.Sc Thesis, Naval Post Graduate School, Monterey, pp2 – 17.

10) **Inform GmbH (2001),** fuzzyTECH® 5.5 User's Manual, Inform Software Corporation, pp1 - 147.

11) **STERGIOU, C. (**1995), "What Is A Neural Network?"

www.dse.doc.ic.ac.uk/~nd/surprise_96/journal/vol1/cs11/article1.html

12) **SANAL, R. P. (1999),** "Intelligent Traffic Signals Using Fuzzy Logic Control", Proceedings of 11th Kerala Science Congress, Kerala-India, pp1 - 6.

13) **WANG, X. and TSOUKALAS, L. H. (2002),** "A New Load Forecasting Methodology".

www.helios.ecn.purdue.edu/~aisl/misc/paperpdf/conference/turkey_2002.PDF

14) **AHMAD, A. M., CHUAN, C. S., and MOHAMAD, F. (2002),** "Weather Prediction Using Artificial Neural Network", Proceedings of ITC-CSCC 2002: The International Technical Conference On Circuits/Systems, Computers and Communications, Phuket-Thailand.

http://216.239.41.104/pdf/17_07_45/WP1_OB/3.pdf

15) **LOTFI, A. (2002),** "Application of Learning Fuzzy Inference Systems in Electricity Load Forecast"

http://neuron.tuke.sk/competition/reports/AhmadLotfi.pdf

16) **CUEVAS, E., ZALDIVAR, D., and ROJAS, R. (2004),** "NeuroFuzzy Prediction For Visual Tracking", International Conference On Electrical and Electronics Engineering, IEEE, ICEEE 2004.

http://page..mi.fu-berlin.de/~zaldivar/iceee-Neuro_fuzzy.pdf

17) **KOOTHS, S. (1999),** "Modeling Rule-and Experience-Based Expectations Using Neuro-Fuzzy Systems"

www.makromat.de/cef99-kooths%20(presentation).pdf

18) **LEE, B., GUESGEN, H. W., BALTES, J. and JEONG, S. (2003),** "The Application of Fuzzy Reasoning To The Opening Games of 19x19 Go"

www.citr.auckland.ac.nz/techreports/2003/CITR-TR-135.pdf

19) **GORZALCZANY, M. B. and GLUSZEK, A. (2000),** "NeuroFuzzy System For Rule-Based Modeling of Dynamic Processes", Proceedings of ESIT 2000, Aachen-Germany, pp416 - 422.

20) **TOMÉ, J. A. B. and CARVALHO, J. P. (2005),"**One Step Ahead Prediction Using Fuzzy Boolean Neural Networks", Joint EUSFLAT-LFA 2005, 4th Conference of The European Society For Fuzzy Logic and Technology, Barcelona, Spain.

www.inesc-id.pt/ficheiros/publicacoes/2533.pdf

21) **EASTON, V. J. and McCOLL, J. H. (1997),** "Time Series Data", STEPS' Statistics Glossary v1.1.

www.stats.gla.ac.uk/steps/glossary/time_series.html

22) **ROBERTS, P. (2004),** "Moving Averages: The Heart of Trend Analysis", Alchemist, Journal of The London Bullion Market Association, Issue 33, pp12-14.
www.lmba.org.uk/publications/alchemist/alch33_averages.pdf

23) **ABBASOV, A. M., and MAMEDOVA, M. H., (2003),** "Application of Fuzzy Time Series to Population Forecasting", Vienna University of Technology.
www.corp.at/corp/archive/papers/2003/CORP2003_Abbasov.pdf

24) **LEE, T-S., CHIU, C-C., and LIN, F-C. (2001),** "Prediction of the Unemployment Rate Using Fuzzy Time Series With Box-Jenkins Methodology", International Journal of Fuzzy Systems, Vol.3, No.4, pp577-585.

25) Engineering Statistics Handbook: "Introduction to Time Series Analysis".
www.itl.nist.gov/div898/handbook/pmc/section4/pmc4.htm

26) **WANG, X., HATZIARGYRIOU, N. and TSOUKALAS, L. H. (2002),** "A New Methodology For Nodal Load Forecasting In Deregulated Power Systems", IEEE Power Engineering Review, pp48 - 51.

27) **TSOUKALAS, L. H. (1998),** "NeuroFuzzy Approaches To Anticipation: A New Paradigm For Intelligent Systems", IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics, Vol.28, No.4, pp573 - 582.

28) **Decision Engineering (2005),** Risk Analysis Overview: What is Time-Series Forecasting?
www.decisioneering.com/index.html

29) **LIU, X., KWAN, B. W., and FOO, S. Y. (2001),** "Time Series Prediction Based On Fuzzy Principles"
http://devils.eng.fsu.edu/download/Time_Series_Prediction_Based_on_Fuzzy_Principles.pdf

30) "Weather Forecasting"
www.rcn27.dial.pipex.com/cloudrus/forecasting.html

31) **CHEONG, C. W., LAN, A. L. H., and RAMACHANDRAN, V. C. (1999),** "Web Server Workload Forecasting – Fuzzy Linguistic Approach"
http://persona.mmu.edu.my/~wcchin/edu/edu.html

32) **LINACRE, E. and GEERTS, B., (2002),** "Effect of Local Sea Surface Temperature on Inland Rainfall".
www.das.uwyo.edu/~geerts/cwx/notes/chap10/localsst_rain.html

33) **GRODSKY, S. A., and CARTON, J. A., (2002),** "Effect of Intra-seasonal Wind Fluctuations in The West Africa Monsoon on Air-Sea Fluxes"
www.soc.soton.ac.uk/JRD/MET/WGASF/workshop/WABSTRACTS/Grodsky.php3

34) **KELEMEN, A., KOZMA, R., and LIANG, Y. (2002),** "Neuro-Fuzzy Classification For The Job Assignment Problem"
www.msci.memphis.edu/~kelemena/paper/WCCI/fuzzypaper7camera.pdf

35) **FINZI, G., NUNNARI, G., and VOLTA, M. (1999),** "Modeling And Short Term Forecasting of Petrochemical Pollution By Soft Computing Techniques"
www.iemss.org/iemss2002/proceedings/pdf/volume%20due/359_finzi.pdf

36) "NNI: The Perceptron Learning Rule"
www.abo.fi/rfuller/nf8.pdf

37) "Artificial Neural Network: Non-Linear Black Box Modeling Techniques", www.ncl.ac.uk/cpact/neural_networks.html

38) **VALPOLA, H. (2004),** "Neural Networks Lecture 3: Multi-Layer Perceptrons", Artificial Intelligence Lab, University of Zurich. http://ailab/teaching/classes/2004ss/nn/lecture3.pdf

39) **ZADEH, L. A., (1965), "**Fuzzy Sets", Information and Control, Vol. 8, pp338-353.

40) **JANG, J-S. R., SUN, C. T., and MIZUTANI, E., (1997),** Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, Prentice-Hall Inc, pp1-328.

41) **BLUM, A., (1992),** Neural Networks in C++: An Object-Oriented Framework for Building Connectionist Systems, John Wiley and Sons Inc. pp35-111.

42) **IHE Delft (2000),** "Use of Artificial Neural Network And Fuzzy Logic For Integrated Water Management: Review of Applications" Project Report, pp3- 5.

43) "Chapter 19, Sections 1 - 5: Neural Networks" http://aima.eecs.berkeley.edu/slides-pdf/chapter19-6pp.pdf

44) **MUKHERJEE, A. (2003),** "Advances In Battery Management Using Neural Networks And Fuzzy Logic", Departmental Seminar (ENGRC350), School of Electrical and Computer Engineering, Cornell University, ppA.3 - A.7 www.engineering.cornell.edu/engrc350/ingenuity/Muhkrejee_A_issue_1.pdf

45) **ISIK, C. (1999),** "Neural Networks And Fuzzy Logic", CNY-ACE Lecture Notes. www.syr.edu/~cisik/nn_fl.ppt.

46) **WATKINS, D. (2000),** "Neural Networks MasterClass", SPSS CLUG 2000. www.spss.com/clementine/clug/ppts/NNM.ppt

47) **TIŇO, P. (2000),** "Neural Network Applications" www.cs.bham.ac.uk//~pxt/TALKS.nn.appl.pdf

48) **PURVIS, M., KASABOV, N., BENWELL, G., ZHOU, G., and ZHANG, F. (1999),** "Neuro-Fuzzy Methods For Environmental Modeling", Journal of Systems Research And Information Systems, Vol.8, pp221 - 239.

49) **ISLAM, S. M., AL-ALAWI, S. M., and ELLITHY, K. A., (1995),** "Forecasting Monthly Electric Load And Energy For A fast Growing Utility Using An Artificial Neural Network", Electric Power Systems Research, vol.34, No.1, pp1-9.

50) **ARISARIYAWONG, S. and CHAROENSEANG, S. (2001),** "Reducing Steady State Error of A Direct Drive Robot Using NeuroFuzzy Control", Proceedings of Second Asian Symposium on Industrial Automation And Robotics, Bangkok-Thailand.

51) **ROTSHTEIN, A., POSNER, M., and RAKYTYANSKA, H. (2003),** "Prediction of The results of Football Games Based On Fuzzy Model With Genetic And Neuro Tuning", Eastern European Journal of Enterprise Technologies, pp10 - 15.

52) **EDWARDS, V. (1998),** "Artificial Neural Networks", http://aspac.web.unsw.edu.au/Media%20Room/week4.pdf

53) **CHUJO, N., KUROYANAGI, S., DOKI, S., and OKUMA, S. (2002),** "An Iterative Calculation Method of Neuron Model With Sigmoid Function" www.iee.or.jp/honbu/back_number/journal/index_back_number/2002/2002_03c_07.pdf

54) **BRASIL, L. M., DE AZEVEDO, F. M., BARRETO, J. M., and NOIRHOMME-FRAITURE, M. (1998),** "A New Approach To Classical BackPropagation Algorithm For Neuro-Fuzzy-GA Systems Learning"

www.fundp.ac.be/recherche/personnes/publications/fr/01000245.html

55) **"**Back Propagation Algorithm"

www.xs4all.nl/~dpsol/data_machine/nmtutorial/backpropagationalgorithm.htm

56) "Chapter 3: Neural Networks"

www.control.hut.fi/Kurssit/AS-74.115/Material/SOFTCOMPCH03a.pdf

57) **HEATON, J. (2003), "**Chapter 5: Understanding Back Propagation"

www.jeffheaton.com

58) **SCARDI, M. (1999),** "An introduction To Error Back Propagation Neural Networks."

www.mare-net.com/mscardi/work/nnintro.htm

59) **BRASIL, L. M., DE AZEVEDO, F. M., BARRETO, J. M., and NOIRHOMME-FRAITURE, M. (1998), "**Training Algorithm For Neuro-Fuzzy-GA Systems"

www.inf.ufsc.br/~barreto/artiges/Barreto98.pdf

60) **HIIRSALMI, M., KOTSAKSIS, F., POSENEN, A., and WOLSKI, A. (2000),** "Discovery of Fuzzy Models From Observation Data", FUME Project, VTT Info Research, Report TTE1-2000-43, v1.0

www.vt.fi/ttr/prjects/fume/

61) **"**Definition of Logistic Function"

www.wordiq.com/definition/Logistic_function

62) **JONES, A. (1998),** "What If It's Not Linear? (nets)"

http://users.aber.ac.uk/auj/talks/depttalk97/nets.html

63) **MATTHEWS, J. (1999),** "An Introduction To Fuzzy Logic"

www.generation5.org/fuzzyintro.shtml

64) "FSI: Fuzzy Sets And Fuzzy Logic"

www.abo.fi/rfuller/nf1.pdf

65) "NeuroFuzzy And Hybrid Approaches"

www.stowa-nn.ihe.nl/NFS.htm

66) **GAO, Z., TRAUTZSCH, T. A., and DAWSON, J. G. (2000),** "A Stable Self-Tuning Fuzzy Logic Control System For Industrial Temperature Regulation", http://academic.csuohio.edu/aerl/papers/IAS2000_FLC_TempCtrl.pdf

67) **BRULE, J. (1985),** "Fuzzy Systems - A Tutorial"

www.austinlinks.com/Fuzzy/tutorial.html

68) **MAHABIR, C., HICKS, F. E., and ROBINSON FAYEK, A. (2003),** "Application of Fuzzy Logic To Forecast Seasonal Runoff", Hydrol. Process, 17, pp3749 - 3762. Published On-Line: www.interscience.wiley.com

69) **PYTELKOVA, R. (2001),** "Modeling And Control With NeuroFuzzy Systems", Helsinki Summer School, Department of Control Engineering, FEE, CTU.

www.helsinki.fi/~niskanen/sc2001/rp.pdf

70) **SOKOLOWSKI, A. (2003),** "On Some Aspects of Fuzzy Logic System" , Proceedings of AI-METH 2003, Gilwice-Poland, pp128 - 129.

71) **COSTA BRANCO, P. J. and DENTE, J. A. (1998),** "Design of An Electro-Hydraulic System Using NeuroFuzzy Techniques"

http://arxiv.org/pdf/cs.RO/0010001

72) **ZAPTRON Systems Inc. (1999),** "NeuroFuzzy: A Different Type of Neural Networks- Zaptron's High-Order Non-Linear Neural Networks"

www.zaptron.com/literature/neurofuzzy.htm

73) **ABRAHAM, A. (2001),** "Cerebral Quotient of NeuroFuzzy Techniques- Hype or Hallelujah?"

http://ajith.softcomputing.net

74) **KLOSE, A., NURNBERGER, J. A., and NAUCK, D. (2000)**, "Some Approaches To Improve The Interpretability of Neuro-Fuzzy Classifiers"

http://fuzzy.cs.uni-magdeburg.de

75) **XIAO, H., SUN, H., and RAN, B. (2004),** "The Special Factor Adjustment Model Using Fuzzy-Neural Network In Traffic Prediction", TRB2004 Annual Meeting CD-ROM.

www.topslab.wisc.edu/resources/publications/ran/2004/ran_2688.pdf

76) **ABRAHAM, A. (2001),** "Neuro Fuzzy Systems: State-of-the-Art Modeling Techniques"

www.infotech.monash.edu.au/research/General_info/strengths/intelligent.pdf

77) FSV: Fuzzy Rule-Based Systems"

www.abo.fi/rfuller/nf5.pdf

78) **GORZALCZANY, M. B. (2002),** Computational Intelligence Systems And Applications: Neuro-Fuzzy and Fuzzy Neural Synergisms, Studies in Fuzziness And Soft Computing, Physica-Verlag, pp17 – 184.

79) **HESSAMI, M., ANCTIL, F., and VIAU, A. A. (2004), "**Selection of An Artificial Neural Network Model For The Calibration of Weather Radar Rainfall Estimation", Journal of Data Science, Vol. 2, pp107 – 124.

80) **TODOROV, E. (2004),** "Lecture 6: Computing Gradients Via Back-Propagation", CogSci 108E: Neural Network Models of Cognition, Cognitive Science Department, University of California, San Diego.
http://eotroscollegium.hu/~szityu/Lec6-Backpropagation.pdf

81) **VON ALTROCK, C. (1995),** Fuzzy Logic And NeuroFuzzy Applications Explained, Prentice Hall PTR, pp29 – 319.

82) **ROJAS, I. And POMARES, H. (2004),** "Soft Computing Techniques For Time Series Forecasting", European Symposium on Artificial Neural Networks, Bruges, Belgium, ESANN'2004 Proceedings, pp93 – 102.

83) **MONKS, J. G. (1996),** Schaum's Outline of Theory And Problems of Operations Management, 2.ed, McGraw-Hill Inc, pp39 – 43.

84) "Introduction To Moving Averages"
www.cne.gmu.edu/modules/dau/mvavgs/mvavgs_frm.html

85) **"**Averaging And Exponential Smoothing Models" (Decision 411 Forecasting)
www.duke.edu/~mau/411avg.htm

86) **MAMDANI, E. H. and ASSILIAN, S. (1973),** "An Experiment in Linguistic Synthesis With A Fuzzy Logic Controller", International Journal of Man-Machine Studies, vol.7, pp1 – 13.

87) **CHEN, S-M., and HSU, C-C. (2004),** "A New Method to Forecast Enrollments Using Fuzzy Time Series", International Journal of Applied Science and Engineering, Vol.2, No.3, pp234-244.

88) **WU, S. J. [2001],** "On The Modeling of Wind Characteristics in Taiwan Using A Web-Based Fuzzy Forecasting System"
http://utfinance.yuntech.edu.tw/images/School/SJSMIT/93/D1.PDF

89) "What's The Bottom Line? How To Compare Models" (Decision 411 Forecasting)
www.duke.edu/~mau/compare.htm

90) "Weather Forecasting Models", Science History Study Guide.
www.bookrags.com/sciences/sciencehistory/weather-forecasting-models-wsd.html

91) **SOUTH AFRICAN WEATHER SERVICE [2006],** "Research and Technology: Prediction Research"
www.saweather.co.za/Menus/Research.jsp

92) **MU'AZU, M.B. and BOZIMO, T. [2006],** "Design and Simulation Of An Intelligent Fuzzy Logic Controlled Electronic Thermostat", International Conference on Artificial Intelligence (ICAI'06), Las Vegas, U.S.A.

93) **ALBEANU, G., and POPENTIU-VLADICESCU, F. [2005],** " On Using The Fuzzy Nearest Neighbour method for time series forecasting in software reliability", Proceedings of SIG'05, Sinaia, Romania, pp1-4.

94) **SINGH, S. [1998],** "Fuzzy Nearest Neighbour Method For Time Series Forecasting", Proceedings of 6th European Congress on Intelligent Techniques and Soft Computing, Germany, vol. 3, pp1901-1905

APPENDIX A1

COMPLETE TRAINING DATA (1993 – 2002)

MONTH	REL_HUM	WIND_DIR	WIND_SPD	RAINFALL
APRIL	42.7	29	200.3	38.9
MAY	62.9	32	235.22	83.6
JUNE	72.7	129	210.6	88
JULY	78.4	104	179.68	244.3
AUGUST	81.8	102	163.7	281.9
SEPTEMBER	76.7	65	129.61	199.7
OCTOBER	63.1	26	107.88	18.8
APRIL	49	42	222.73	33.1
MAY	71.7	57	185.03	78.7
JUNE	77.4	83	230.44	137.1
JULY	84.3	66	202.56	125.5
AUGUST	85	93	149.31	352.4
SEPTEMBER	82.8	51	97.02	203.7
OCTOBER	76.4	28	46.06	166.1
APRIL	41.2	13	197.3	59.1
MAY	68	24	185.3	103
JUNE	70.6	31	211.78	153.7
JULY	79.5	33	200.28	235.5
AUGUST	81.8	10	138.25	294.3
SEPTEMBER	81.3	20	115.88	112
OCTOBER	74.3	10	111.43	32.9
APRIL	55.8	32	175.53	8.9
MAY	69.8	82	244.61	149.6
JUNE	77.5	80	218.16	186.3
JULY	79.4	64	227.34	184.8
AUGUST	82	63	164.86	267
SEPTEMBER	81.5	52	155.57	173.5
OCTOBER	65.5	24	142.36	59.9
APRIL	57.2	49	223.27	29.4
MAY	74.9	64	195.75	176.9
JUNE	79.6	80	196.57	146.8
JULY	84.5	65	208.48	232.8
AUGUST	83.4	48	146.6	355.7
SEPTEMBER	80.8	33	124.33	192.3
OCTOBER	78.6	33	125.09	64.9
APRIL	45	98	190.24	33.4
MAY	76.7	73	252.16	123.3
JUNE	82.4	65	180.7	144
JULY	84.3	96	209.59	184.3
AUGUST	87.6	72	151.63	473.1
SEPTEMBER	86.1	56	131.14	236.8
OCTOBER	78.8	33	130.32	71.4
APRIL	42	42	219.12	10.5

MONTH	REL_HUM	WIND_DIR	WIND_SPD	RAINFALL
MAY	61.2	66	201.16	7.2
JUNE	78.4	49	180.7	229.5
JULY	87.1	77	168.22	245.7
AUGUST	89.1	79	151.63	144.7
SEPTEMBER	86.1	47	126.56	268.2
OCTOBER	75.4	27	108.39	78
APRIL	53	53	219.12	17.8
MAY	64.1	76	202.24	107.6
JUNE	82.1	79	208.48	157.1
JULY	85.9	87	181.08	268
AUGUST	87.9	83	154.87	298.4
SEPTEMBER	82.4	60	149.44	177.7
OCTOBER	71.2	35	128.39	63.3
APRIL	52.4	73	230.37	99.8
MAY	75.7	77	201.82	100.4
JUNE	81.2	98	206.36	189
JULY	87.2	83	181.23	255.7
AUGUST	89.9	80	153	244.9
SEPTEMBER	86.7	41	100.77	312.6
OCTOBER	63	12	114.87	0
APRIL	68	70	209.1	35
MAY	79.4	97	213.53	10.9
JUNE	85.7	93	201.28	85.9
JULY	90.1	96	185.07	181.2
AUGUST	90.6	67	132.77	210.2
SEPTEMBER	88.7	51	102.6	200.1
OCTOBER	86.6	33	81.8	128.7

APPENDIX A2

COMPLETE RULE BASE (625 RULES)

Aggregation:	MINMAX
Parameter:	0.00
Result Aggregation:	MAX
Number of Inputs:	3
Number of Outputs:	1
Number of Rules:	625

IF			THEN	
REL_HUM	**WIND_DIR**	**WIND_SPD**	**DoS**	**RAINFALL**
very_low	very_low	very_low	0.00	very_low
very_low	very_low	very_low	0.00	Low
very_low	very_low	very_low	0.00	Medium
very_low	very_low	very_low	0.00	High
very_low	very_low	very_low	0.00	very_high
very_low	very_low	low	0.00	very_low
very_low	very_low	low	0.00	Low
very_low	very_low	low	0.00	Medium
very_low	very_low	low	0.00	High
very_low	very_low	low	0.00	very_high
very_low	very_low	medium	0.00	very_low
very_low	very_low	medium	0.00	Low
very_low	very_low	medium	0.00	Medium
very_low	very_low	medium	0.00	High
very_low	very_low	medium	0.00	very_high
very_low	very_low	high	0.09	very_low
very_low	very_low	high	0.00	Low
very_low	very_low	high	0.00	Medium
very_low	very_low	high	0.00	High
very_low	very_low	high	0.00	very_high
very_low	very_low	very_high	0.05	very_low
very_low	very_low	very_high	0.00	Low
very_low	very_low	very_high	0.00	Medium
very_low	very_low	very_high	0.00	High
very_low	very_low	very_high	0.00	very_high
very_low	Low	very_low	0.00	very_low
very_low	Low	very_low	0.00	Low
very_low	Low	very_low	0.00	Medium
very_low	Low	very_low	0.00	High
very_low	Low	very_low	0.00	very_high

IF				THEN
very_low	Low	low	0.00	very_low
very_low	Low	low	0.00	Low
very_low	Low	low	0.00	Medium
very_low	Low	low	0.00	High
very_low	Low	low	0.00	very_high
very_low	Low	medium	0.00	very_low
very_low	Low	medium	0.00	Low
very_low	Low	medium	0.00	Medium
very_low	Low	medium	0.00	High
very_low	Low	medium	0.00	very_high
very_low	Low	high	0.06	very_low
very_low	Low	high	0.00	Low
very_low	Low	high	0.00	medium
very_low	Low	high	0.00	high
very_low	Low	high	0.00	very_high
very_low	Low	very_high	0.00	very_low
very_low	Low	very_high	0.00	Low
very_low	Low	very_high	0.00	medium
very_low	Low	very_high	0.00	high
very_low	Low	very_high	0.00	very_high
very_low	Medium	very_low	0.00	very_low
very_low	Medium	very_low	0.00	Low
very_low	Medium	very_low	0.00	medium
very_low	Medium	very_low	0.00	high
very_low	Medium	very_low	0.00	very_high
very_low	Medium	low	0.00	very_low
very_low	Medium	low	0.00	Low
very_low	Medium	low	0.00	medium
very_low	Medium	low	0.00	high
very_low	Medium	low	0.00	very_high
very_low	Medium	medium	0.00	very_low
very_low	Medium	medium	0.00	Low
very_low	Medium	medium	0.00	medium
very_low	Medium	medium	0.00	high
very_low	Medium	medium	0.00	very_high
very_low	Medium	high	0.00	very_low
very_low	Medium	high	0.00	Low
very_low	Medium	high	0.00	medium
very_low	Medium	high	0.00	high
very_low	Medium	high	0.00	very_high
very_low	Medium	very_high	0.00	very_low
very_low	Medium	very_high	0.00	Low
very_low	Medium	very_high	0.00	medium

IF			THEN	
very_low	medium	very_high	0.00	high
very_low	medium	very_high	0.00	very_high
very_low	high	very_low	0.00	very_low
very_low	high	very_low	0.00	Low
very_low	high	very_low	0.00	medium
very_low	high	very_low	0.00	high
very_low	high	very_low	0.00	very_high
very_low	high	low	0.00	very_low
very_low	high	low	0.00	Low
very_low	high	low	0.00	medium
very_low	high	low	0.00	high
very_low	high	low	0.00	very_high
very_low	high	medium	0.00	very_low
very_low	high	medium	0.00	Low
very_low	high	medium	0.00	medium
very_low	high	medium	0.00	high
very_low	high	medium	0.00	very_high
very_low	high	high	0.04	very_low
very_low	high	high	0.00	Low
very_low	high	high	0.00	medium
very_low	high	high	0.00	high
very_low	high	high	0.00	very_high
very_low	high	very_high	0.00	very_low
very_low	high	very_high	0.00	Low
very_low	high	very_high	0.00	medium
very_low	high	very_high	0.00	high
very_low	high	very_high	0.00	very_high
very_low	very_high	very_low	0.00	very_low
very_low	very_high	very_low	0.00	Low
very_low	very_high	very_low	0.00	medium
very_low	very_high	very_low	0.00	high
very_low	very_high	very_low	0.00	very_high
very_low	very_high	low	0.00	very_low
very_low	very_high	low	0.00	low
very_low	very_high	low	0.00	medium
very_low	very_high	low	0.00	high
very_low	very_high	low	0.00	very_high
very_low	very_high	medium	0.00	very_low
very_low	very_high	medium	0.00	low
very_low	very_high	medium	0.00	medium
very_low	very_high	medium	0.00	high
very_low	very_high	medium	0.00	very_high
very_low	very_high	high	0.00	very_low

IF			THEN	
very_low	very_high	high	0.00	low
very_low	very_high	high	0.00	medium
very_low	very_high	high	0.00	high
very_low	very_high	high	0.00	very_high
very_low	very_high	very_high	0.00	very_low
very_low	very_high	very_high	0.00	low
very_low	very_high	very_high	0.00	medium
very_low	very_high	very_high	0.00	high
very_low	very_high	very_high	0.00	very_high
low	very_low	very_low	0.08	very_low
low	very_low	very_low	0.00	low
low	very_low	very_low	0.00	medium
low	very_low	very_low	0.00	high
low	very_low	very_low	0.00	very_high
low	very_low	low	0.09	very_low
low	very_low	low	0.00	low
low	very_low	low	0.00	medium
low	very_low	low	0.00	high
low	very_low	low	0.00	very_high
low	very_low	medium	0.00	very_low
low	very_low	medium	0.00	low
low	very_low	medium	0.00	medium
low	very_low	medium	0.00	high
low	very_low	medium	0.00	very_high
low	very_low	high	0.00	very_low
low	very_low	high	0.00	low
low	very_low	high	0.00	medium
low	very_low	high	0.00	high
low	very_low	high	0.00	very_high
low	very_low	very_high	0.00	very_low
low	very_low	very_high	0.08	low
low	very_low	very_high	0.00	medium
low	very_low	very_high	0.00	high
low	very_low	very_high	0.00	very_high
low	low	very_low	0.00	very_low
low	low	very_low	0.00	low
low	low	very_low	0.00	medium
low	Low	very_low	0.00	high
low	Low	very_low	0.00	very_high
low	Low	low	0.00	very_low
low	Low	low	0.00	low
low	Low	low	0.00	medium
low	Low	low	0.00	high

IF			THEN	
low	Low	low	0.00	very_high
low	Low	medium	0.00	very_low
low	Low	medium	0.00	low
low	Low	medium	0.00	medium
low	Low	medium	0.00	high
low	Low	medium	0.00	very_high
low	Low	high	0.00	very_low
low	Low	high	0.00	low
low	Low	high	0.00	medium
low	Low	high	0.00	high
low	Low	high	0.00	very_high
low	Low	very_high	0.00	very_low
low	Low	very_high	0.00	low
low	Low	very_high	0.00	medium
low	Low	very_high	0.00	high
low	Low	very_high	0.00	very_high
low	medium	very_low	0.00	very_low
low	medium	very_low	0.00	low
low	medium	very_low	0.00	medium
low	medium	very_low	0.00	high
low	medium	very_low	0.00	very_high
low	medium	low	0.00	very_low
low	medium	low	0.00	low
low	medium	low	0.00	medium
low	medium	low	0.00	high
low	medium	low	0.00	very_high
low	medium	medium	0.00	very_low
low	medium	medium	0.00	low
low	medium	medium	0.00	medium
low	medium	medium	0.00	high
low	medium	medium	0.00	very_high
low	medium	high	0.00	very_low
low	medium	high	0.00	low
low	medium	high	0.00	medium
low	medium	high	0.00	high
low	medium	high	0.00	very_high
low	medium	very_high	0.00	very_low
low	medium	very_high	0.08	low
low	medium	very_high	0.00	medium
low	medium	very_high	0.00	high
low	medium	very_high	0.00	very_high
low	high	very_low	0.00	very_low
low	high	very_low	0.00	low

IF			THEN	
low	high	very_low	0.00	medium
low	high	very_low	0.00	high
low	high	very_low	0.00	very_high
low	high	low	0.00	very_low
low	high	low	0.00	low
low	high	low	0.00	medium
low	high	low	0.00	high
low	high	low	0.00	very_high
low	high	medium	0.00	very_low
low	high	medium	0.00	low
low	high	medium	0.00	medium
low	high	medium	0.00	high
low	high	medium	0.00	very_high
low	high	high	0.00	very_low
low	high	high	0.00	low
low	high	high	0.00	medium
low	high	high	0.00	high
low	high	high	0.00	very_high
low	high	very_high	0.00	very_low
low	high	very_high	0.00	low
low	high	very_high	0.00	medium
low	high	very_high	0.00	high
low	high	very_high	0.00	very_high
low	very_high	very_low	0.00	very_low
low	very_high	very_low	0.00	low
low	very_high	very_low	0.00	medium
low	very_high	very_low	0.00	high
low	very_high	very_low	0.00	very_high
low	very_high	low	0.00	very_low
low	very_high	low	0.00	low
low	very_high	low	0.00	medium
low	very_high	low	0.00	high
low	very_high	low	0.00	very_high
low	very_high	medium	0.00	very_low
low	very_high	medium	0.00	low
low	very_high	medium	0.00	medium
low	very_high	medium	0.00	high
low	very_high	medium	0.00	very_high
low	very_high	high	0.00	very_low
low	very_high	high	0.00	low
low	very_high	high	0.00	medium
Low	very_high	high	0.00	high
Low	very_high	high	0.00	very_high

IF			THEN	
Low	very_high	very_high	0.00	very_low
Low	very_high	very_high	0.00	low
Low	very_high	very_high	0.00	medium
Low	very_high	very_high	0.00	high
Low	very_high	very_high	0.00	very_high
medium	very_low	very_low	0.00	very_low
medium	very_low	very_low	0.00	low
medium	very_low	very_low	0.00	medium
medium	very_low	very_low	0.00	high
medium	very_low	very_low	0.00	very_high
medium	very_low	low	0.00	very_low
medium	very_low	low	0.00	low
medium	very_low	low	0.00	medium
medium	very_low	low	0.00	high
medium	very_low	low	0.00	very_high
medium	very_low	medium	0.00	very_low
medium	very_low	medium	0.00	low
medium	very_low	medium	0.00	medium
medium	very_low	medium	0.00	high
medium	very_low	medium	0.00	very_high
medium	very_low	high	0.00	very_low
medium	very_low	high	0.01	low
medium	very_low	high	0.00	medium
medium	very_low	high	0.00	high
medium	very_low	high	0.00	very_high
medium	very_low	very_high	0.00	very_low
medium	very_low	very_high	0.00	low
medium	very_low	very_high	0.00	medium
medium	very_low	very_high	0.00	high
medium	very_low	very_high	0.00	very_high
medium	Low	very_low	0.00	very_low
medium	Low	very_low	0.00	low
medium	Low	very_low	0.00	medium
medium	Low	very_low	0.00	high
medium	Low	very_low	0.00	very_high
medium	Low	low	0.00	very_low
medium	Low	low	0.00	low
medium	Low	low	0.00	medium
medium	Low	low	0.00	high
medium	Low	low	0.00	very_high
medium	Low	medium	0.00	very_low
medium	Low	medium	0.00	low
medium	Low	medium	0.00	medium

IF			THEN	
medium	Low	medium	0.00	high
medium	Low	medium	0.00	very_high
medium	Low	high	0.00	very_low
medium	Low	high	0.00	low
medium	Low	high	0.00	medium
medium	Low	high	0.00	high
medium	Low	high	0.00	very_high
medium	Low	very_high	0.00	very_low
medium	Low	very_high	0.00	low
medium	Low	very_high	0.00	medium
medium	Low	very_high	0.00	high
medium	Low	very_high	0.00	very_high
medium	medium	very_low	0.00	very_low
medium	medium	very_low	0.00	low
medium	medium	very_low	0.00	medium
medium	medium	very_low	0.00	high
medium	medium	very_low	0.00	very_high
medium	medium	low	0.00	very_low
medium	medium	low	0.00	low
medium	medium	low	0.00	medium
medium	medium	low	0.00	high
medium	medium	low	0.00	very_high
medium	medium	medium	0.00	very_low
medium	medium	medium	0.00	low
medium	medium	medium	0.00	medium
medium	medium	medium	0.00	high
medium	medium	medium	0.00	very_high
medium	medium	high	0.06	very_low
medium	medium	high	0.00	low
medium	medium	high	0.00	medium
medium	medium	high	0.00	high
medium	medium	high	0.00	very_high
medium	medium	very_high	0.11	very_low
medium	medium	very_high	0.00	low
medium	medium	very_high	0.00	medium
medium	medium	very_high	0.00	high
medium	medium	very_high	0.00	very_high
medium	high	very_low	0.00	very_low
medium	high	very_low	0.00	low
medium	high	very_low	0.00	medium
medium	high	very_low	0.00	high
medium	high	very_low	0.00	very_high
medium	high	low	0.00	very_low

IF			THEN	
medium	high	low	0.00	low
medium	high	low	0.00	medium
medium	high	low	0.00	high
medium	high	low	0.00	very_high
medium	high	medium	0.00	very_low
medium	high	medium	0.00	low
medium	high	medium	0.00	medium
medium	high	medium	0.00	high
medium	high	medium	0.00	very_high
medium	high	high	0.00	very_low
medium	high	high	0.00	low
medium	high	high	0.00	medium
medium	high	high	0.00	high
medium	high	high	0.00	very_high
medium	high	very_high	0.00	very_low
medium	high	very_high	0.05	low
medium	high	very_high	0.00	medium
medium	high	very_high	0.00	high
medium	high	very_high	0.00	very_high
medium	very_high	very_low	0.00	very_low
medium	very_high	very_low	0.00	low
medium	very_high	very_low	0.00	medium
medium	very_high	very_low	0.00	high
medium	very_high	very_low	0.00	very_high
medium	very_high	low	0.00	very_low
medium	very_high	low	0.00	low
medium	very_high	low	0.00	medium
medium	very_high	low	0.00	high
medium	very_high	low	0.00	very_high
medium	very_high	medium	0.00	very_low
medium	very_high	medium	0.00	low
medium	very_high	medium	0.00	medium
medium	very_high	medium	0.00	high
medium	very_high	medium	0.00	very_high
medium	very_high	high	0.00	very_low
medium	very_high	high	0.03	low
medium	very_high	high	0.00	medium
medium	very_high	high	0.00	high
medium	very_high	high	0.00	very_high
medium	very_high	very_high	0.00	very_low
medium	very_high	very_high	0.00	low
medium	very_high	very_high	0.00	medium
medium	very_high	very_high	0.00	high

IF				THEN
medium	very_high	very_high	0.00	very_high
high	very_low	very_low	0.09	very_low
high	very_low	very_low	0.00	low
high	very_low	very_low	0.00	medium
high	very_low	very_low	0.09	high
high	very_low	very_low	0.00	very_high
high	very_low	low	0.63	very_low
high	very_low	low	0.00	low
high	very_low	low	0.00	medium
high	very_low	low	0.00	high
high	very_low	low	0.00	very_high
high	very_low	medium	0.00	very_low
high	very_low	medium	0.00	low
high	very_low	medium	0.00	medium
high	very_low	medium	0.00	high
high	very_low	medium	0.00	very_high
high	very_low	high	0.00	very_low
high	very_low	high	0.00	low
high	very_low	high	0.00	medium
high	very_low	high	0.00	high
high	very_low	high	0.00	very_high
high	very_low	very_high	0.00	very_low
high	very_low	very_high	0.00	low
high	very_low	very_high	0.00	medium
high	very_low	very_high	0.00	high
high	very_low	very_high	0.00	very_high
high	low	very_low	0.00	very_low
high	low	very_low	0.00	low
high	low	very_low	0.00	medium
high	low	very_low	0.00	high
high	low	very_low	0.00	very_high
high	low	low	0.00	very_low
high	low	low	0.00	low
high	low	low	0.00	medium
high	low	low	0.00	high
high	low	low	0.00	very_high
high	low	medium	0.00	very_low
high	low	medium	0.00	low
high	low	medium	0.00	medium
high	low	medium	0.00	high
high	low	medium	0.00	very_high
high	low	high	0.00	very_low
high	low	high	0.00	low

IF			THEN	
high	low	high	0.00	medium
high	low	high	0.00	high
high	low	high	0.00	very_high
high	low	very_high	0.00	very_low
high	low	very_high	0.00	low
high	low	very_high	0.00	medium
high	low	very_high	0.00	high
high	low	very_high	0.00	very_high
high	medium	very_low	0.00	very_low
high	medium	very_low	0.00	low
high	medium	very_low	0.00	medium
high	medium	very_low	0.00	high
high	medium	very_low	0.00	very_high
high	medium	low	0.00	very_low
high	medium	low	0.00	low
high	medium	low	0.00	medium
high	medium	low	0.00	high
high	medium	low	0.00	very_high
high	medium	medium	0.00	very_low
high	medium	medium	0.00	low
high	medium	medium	0.00	medium
high	Medium	medium	0.00	high
high	Medium	medium	0.00	very_high
high	Medium	high	0.00	very_low
high	Medium	high	0.00	low
high	Medium	high	0.03	medium
high	Medium	high	0.00	high
high	Medium	high	0.00	very_high
high	Medium	very_high	0.20	very_low
high	Medium	very_high	0.00	low
high	Medium	very_high	0.00	medium
high	Medium	very_high	0.00	high
high	Medium	very_high	0.05	very_high
high	High	very_low	0.00	very_low
high	High	very_low	0.00	low
high	High	very_low	0.00	medium
high	High	very_low	0.00	high
high	High	very_low	0.00	very_high
high	High	low	0.00	very_low
high	High	low	0.00	low
high	High	low	0.00	medium
high	High	low	0.00	high
high	High	low	0.00	very_high

IF			THEN	
high	High	medium	0.00	very_low
high	High	medium	0.00	low
high	High	medium	0.00	medium
high	High	medium	0.00	high
high	High	medium	0.00	very_high
high	High	high	0.49	very_low
high	High	high	0.00	low
high	High	high	0.00	medium
high	High	high	0.01	high
high	High	high	0.00	very_high
high	High	very_high	0.00	very_low
high	High	very_high	0.00	low
high	High	very_high	0.00	medium
high	High	very_high	0.00	high
high	High	very_high	0.00	very_high
high	very_high	very_low	0.00	very_low
high	very_high	very_low	0.00	low
high	very_high	very_low	0.00	medium
high	very_high	very_low	0.00	high
high	very_high	very_low	0.00	very_high
high	very_high	low	0.00	very_low
high	very_high	low	0.00	low
high	very_high	low	0.00	medium
high	very_high	low	0.00	high
high	very_high	low	0.00	very_high
high	very_high	medium	0.00	very_low
high	very_high	medium	0.00	low
high	very_high	medium	0.00	medium
high	very_high	medium	0.00	high
high	very_high	medium	0.09	very_high
high	very_high	high	0.00	very_low
high	very_high	high	0.00	low
high	very_high	high	0.00	medium
high	very_high	high	0.00	high
high	very_high	high	0.00	very_high
high	very_high	very_high	0.00	very_low
high	very_high	very_high	0.00	low
high	very_high	very_high	0.00	medium
high	very_high	very_high	0.00	high
high	very_high	very_high	0.00	very_high
very_high	very_low	very_low	0.00	very_low
very_high	very_low	very_low	0.09	low
very_high	very_low	very_low	0.00	medium

IF			THEN	
very_high	very_low	very_low	0.00	high
very_high	very_low	very_low	0.00	very_high
very_high	very_low	low	0.08	very_low
very_high	very_low	low	0.00	low
very_high	very_low	low	0.02	medium
very_high	very_low	low	0.00	high
very_high	very_low	low	0.00	very_high
very_high	very_low	medium	0.00	very_low
very_high	very_low	medium	0.00	low
very_high	very_low	medium	0.00	medium
very_high	very_low	medium	0.00	high
very_high	very_low	medium	0.12	very_high
very_high	very_low	high	0.00	very_low
very_high	very_low	high	0.00	low
very_high	very_low	high	0.00	medium
very_high	very_low	high	0.00	high
very_high	very_low	high	0.00	very_high
very_high	very_low	very_high	0.00	very_low
very_high	very_low	very_high	0.00	low
very_high	very_low	very_high	0.00	medium
very_high	very_low	very_high	0.00	high
very_high	very_low	very_high	0.00	very_high
very_high	Low	very_low	0.05	very_low
very_high	Low	very_low	0.00	low
very_high	Low	very_low	0.00	medium
very_high	Low	very_low	0.00	high
very_high	Low	very_low	0.00	very_high
very_high	Low	low	0.09	very_low
very_high	Low	low	0.00	low
very_high	Low	low	0.08	medium
very_high	Low	low	0.00	high
very_high	Low	low	0.13	very_high
very_high	Low	medium	0.13	very_low
very_high	Low	medium	0.16	low
very_high	Low	medium	0.06	medium
very_high	Low	medium	0.13	high
very_high	Low	medium	0.21	very_high
very_high	Low	high	0.00	very_low
very_high	Low	high	0.00	low
very_high	Low	high	0.00	medium
very_high	Low	high	0.00	high
very_high	Low	high	0.00	very_high
very_high	Low	very_high	0.00	very_low

IF			THEN	
very_high	Low	very_high	0.00	low
very_high	Low	very_high	0.00	medium
very_high	Low	very_high	0.00	high
very_high	Low	very_high	0.00	very_high
very_high	Medium	very_low	0.00	very_low
very_high	Medium	very_low	0.00	low
very_high	Medium	very_low	0.00	medium
very_high	Medium	very_low	0.00	high
very_high	Medium	very_low	0.00	very_high
very_high	Medium	low	0.32	very_low
very_high	Medium	low	0.00	low
very_high	Medium	low	0.00	medium
very_high	Medium	low	0.00	high
very_high	Medium	low	0.00	very_high
very_high	Medium	medium	0.02	very_low
very_high	Medium	medium	0.00	low
very_high	Medium	medium	0.00	medium
very_high	Medium	medium	0.00	high
very_high	Medium	medium	0.18	very_high
very_high	Medium	high	0.19	very_low
very_high	Medium	high	0.00	low
very_high	Medium	high	0.00	medium
very_high	Medium	high	0.00	high
very_high	Medium	high	0.03	very_high
very_high	Medium	very_high	0.00	very_low
very_high	Medium	very_high	0.07	low
very_high	Medium	very_high	0.00	medium
very_high	Medium	very_high	0.16	high
very_high	Medium	very_high	0.03	very_high
very_high	High	very_low	0.00	very_low
very_high	High	very_low	0.00	low
very_high	High	very_low	0.00	medium
very_high	High	very_low	0.00	high
very_high	High	very_low	0.00	very_high
very_high	High	low	0.00	very_low
very_high	High	low	0.00	low
very_high	High	low	0.00	medium
very_high	High	low	0.00	high
very_high	High	low	0.00	very_high
very_high	High	medium	0.16	very_low
very_high	High	medium	0.00	low
very_high	High	medium	0.00	medium
very_high	High	medium	0.00	high

IF			THEN	
very_high	High	medium	0.31	very_high
very_high	High	high	0.17	very_low
very_high	High	high	0.08	low
very_high	High	high	0.00	medium
very_high	High	high	0.37	high
very_high	High	high	0.03	very_high
very_high	High	very_high	0.73	very_low
very_high	High	very_high	0.06	low
very_high	High	very_high	0.08	medium
very_high	High	very_high	0.01	high
very_high	High	very_high	0.20	very_high
very_high	very_high	very_low	0.00	very_low
very_high	very_high	very_low	0.00	low
very_high	very_high	very_low	0.00	medium
very_high	very_high	very_low	0.00	high
very_high	very_high	very_low	0.00	very_high
very_high	very_high	low	0.00	very_low
very_high	very_high	low	0.00	low
very_high	very_high	low	0.00	medium
very_high	very_high	low	0.00	high
very_high	very_high	low	0.00	very_high
very_high	very_high	medium	0.00	very_low
very_high	very_high	medium	0.00	low
very_high	very_high	medium	0.00	medium
very_high	very_high	medium	0.02	high
very_high	very_high	medium	0.00	very_high
very_high	very_high	high	0.00	very_low
very_high	very_high	high	0.00	low
very_high	very_high	high	0.00	medium
very_high	very_high	high	0.06	high
very_high	very_high	high	0.00	very_high
very_high	very_high	very_high	0.00	very_low
very_high	very_high	very_high	0.00	low
very_high	very_high	very_high	0.00	medium
very_high	very_high	very_high	0.00	high
very_high	very_high	very_high	0.09	very_high

APPENDIX A3

RULE BASE OF THE RAIN FORECAST MODEL (57 RULES)

Aggregation: MINMAX
Parameter: 0.00
Result Aggregation: MAX
Number of Inputs: 3
Number of Outputs: 1
Number of Rules: 57

IF			THEN	
Rel_Hum	**Wind_Dir**	**Wind_Spd**	**DoS**	**Rainfall**
very_low	very_low	high	0.09	very_low
very_low	very_low	very_high	0.02	very_low
very_low	Low	high	0.09	very_low
very_low	High	high	0.10	very_low
Low	very_low	very_low	0.09	very_low
Low	very_low	low	0.06	very_low
Low	very_low	very_high	0.04	Low
Low	High	very_high	0.06	Low
medium	very_low	high	0.02	Low
medium	Medium	high	0.12	very_low
medium	Medium	very_high	0.98	very_low
medium	Medium	very_high	0.01	High
medium	High	very_high	0.17	very_low
medium	High	very_high	0.06	High
medium	very_high	high	0.09	Low
High	very_low	very_low	0.06	very_low
High	very_low	very_low	0.02	High
High	very_low	low	0.42	very_low
High	very_low	low	0.12	Medium
High	Medium	high	0.90	Low
High	High	high	0.74	very_low
High	High	high	0.05	very_high
very_high	very_low	very_low	0.11	very_low
very_high	very_low	very_low	0.10	High
very_high	very_low	low	0.89	very_low
very_high	very_low	low	0.05	Medium
very_high	very_low	low	0.27	High
very_high	very_low	low	0.02	very_high
very_high	very_low	medium	0.31	very_high
very_high	Low	low	0.80	very_low

IF			THEN	
very_high	Low	low	0.10	Low
very_high	Low	low	0.80	Medium
very_high	Low	low	0.01	very_high
very_high	Low	medium	0.43	very_low
very_high	Low	medium	0.05	Low
very_high	Low	medium	0.90	High
very_high	Low	medium	0.60	very_high
very_high	Medium	low	0.40	very_low
very_high	Medium	low	0.10	Low
very_high	Medium	medium	0.98	High
very_high	Medium	medium	0.90	very_high
very_high	Medium	high	0.20	very_low
very_high	Medium	high	0.40	Low
very_high	Medium	very_high	0.38	very_low
very_high	Medium	very_high	0.09	High
very_high	Medium	very_high	0.27	very_high
very_high	High	medium	0.09	very_low
very_high	High	medium	0.05	High
very_high	High	medium	0.18	very_high
very_high	High	high	0.94	very_low
very_high	High	high	0.01	Medium
very_high	High	high	0.01	very_high
very_high	High	very_high	0.75	very_low
very_high	High	very_high	0.18	Low
very_high	High	very_high	0.02	Medium
very_high	High	very_high	0.09	High
very_high	High	very_high	0.24	very_high

APPENDIX A4

COMPLETE PROGRAM LISTING

```
PROJECT {
        NAME= RFM_DHP_v3.FTL;
        TITLE = Rain_Forecast_Model;
        DATEFORMAT = DD.MM.YYYY;
        LASTCHANGE = 20.11.2005;
        CREATED = 05.07.2005;
SHELLOPTIONS {
        COMMENTS = ON;
        FTL_BUFFER = (OFF, PAR(1));
        PASSWORD = OFF;
        PUBLIC_IO = ON;
        FAST_CMBF = OFF;
        FAST_COA = ON;
        BTYPE = DOUBLE;
        C_TYPE  = ANSI;
} /* SHELLOPTIONS */
MODEL {
VARIABLE_SECTION {
LVAR {
        /*This is used to define one linguistic variable at a time*/
        NAME = Rel_Hum;
        BASEVAR = percent;
        LVRANGE =  MIN(41.2), MAX(90.6), MINDEF(0), MAXDEF(65535),
                        DEFAULT_OUTPUT(65.9);
                        RESOLUTION = XGRID(1.0), YGRID(1.0),SHOWGRID (ON),
                        SNAPTOGRID(ON);
                        COLOR = RED (255), GREEN (0), BLUE (0);
                        INPUT = CMBF;
                        POS = -225, -35;
                TERM {
                /*Each term object has one term and membership function definition
                */
                TERMNAME = very_low;
                POINTS = (41.2, 1.0), (49.933, 1.0), (57.667, 0.0), (90.6, 0.0);
                /*The slot 'points' defines piecewise linear membership function as (x,y)
                co-ordinates */
                SHAPE = LINEAR;
                COLOR = RED (100), GREEN (0), BLUE (0);
                }
                TERM {
                TERMNAME = low;
```

POINTS = (41.2, 0.0), (49.933, 0.0), (57.667, 1.0), (65.9, 0.0), (90.6, 0.0);
SHAPE = LINEAR;
COLOR = RED (255), GREEN (0), BLUE (0);
}
TERM {
TERMNAME = medium;
POINTS = (41.2, 0.0), (57.667, 0.0), (65.9, 1.0), (74.133, 0.0), (90.6, 0.0);
SHAPE = LINEAR;
COLOR = RED (0), GREEN (100), BLUE (0);
}
TERM {
TERMNAME = high;
POINTS = (41.2, 0.0), (65.9, 0.0), (74.133, 1.0), (82.367, 0.0), (90.6, 0.0);
SHAPE = LINEAR;
COLOR = RED (0), GREEN (0), BLUE (100);
}
TERM {
TERMNAME = very_high;
POINTS = (41.2, 0.0), (74.133, 0.0), (82.367, 1.0), (90.6, 1.0);
SHAPE = LINEAR;
COLOR = RED (0), GREEN (0), BLUE (255);
}
}
LVAR {
NAME = Wind_Dir;
BASEVAR = degree_SW;
LVRANGE = MIN(10.0),MAX(129.0),MINDEF(0),MAXDEF(65535),
DEFAULT_OUTPUT(69.5);
RESOLUTION = XGRID(0.3), YGRID(1.0), SHOWGRID (ON),
SNAPTOGRID(ON);
COLOR = RED (0), GREEN (255), BLUE (0);
INPUT = CMBF;
POS = -225, 30;
TERM {
TERMNAME = very_low;
POINTS = (10.0, 1.0), (29.834, 1.0), (49.666, 0.0), (129.0, 0.0);
SHAPE = LINEAR;
COLOR = RED (100), GREEN (0), BLUE (0);
}
TERM {
TERMNAME = low;
POINTS = (10.0, 0.0),(29.834, 0.0), (49.666, 1.0), (65.9, 0.0), (129.0, 0.0);
SHAPE = LINEAR;
COLOR = RED (255), GREEN (0), BLUE (0);
}
TERM {

```
        TERMNAME = medium;
        POINTS =(10.0, 0.0), (49.666, 0.0), (65.9, 1.0), (89.334, 0.0), (129.0, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (0), GREEN (100), BLUE (0);
        }
        TERM {
        TERMNAME = high;
        POINTS=(10.0, 0.0),(65.9, 0.0), (89.334, 1.0), (109.166, 0.0), (129.0, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (0), GREEN (0), BLUE (100);
        }
        TERM {
        TERMNAME = very_high;
        POINTS = (10.0, 0.0), (89.334, 0.0), (109.166, 1.0), (129.0, 1.0);
        SHAPE = LINEAR;
        COLOR = RED (0), GREEN (0), BLUE (255);
        }
        }
LVAR {
NAME = Wind_Spd;
BASEVAR = km_hr;
LVRANGE =  MIN(46.06),MAX(252.16),MINDEF(0), MAXDEF(65535),
                DEFAULT_OUTPUT(149.11);
                RESOLUTION = XGRID(0.6), YGRID(1.0),SHOWGRID (ON),
                SNAPTOGRID(ON);
                COLOR = RED (0), GREEN (0), BLUE (255);
                INPUT = CMBF;
                POS = -225, 100;
        TERM {
        TERMNAME = very_low;
        POINTS = (46.06, 1.0), (80.41, 1.0), (114.76, 0.0), (252.16, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (100), GREEN (0), BLUE (0);
        }
        TERM {
        TERMNAME = low;
        POINTS=(46.06, 0.0),(80.41, 0.0),(114.76,1.0),(149.11, 0.0),(252.16, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (255), GREEN (0), BLUE (0);
        }
        TERM {
        TERMNAME = medium;
        POINTS=(46.06,0.0),(114.76,0.0),(149.11,1.0),(183.46, 0.0),(252.16, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (0), GREEN (100), BLUE (0);
        }
```

```
        TERM {
        TERMNAME = high;
        POINTS=(46.06,0.0),(149.11,0.0),(183.46,1.0),(217.81,0.0), (252.16, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (0), GREEN (0), BLUE (100);
        }
        TERM {
        TERMNAME = very_high;
        POINTS = (46.06, 0.0), (183.46, 0.0), (217.81, 1.0), (252.16, 1.0);
        SHAPE = LINEAR;
        COLOR = RED (0), GREEN (0), BLUE (255);
        }
        }
LVAR {
NAME = Rainfall;
BASEVAR = mm;
LVRANGE =  MIN(-107.15),MAX(589.15),MINDEF(0),MAXDEF(65535),
                DEFAULT_OUTPUT(241.0);
                RESOLUTION = XGRID(0.25), YGRID(1.0), SHOWGRID(ON),
                SNAPTOGRID(ON);
                COLOR = RED (0), GREEN (0), BLUE (255);
                OUTPUT = COM;
                POS = 105, 0;
        TERM {
        TERMNAME = very_low;
        POINTS = (-107.15, 1.0), (8.9, 1.0), (124.95, 0.0), (589.15, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (100), GREEN (0), BLUE (0);
        }
        TERM {
        TERMNAME = low;
        POINTS=(-107.15,0.0),(8.9,0.0), (124.95, 1.0), (241.0, 0.0), (589.15, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (255), GREEN (0), BLUE (0);
        }
        TERM {
        TERMNAME = medium;
        POINTS=(-107.15,0.0),(124.95,0.0),(241.0,1.0),(357.05,0.0),(589.15,0.0);
        SHAPE = LINEAR;
        COLOR = RED (0), GREEN (100), BLUE (0);
        }
        TERM {
        TERMNAME = high;
        POINTS=(-107.15,0.0),(241.0,0.0),(357.05,1.0),(473.1,0.0), (589.15, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (0), GREEN (0), BLUE (100);
```

```
        }
        TERM {
        TERMNAME = very_high;
        POINTS = (-107.15, 0.0), (357.05, 0.0), (473.1, 1.0), (589.15, 0.0);
        SHAPE = LINEAR;
        COLOR = RED (0), GREEN (0), BLUE (255);
        }
        }
}/*VARIABLE SECTION*/

OBJECT_SECTION {
/*The object section contains all structural elements of a fuzzy logic system, such as
interfaces and rule blocks */
        REMARK {
        TEXT = RAIN FORECAST MODEL;
        POS = -175, -135;
        FONTSPEC = 50, 0, 0, 0, 0, 16, 0;
        FONTNAME =Times New Roman;
        COLOR = RED (200), GREEN (10), BLUE (10);
        }
        RULEBLOCK {
        /* A rule block object contains the definition of its input and output variable as
          well as its rules */
        NAME = RFM_Rule_Block;
        INPUT = Rel_Hum, Wind_Dir, Wind_Spd;
        OUTPUT = Rainfall;
        AGGREGATION = (MIN_MAX, PAR(0.0));
        RESULT_AGGR = MAX;
        POS        = -85, -7;
        RULES {
        IF Rel_Hum=very_low AND Wind_Dir=very_low AND Wind_Spd=high
                                        THEN Rainfall = very_low WITH 0.090;
        IF Rel_Hum=very_low AND Wind_Dir = very_low AND Wind_Spd = very_high
                                        THEN Rainfall = very_low WITH 0.020;
        IF Rel_Hum = very_low AND Wind_Dir = low AND Wind_Spd = high
                                        THEN Rainfall = very_low WITH 0.090;
        IF Rel_Hum = very_low AND Wind_Dir = high AND Wind_Spd = high
                                        THEN Rainfall = very_low WITH 0.100;
        IF Rel_Hum = low AND Wind_Dir = very_low AND Wind_Spd = very_low
                                        THEN Rainfall = very_low WITH 0.090;
        IF Rel_Hum = low AND Wind_Dir = very_low AND Wind_Spd = low
                                        THEN Rainfall = very_low WITH 0.060;
        IF Rel_Hum = low AND Wind_Dir = very_low AND Wind_Spd = very_high
                                        THEN Rainfall = low WITH 0.040;
        IF Rel_Hum = low AND Wind_Dir = high AND Wind_Spd = very_high
```

THEN Rainfall = low WITH 0.060;
IF Rel_Hum = medium AND Wind_Dir = very_low AND Wind_Spd = high
THEN Rainfall = low WITH 0.020;
IF Rel_Hum = medium AND Wind_Dir = medium AND Wind_Spd = high
THEN Rainfall = very_low WITH 0.120;
IF Rel_Hum = medium AND Wind_Dir = medium AND Wind_Spd = very_high
THEN Rainfall = very_low WITH 0.980;
IF Rel_Hum = medium AND Wind_Dir = medium AND Wind_Spd = very_high
THEN Rainfall = high WITH 0.010;
IF Rel_Hum = medium AND Wind_Dir = high AND Wind_Spd = very_high
THEN Rainfall = very_low WITH 0.170;
IF Rel_Hum = medium AND Wind_Dir = high AND Wind_Spd = very_high
THEN Rainfall = high WITH 0.060;
IF Rel_Hum = medium AND Wind_Dir = very_high AND Wind_Spd = high
THEN Rainfall = low WITH 0.090;
IF Rel_Hum = high AND Wind_Dir = very_low AND Wind_Spd = very_low
THEN Rainfall = very_low WITH 0.060;
IF Rel_Hum = high AND Wind_Dir = very_low AND Wind_Spd = very_low
THEN Rainfall = high WITH 0.020;
IF Rel_Hum = high AND Wind_Dir = very_low AND Wind_Spd = low
THEN Rainfall = very_low WITH 0.420;
IF Rel_Hum = high AND Wind_Dir = very_low AND Wind_Spd = low
THEN Rainfall = medium WITH 0.120;
IF Rel_Hum = high AND Wind_Dir = medium AND Wind_Spd = high
THEN Rainfall = low WITH 0.900;
IF Rel_Hum = high AND Wind_Dir = high AND Wind_Spd = high
THEN Rainfall = very_low WITH 0.740;
IF Rel_Hum = high AND Wind_Dir = high AND Wind_Spd = high
THEN Rainfall = very_high WITH 0.050;
IF Rel_Hum=very_high AND Wind_Dir = very_low AND Wind_Spd = very_low
THEN Rainfall = very_low WITH 0.110;
IF Rel_Hum=very_high AND Wind_Dir = very_low AND Wind_Spd = very_low
THEN Rainfall = high WITH 0.100;
IF Rel_Hum = very_high AND Wind_Dir = very_low AND Wind_Spd = low
THEN Rainfall = very_low WITH 0.890;
IF Rel_Hum = very_high AND Wind_Dir = very_low AND Wind_Spd = low
THEN Rainfall = medium WITH 0.050;
IF Rel_Hum = very_high AND Wind_Dir = very_low AND Wind_Spd = low
THEN Rainfall = high WITH 0.270;
IF Rel_Hum = very_high AND Wind_Dir = very_low AND Wind_Spd = low
THEN Rainfall = very_high WITH 0.020;
IF Rel_Hum = very_high AND Wind_Dir = very_low AND Wind_Spd = medium
THEN Rainfall = very_high WITH 0.310;
IF Rel_Hum = very_high AND Wind_Dir = low AND Wind_Spd = low
THEN Rainfall = very_low WITH 0.800;
IF Rel_Hum = very_high AND Wind_Dir = low AND Wind_Spd = low

THEN Rainfall = low WITH 0.100;
IF Rel_Hum = very_high AND Wind_Dir = low AND Wind_Spd = low
THEN Rainfall = medium WITH 0.800;
IF Rel_Hum = very_high AND Wind_Dir = low AND Wind_Spd = low
THEN Rainfall = very_high WITH 0.010;
IF Rel_Hum = very_high AND Wind_Dir = low AND Wind_Spd = medium
THEN Rainfall = very_low WITH 0.430;
IF Rel_Hum = very_high AND Wind_Dir = low AND Wind_Spd = medium
THEN Rainfall = low WITH 0.050;
IF Rel_Hum = very_high AND Wind_Dir = low AND Wind_Spd = medium
THEN Rainfall = high WITH 0.900;
IF Rel_Hum = very_high AND Wind_Dir = low AND Wind_Spd = medium
THEN Rainfall = very_high WITH 0.600;
IF Rel_Hum = very_high AND Wind_Dir = medium AND Wind_Spd = low
THEN Rainfall = very_low WITH 0.400;
IF Rel_Hum = very_high AND Wind_Dir = medium AND Wind_Spd = low
THEN Rainfall = low WITH 0.100;
IF Rel_Hum = very_high AND Wind_Dir = medium AND Wind_Spd = medium
THEN Rainfall = high WITH 0.980;
IF Rel_Hum = very_high AND Wind_Dir = medium AND Wind_Spd = medium
THEN Rainfall = very_high WITH 0.900;
IF Rel_Hum = very_high AND Wind_Dir = medium AND Wind_Spd = high
THEN Rainfall = very_low WITH 0.200;
IF Rel_Hum = very_high AND Wind_Dir = medium AND Wind_Spd = high
THEN Rainfall = low WITH 0.400;
IF Rel_Hum=very_high AND Wind_Dir = medium AND Wind_Spd = very_high
THEN Rainfall = very_low WITH 0.380;
IF Rel_Hum=very_high AND Wind_Dir = medium AND Wind_Spd = very_high
THEN Rainfall = high WITH 0.090;
IF Rel_Hum=very_high AND Wind_Dir = medium AND Wind_Spd = very_high
THEN Rainfall = very_high WITH 0.270;
IF Rel_Hum = very_high AND Wind_Dir = high AND Wind_Spd = medium
THEN Rainfall = very_low WITH 0.090;
IF Rel_Hum = very_high AND Wind_Dir = high AND Wind_Spd = medium
THEN Rainfall = high WITH 0.050;
IF Rel_Hum = very_high AND Wind_Dir = high AND Wind_Spd = medium
THEN Rainfall = very_high WITH 0.180;
IF Rel_Hum = very_high AND Wind_Dir = high AND Wind_Spd = high
THEN Rainfall = very_low WITH 0.940;
IF Rel_Hum = very_high AND Wind_Dir = high AND Wind_Spd = high
THEN Rainfall = medium WITH 0.010;
IF Rel_Hum = very_high AND Wind_Dir = high AND Wind_Spd = high
THEN Rainfall = very_high WITH 0.010;
IF Rel_Hum = very_high AND Wind_Dir = high AND Wind_Spd = very_high
THEN Rainfall = very_low WITH 0.750;
IF Rel_Hum = very_high AND Wind_Dir = high AND Wind_Spd = very_high

```
                                        THEN Rainfall = low WITH 0.180;
        IF Rel_Hum = very_high AND Wind_Dir = high AND Wind_Spd = very_high
                                        THEN Rainfall = medium WITH 0.020;
        IF Rel_Hum=very_high AND Wind_Dir=high AND Wind_Spd= very_high
                                        THEN Rainfall = high WITH 0.090;
        IF Rel_Hum=very_high AND Wind_Dir=high AND Wind_Spd= very_high
                                        THEN Rainfall = very_high WITH 0.240;
            }
            }
        REMARK {
        TEXT = Rule Block;
        POS = -40, -70;
        FONTSPEC = 24, 0, 0, 0, 0, 16, 0;
        FONTNAME =Times New Roman;
        COLOR = RED (200), GREEN (200), BLUE (0);
        }
        REMARK {
        TEXT = Input Interfaces;
        POS = -205, -70;
        FONTSPEC = 24, 0, 0, 0, 0, 16, 0;
        FONTNAME =Times New Roman;
        COLOR = RED (0), GREEN (0), BLUE (200);
        }
        REMARK {
        TEXT = Output Interface;
        POS = 90, -70;
        FONTSPEC = 24, 0, 0, 0, 0, 16, 0;
        FONTNAME =Times New Roman;
        COLOR = RED (0), GREEN (80), BLUE (200);
        }
    } /* OBJECT_SECTION */
} /* MODEL */
} /* PROJECT */
```

APPENDIX A5

FORECAST RESULTS OF MOVING AVERAGES AND EXPONENTIAL SMOOTHING METHODS

WIND_SPD FORECAST TABLE BY 9-YEAR MOVING AVERAGES AND EXPONENTIAL SMOOTHING METHODS (α=0.286)														
	APRIL		**MAY**		**JUNE**		**JULY**		**AUGUST**		**SEPTEMBER**		**OCTOBER**	
1993	200.3	**200.3**	235.2	**235.2**	210.6	**210.6**	179.7	**179.7**	163.7	**163.7**	129.61	**129.6**	107.88	**107.9**
1994	222.73	**206.71**	185.03	**220.87**	230.44	**216.27**	168.47	**176.47**	149.31	**159.58**	97.02	**120.29**	46.06	**90.20**
1995	197.30	**204.02**	185.30	**210.69**	211.78	**214.99**	179.43	**177.32**	138.25	**153.48**	115.88	**119.03**	111.43	**96.27**
1996	175.53	**195.87**	244.61	**220.39**	218.16	**215.90**	181.34	**178.47**	164.86	**156.74**	146.60	**126.91**	142.36	**109.45**
1997	223.27	**203.71**	195.75	**213.35**	196.57	**210.37**	180.38	**179.02**	146.60	**153.84**	124.33	**126.17**	125.09	**113.93**
1998	190.24	**199.86**	252.16	**224.45**	180.70	**201.88**	176.49	**178.29**	151.63	**153.21**	131.14	**127.59**	130.32	**118.61**
1999	219.12	**205.37**	201.16	**217.79**	180.70	**195.82**	168.22	**175.41**	151.63	**152.76**	126.56	**127.30**	108.39	**115.69**
2000	219.12	**209.30**	202.24	**213.34**	208.48	**199.44**	181.08	**177.03**	154.87	**153.36**	149.44	**133.63**	123.39	**117.89**
2001	230.37	**215.33**	201.82	**210.05**	206.36	**201.42**	181.23	**178.23**	153.00	**153.26**	100.77	**124.23**	114.87	**117.03**
2002	209.10	**213.55**	213.53	**211.04**	201.28	**201.38**	180.05	**178.75**	132.77	**147.40**	102.60	**118.05**	81.80	**106.95**
2003	**209.64**	**212.43**	**209.07**	**210.48**	**203.83**	**202.08**	**177.41**	**178.37**	**149.21**	**147.92**	**121.59**	**119.06**	**109.30**	**107.62**
2004	**208.19**	**211.22**	**211.74**	**210.84**	**200.87**	**201.74**	**178.40**	**178.38**	**149.20**	**148.28**	**124.32**	**120.57**	**116.33**	**110.11**
2005	**209.40**	**210.70**	**214.67**	**211.94**	**199.66**	**201.14**	**178.29**	**178.35**	**150.42**	**148.90**	**125.26**	**121.91**	**116.87**	**112.05**
2006	**213.16**	**211.40**	**211.35**	**211.77**	**197.61**	**200.13**	**177.95**	**178.24**	**148.82**	**148.87**	**122.89**	**122.19**	**114.04**	**112.62**
2007	**212.04**	**211.58**	**213.08**	**212.14**	**197.72**	**199.44**	**177.68**	**178.08**	**149.06**	**148.93**	**122.73**	**122.34**	**112.81**	**112.67**
2008	**214.46**	**212.41**	**208.74**	**211.17**	**199.61**	**199.49**	**177.81**	**178.00**	**148.78**	**148.88**	**121.80**	**122.19**	**110.87**	**112.16**
2009	**213.94**	**212.85**	**209.58**	**210.72**	**201.71**	**200.13**	**178.88**	**178.25**	**148.46**	**148.76**	**121.27**	**121.92**	**111.14**	**111.87**
2010	**213.37**	**212.99**	**210.40**	**210.62**	**200.96**	**200.37**	**178.63**	**178.36**	**147.75**	**148.47**	**118.14**	**120.84**	**109.78**	**111.27**

WIND_DIR FORECAST TABLE BY 9-YEAR MOVING AVERAGES AND EXPONENTIAL SMOOTHING METHODS (α=0.286)														
	APRIL		**MAY**		**JUNE**		**JULY**		**AUGUST**		**SEPTEMBER**		**OCTOBER**	
1993	29	**29**	32	**32**	129	**129**	104	**104**	102	**102**	65	**65**	26	**26**
1994	42	**33**	57	**39**	83	**116**	66	**92**	93	**73**	51	**61**	28	**27**
1995	13	**27**	24	**35**	31	**92**	33	**76**	10	**39**	20	**49**	10	**22**
1996	32	**28**	82	**48**	80	**88**	64	**25**	63	**32**	52	**50**	24	**22**
1997	49	**34**	64	**53**	80	**86**	65	**64**	48	**51**	33	**45**	33	**25**
1998	98	**53**	73	**59**	65	**80**	96	**62**	72	**44**	56	**48**	33	**28**
1999	42	**50**	66	**61**	49	**71**	77	**73**	79	**63**	47	**48**	27	**27**
2000	53	**51**	76	**65**	77	**73**	87	**81**	83	**57**	60	**51**	35	**30**
2001	73	**57**	77	**68**	98	**80**	83	**83**	80	**66**	41	**48**	12	**25**
2002	70	**61**	97	**77**	93	**84**	96	**85**	67	**48**	51	**49**	33	**27**
2003	**52**	**58**	**68**	**74**	**73**	**81**	**74**	**69**	**66**	**55**	**46**	**48**	**26**	**27**
2004	**54**	**57**	**70**	**73**	**72**	**78**	**75**	**69**	**63**	**51**	**45**	**47**	**26**	**26**
2005	**58**	**57**	**75**	**74**	**76**	**78**	**80**	**68**	**69**	**52**	**48**	**47**	**28**	**27**
2006	**61**	**58**	**74**	**74**	**76**	**77**	**81**	**73**	**70**	**54**	**47**	**47**	**28**	**27**
2007	**62**	**60**	**75**	**74**	**75**	**77**	**83**	**74**	**72**	**54**	**49**	**48**	**28**	**27**
2008	**58**	**59**	**75**	**74**	**77**	**77**	**82**	**75**	**72**	**56**	**48**	**48**	**27**	**27**
2009	**60**	**59**	**76**	**75**	**80**	**77**	**82**	**75**	**71**	**55**	**48**	**48**	**27**	**27**
2010	**61**	**60**	**76**	**75**	**80**	**78**	**82**	**74**	**70**	**55**	**47**	**48**	**26**	**27**

REL_HUM FORECAST TABLE BY 9-YEAR MOVING AVERAGES AND EXPONENTIAL SMOOTHING METHODS (α=0.286)														
	APRIL		**MAY**		**JUNE**		**JULY**		**AUGUST**		**SEPTEMBER**		**OCTOBER**	
1993	42.7	**42.7**	62.9	**62.9**	72.7	**72.7**	78.4	**78.4**	81.8	**81.8**	76.7	**76.7**	63.1	**63.1**
1994	49	**44.5**	71.7	**65.4**	77.4	**74.0**	84.3	**80.1**	85	**82.7**	82.8	**78.4**	76.4	**66.9**
1995	41.2	**43.6**	68	**66.2**	70.6	**73.1**	79.5	**79.9**	81.8	**82.5**	81.3	**79.3**	74.3	**69.0**
1996	55.8	**47.1**	69.8	**67.2**	77.5	**74.3**	79.4	**79.8**	82	**82.3**	81.5	**79.9**	65.5	**68.0**
1997	57.2	**50.0**	74.9	**69.4**	79.6	**75.8**	84.5	**81.1**	83.4	**82.6**	80.8	**80.2**	78.6	**71.0**
1998	45	**48.5**	76.7	**71.5**	82.4	**77.7**	84.3	**82.0**	87.6	**84.1**	86.1	**81.9**	78.8	**73.3**
1999	42	**46.7**	61.2	**68.5**	78.4	**77.9**	87.1	**83.5**	89.1	**85.5**	86.1	**83.1**	75.4	**73.9**
2000	53	**48.5**	64.1	**67.3**	82.1	**79.1**	85.9	**84.2**	87.9	**86.2**	82.4	**82.9**	71.2	**73.1**
2001	52.4	**49.6**	75.7	**69.7**	81.2	**79.7**	87.2	**85.0**	89.9	**87.2**	86.7	**84.0**	63	**70.2**
2002	68	**54.9**	79.4	**72.5**	85.7	**81.4**	90.1	**86.5**	90.6	**88.2**	88.7	**85.3**	86.6	**74.9**
2003	**51.5**	**53.9**	**71.3**	**72.1**	**79.4**	**80.9**	**84.7**	**86.0**	**86.4**	**87.7**	**84.0**	**85.0**	**74.4**	**74.8**
2004	**51.8**	**53.3**	**71.2**	**71.9**	**79.7**	**80.5**	**84.7**	**85.6**	**86.5**	**87.3**	**84.2**	**84.7**	**74.2**	**74.6**
2005	**53.0**	**53.2**	**71.6**	**71.8**	**80.7**	**80.6**	**85.3**	**85.5**	**87.0**	**87.3**	**84.5**	**84.7**	**74.2**	**74.5**
2006	**52.7**	**53.0**	**71.8**	**71.8**	**81.0**	**80.7**	**86.0**	**85.7**	**87.6**	**87.4**	**84.8**	**84.7**	**75.2**	**74.7**
2007	**52.1**	**52.8**	**71.4**	**71.7**	**81.2**	**80.8**	**86.2**	**85.8**	**88.1**	**87.6**	**85.3**	**84.9**	**74.8**	**74.7**
2008	**52.9**	**52.8**	**70.9**	**71.5**	**81.0**	**80.9**	**86.4**	**86.0**	**88.1**	**87.7**	**85.2**	**85.0**	**74.3**	**74.6**
2009	**54.2**	**53.2**	**71.9**	**71.6**	**81.3**	**81.0**	**86.3**	**86.1**	**88.0**	**87.8**	**85.1**	**85.0**	**74.2**	**74.5**
2010	**54.3**	**53.5**	**72.8**	**71.9**	**81.2**	**81.1**	**86.3**	**86.1**	**88.0**	**87.9**	**85.4**	**85.1**	**74.5**	**74.5**

APPENDIX A6

THE COMPLETE FORECASTED RAINFALL DATA OBTAINED FROM THE RAIN FORECAST MODEL (RFM)

YEAR	MONTH	FORECAST_REL_HUM	FORECAST_WIND_DIR	FORECAST_WIND_SPD	FORECASTED_RAINFALL
2003	APRIL	53.9	58	212.4	9
	MAY	72.1	74	210.5	80.3
	JUNE	80.9	81	202.1	136.6
	JULY	86	69	178.4	203
	AUGUST	87.7	55	147.9	309.4
	SEPTEMBER	85	48	119.1	157.3
	OCTOBER	74.8	27	107.6	79.2
2004	APRIL	53.3	57	211.2	9
	MAY	71.9	73	210.8	76.5
	JUNE	80.5	78	201.7	134.3
	JULY	85.6	69	178.4	203
	AUGUST	87.3	51	148.3	310.2
	SEPTEMBER	84.7	47	120.6	168.3
	OCTOBER	74.6	26	110.1	73.2
2005	APRIL	53.2	57	210.7	9
	MAY	71.8	74	211.9	70.7
	JUNE	80.6	78	201.1	131.8
	JULY	85.5	68	178.4	203
	AUGUST	87.2	52	148.9	310.7
	SEPTEMBER	84.7	47	121.9	178
	OCTOBER	74.5	27	112.1	70.2
2006	APRIL	53	58	211.4	9
	MAY	71.8	74	211.8	71.1
	JUNE	80.7	77	200.1	132.8
	JULY	85.7	73	178.2	186.6
	AUGUST	87.4	54	148.9	310.6
	SEPTEMBER	84.7	47	122.2	180.2

YEAR	M0NTH	FORECAST_REL_HUM	FORECAST_WIND_DIR	FORECAST_WIND_SPD	FORECASTED_RAINFALL
2006	OCTOBER	**74.7**	**27**	**112.6**	**74.7**
2007	APRIL	**52.8**	**60**	**211.6**	**9**
	MAY	**71.7**	**74**	**212.1**	**68.6**
	JUNE	**80.8**	**77**	**199.4**	**129.9**
	JULY	**85.8**	**74**	**178.1**	**181.2**
	AUGUST	**87.6**	**54**	**148.9**	**310.6**
	SEPTEMBER	**84.9**	**48**	**122.3**	**180.9**
	OCTOBER	**74.7**	**27**	**112.7**	**72.8**
2008	APRIL	**52.8**	**59**	**212.4**	**9**
	MAY	**71.5**	**74**	**211.2**	**70.3**
	JUNE	**80.9**	**77**	**199.5**	**130.4**
	JULY	**86**	**75**	**178**	**176.1**
	AUGUST	**87.7**	**56**	**148.9**	**310.5**
	SEPTEMBER	**85**	**48**	**122.2**	**180.2**
	OCTOBER	**74.6**	**27**	**112.2**	**72.4**
2009	APRIL	**53.2**	**59**	**212.9**	**9**
	MAY	**71.6**	**75**	**210.7**	**73.2**
	JUNE	**81**	**77**	**200.1**	**132.8**
	JULY	**86.1**	**75**	**178.3**	**171.8**
	AUGUST	**87.8**	**55**	**148.8**	**310.4**
	SEPTEMBER	**85**	**48**	**121.9**	**178**
	OCTOBER	**74.5**	**27**	**111.9**	**70.3**
2010	APRIL	**53.5**	**60**	**213**	**9**
	MAY	**71.9**	**75**	**210.6**	**77.2**
	JUNE	**81.1**	**78**	**200.4**	**128.5**
	JULY	**86.1**	**74**	**178.4**	**176.8**
	AUGUST	**87.9**	**55**	**148.5**	**310.1**
	SEPTEMBER	**85.1**	**48**	**120.8**	**169.8**
	OCTOBER	**74.5**	**27**	**111.3**	**70.5**

APPENDIX A7

FUZZY TIME SERIES DATA

MONTHS	u_m^1	u_m^2	u_m^3	u_m^4	u_m^5
APRIL	-24.13	-12.99	1.85	9.29	20.43
MAY	-43.58	-23.54	-3.5	16.54	36.58
JUNE	-56.61	-25.03	6.55	38.13	69.71
JULY	-60.91	-33.73	-6.55	20.63	47.81
AUGUST	-58.81	-14.06	15.3	44.66	74.02
OCTOBER	-10.53	-2.19	6.15	14.49	22.83