A Comparison of Fuzzy Logic Models for Gain Scheduling In Load Frequency Control

A Thesis

Submitted in the partial fulfillment of requirements
for the award of the degree of

Master of Engineering

in

Electronic Instrumentation and Control Engineering

Submitted
by

MUNISH PARMAR
Regn.No-80651013

Under the esteemed guidance of:
Dr. YADUVIR SINGH
Associate Professor

Department of Electrical and Instrumentation Engineering
THAPAR UNIVERSITY
PATIALA (PUNJAB)-147004
July – 2008

CERTIFICATE

This is to certify that the thesis titled, "**A Comparison of Fuzzy Logic Models for Gain Scheduling In Load Frequency Control"**, being submitted by Munish Parmar, Regn. No.: 80651013, in partial fulfillment for the requirements of the award of the degree of Master of Engineering in Thapar University, Patiala, is a record of student's own work carried out under my supervision and guidance and this thesis has not been submitted to any other university or institute for award of any degree.

Date: **Munish Parmar**

Regn No. 80651013

It is certified that the above statement made by the student is correct to the best of our knowledge and belief.

Mentor & Supervisor

Dr. Yaduvir Singh
Associate Professor EIED,
Thapar University
Patiala

ConutersignedBy:

Dr. Smarajit Ghosh
Professor and Head
EIED, Thapar University
Patiala

Dr. R.K.Sharma
Dean of Academic affairs
Thapar University
Patiala

DEDICATED TO MY PARENTS

ACKNOWLEDGEMENT

Words are often too less to reveals one's deep regards. An understanding of the work like this is never the outcome of the efforts of a single person. I take this opportunity to express my profound sense of gratitude and respect to all those who helped me through the duration of this thesis.

First of all I would like to thank the **Supreme Power,** one who has always guided me to work on the right path of the life. Without His grace this would never come to be today's reality.

This work would not have been possible without the encouragement and able guidance of my supervisor, **Dr. Yaduvir Singh.** Their enthusiasm and optimism made this experience both rewarding and enjoyable. Most of the novel ideas and solutions found in this thesis are the result of our numerous stimulating discussions. Their feedback and editorial comments were also invaluable for the writing of this thesis.

*No words of thanks are enough for my **dear parents** whose support and care makes me stay on earth. Thanks to be with me.*

At the end, I would like to thank all the faculty members of the department and my friends who directly or indirectly helped me in completion of my thesis.

Munish Parmar

(Regn No. 80651013)

ABSTRACT

Fuzzy logic solves the problem of non linear systems and handles them with great efficiency and provides robustness to the system. However our aim lies in achieving the adaptive Fuzzy logic load frequency control model for gain scheduling. In this thesis the recent data based artificially intelligent techniques like fuzzy and neural network have been customized and used .The application/case study has been taken from a research paper which appeared in a reputed conference. Fuzzy provides a robust inference mechanism with no learning and adaptability and artificial neural network provides learning and adaptability. Artificial neural networks and fuzzy systems have been successfully applied to the LFC problem with rather promising results. The salient feature of these techniques is that they provide a model-free description of control systems and do not require model identification. In this thesis, an adaptive fuzzy gain scheduling scheme for conventional PI and optimal controllers has been simulated and tested for off-nominal operating conditions. From the simulation and the result obtained in this thesis it has been shown that the proposed adaptive fuzzy logic controller offers better performance than fixed gain controllers, to guarantee that the fuel cell is protected by maintaining its cell utilization within its admissible range and 2) to track load changes and regulate the frequency. The two respective loops are called primary and secondary loops. The primary loop maintains constant the ratio of stack current to input fuel flow, and the secondary loop tracks the load and regulates the frequency. A distribution area error (DSE) is introduced to formulate the frequency-control problem. The secondary loop feeds back this DSE signal. Tuning of the parameters is performed using genetic algorithms.

This thesis presents summaries of novel approaches of artificial intelligence (AI) techniques, like fuzzy logic, hybrid fuzzy neural network (HFNN) for the load frequency control of electrical power system. The limitations of conventional controls such as proportional, integral and derivative are slow and lack of efficiency in handling system nonlinearities. Since high frequency deviation may lead to system collapse, this necessitates an accurate and fast acting controller to maintain the constant nominal

system frequency. The intelligent controllers are used for load frequency control for the single area system, multi area interconnected system. The performance of intelligent controllers with the conventional controllers has been thoroughly compared and analyzed

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

FL	Fuzzy Logic
MF	Membership Function
FS	Fuzzy Sets
LFS	Load Frequency Control
HFNN	Hybrid Fuzzy Neural Network
SAN	Simple Recurrent Network
ESN	Echo State Network
ASNN	Associate Neural Network
ITNNS	Instantaneously Trained Neural Network
ANFIS	Adaptive Neural Fuzzy Inference System
COM	Committe of Machines

LITERATURE SURVEY

Momoh, J.A. Ma, X.W. Tomsovic, K.outlines the new economic dispatch model that deals with scheduling of generating stations when load occurs. Power generating scheduling is very important aspect of power system engineering. At present the scheduling is done human operators at the load dispatch centers with the help of SCADA. Data is monitored continuously on SCADA and is used to prepare power schedule on each day. Each day load forecasting is done for the next day, based on the previous data for 3-4 years. The problem with this method is that it does take in to account of any unpredictable factors. Because of this on many occasions the scheduling is not perfect. This results in loss of power, lower efficiency and high cost of generation. This paper describes a novel and efficient method with the scheduling of generating station when load occurs. Here factors like rain at the generating station, amount of water in the reservoir and amount of water required to generate one unit of power at each generating station connected to the grid are taken into consideration. These factors are used as input to a fuzzy controller which gives a cost, real generation and reactive generation as output. The validity and effectiveness of designing the controller has been applied to more generating stations and the result s presented and discussed [1].

D. K. Ranaweera, N. F. Hubele and G. G. Karady introduced a multiobjective thermal power dispatch problem minimizes number of objectives viz cost and emission together while allocating the electricity demand among the committed generating units subject to physical and technological constraints. Such problems are solved to generate non-inferior solutions using weighting method or o-constraint method. Afterwards the decision maker is provided with a set of simple but effective tools to choose the best alternative among non-inferior solutions. The generation of non-inferior solution requires an enormous amount of computation time when the number of objectives is more than two. In the paper, the multiobjective problem has been solved a using weighted technique. The Evolutionary optimization technique has been employed in which the 'preferred'

weightage pattern has been searched to get the 'best' optimal solution in non-inferior domain. Decision making theories attempt to deal with the vagueness or fuzziness inherent in subjective or impressive determination of goals. So fuzzy set theory has been exploited to decide the 'preferred' optimal operating point. The non-inferior solution that attains maximum satisfaction level from the membership functions of the participating objectives has been adjudged the 'best' solution. The proposed method requires few search moves to get the optimal operating point in the non-inferior domain for any number of goals. The validity of the proposed method has been demonstrated on a 25 nodes IEEE system comprising five generators. *Keywords:* Multiobjective optimization; Fuzzy set; Decision making; Membership function; Evolutionary optimization technique [2].

Liu, K. Subbarayan, S. Shoults, R.R. Manry, M.T. Kwan, C. Lewis, F.I. Naccarino, J. , Increasing interest has been seen in applying fuzzy set theory to power systems problems from the number of publications on this topic. As a relatively new research topic a need is felt to pay more attention to the understanding of the basic principles of the theory and the identification of problems suitable for solving by this method. This paper presents a survey of publications on applications of fuzzy set theory to power systems and the basic procedures for fuzzy set based methods to solve specific power systems problems. Simple numerical examples are used to show the practical procedures of problem formulation and solution. Theses examples are: generator maintenance scheduling, dynamic programming, and power system stabilizer [3].

Djukanovic, M.B Vlaisavljevic, D.;.; Sobajic, D.J.; Babic, B.S.,introduced that interactive fuzzy satisfying method for solving optimal power system rescheduling by assuming that the decision-maker has imprecise or fuzzy goals and constraints. An interactive decision-making process is formulated in which decision-maker can learn to recognize good solutions by considering all possibilities of fuzziness. The salient features of the proposed method are: (i) ability of solving multi-objective problem which guarantees that a global no inferior solution will be generated; and (ii) the possibility to obtain a reasonable nonfuzzy solution under consideration of the ambiguity of parameters [4].

Hiyama, T. Tomsovic, K. ,introduced that in recent years, fuzzy set theory applications have received increasing attention in various areas of power systems such as operation, planning, and control. A number of research articles appeared which indicate applicability of fuzzy systems to power systems for wider operating conditions under uncertainties. While most of these systems are still under investigation, however, there already exist several practical applications of fuzzy systems. This paper presents a comprehensive set of references on fuzzy set theory applications in power systems [5].

Francisco Jurado, Manual Castro, Jose Corpio & Rivilla has introduced three practical techniques-fuzzy logic (FL), neural networks (NN), and autoregressive models-for very short-term power system load forecasting are proposed and discussed in this paper. Their performances are evaluated through a computer simulation study. The preliminary study shows that it is feasible to design a simple, satisfactory dynamic forecaster to predict very short-term power system load trends online. FL and NN can be good candidates for this application [6].

Y.S. Brar, Jaspreet S. Dhillon, D.P. Kothari introduced fuzzy system applications which received increasing attention in manufacturing industries, heavy industries, transportation systems, water supply systems. The applications are mainly for controllers, however, there also exist many other areas such as prediction, diagnosis, optimization and planning. In power systems, many applications have been proposed, however, most of them are still under investigation or in the development stage. Fuzzy systems have been increasingly used to develop more efficient schemes for power system operation, planning, control and management. This paper presents the current status of fuzzy system applications to power systems and future considerations of fuzzy system applications [7].

Amalia Sergakia , Kostas Kalaitzakis gave an overview of some fuzzy set-based approaches to scheduling is proposed, emphasizing two distinct uses of fuzzy sets: representing preference profiles and modeling uncertainty distributions. The first setting

leads to a valued, non-compensatory generalization of constraint-directed scheduling. The other setting yields a possibility-theoretic counterpart of PERT, where probability distributions of activity durations are changed into possibility distributions, for the purpose of modelling incomplete information. It is pointed out that a special case of the latter, interval-valued PERT, is a difficult, ill-known problem, regarding the determination of critical activities, latest starting times and floats. Lastly when flexible constraints and uncertain processing times are to be jointly considered, the use of possibility decision theory leads to the computation of robust schedules [8].

Didier Duboi, Helene Fargier and Philippe Fortemps introduced the inspection planning in electric power industry which is used to assess the safety and reliability of system components and to increase the ability of failure situation identification before it actually occurs. It reflects the implications of the available information on the operational and maintenance history of the system. The output is a ranked list of components, with the most critical ones at the top, which indicates the selection of the components to be inspected [9].

Bansal, R.C. in this paper demonstrated the use of a fuzzy relational database model for manipulating the data required for the criticality component ranking in thermal power systems inspection planning, incorporating criteria concerning aspects of safety and reliability, economy, variable operational conditions and environmental impacts. Often, qualitative thresholds and linguistic terms are used for the component criticality analysis. Fuzzy linguistic terms for criteria definitions along with fuzzy inference mechanisms allow the exploitation of the operators' expertise. The proposed database model ensures the representation and handling of the aforementioned fuzzy information and additionally offers to the user the functionality for specifying the precision degree by which the conditions involved in a query are satisfied. In order to illustrate the behavior of the model, a case study is given using real inspection data. The results of an investigation of a fuzzy logic model for short term load forecasting are presented. The proposed methodology uses fuzzy rules to incorporate historical weather and load data. These fuzzy rules are obtained from the historical data using a learning-type algorithm. Test results from daily peak and total load forecasts for one year of data from a large scale

power system indicate that the fuzzy rule bases can produce results similar in accuracy to more complicated statistical and back propagation neural network methods [10].

M.Masiala, M.GHribi & A.Kaddouri, in this paper two robust decentralized control design methodologies for load frequency control (LFC) are proposed. The first one is based on control design using linear matrix inequalities(LMI) technique in order to obtain robustness against uncertainties. The second controller has a simpler structure, which is more appealing from an implementation point of view, and it is tuned by a proposed novel robust control design algorithm to achieve the same robust performance as the first one. More specifically, genetic algorithms (GAs) optimization is used to tune the control parameters of the proportional-integral (PI) controller subject to the constraints in terms of LMI. Hence, the second control design is called GALMI. Both proposed controllers are tested on a three-area power system with three scenarios of load disturbances to demonstrate their robust performances. genetic algorithms, , linear matrix inequalities, load frequency control, robust control [11].

C. T. PAN and C. M. LIAW introduced an adaptive controller is presented for load-frequency control of power systems. The new controller uses a PI adaptation to satisfy the hyper stability condition for taking care of the parameter charlges of the system. Only the available information of the states and output of the model as well as the plant output are required f.or the control. No explicit parameter identification is required. Furthermore, the proposed controller can be designed by using a reduced plant model to simplify the design without degrading much the performance. It follow9 that the new controller possesses a very attractive merit of being very easy to implement practically. The simulation results indicate that good control performance can be obtained by this proposed controller, and the performance is insensitive to the plant parameter changes. Moreover, the effectiveness of the proposed controller for considering the generation rate limit can also be confirmed [12].

Y.wang , R zhou,C wen introduced a robust controller, based on the Riccati-equation approach, is proposed for power system load-frequency control. Only the bounds of the

system parameters are required to design the robust load-frequency controller. The pro-posed robust controller is simple, effective and can ensure that the overall system is asymptotically stable for all admissible uncertainties. Simulation results show that, for the example system, the pro posed robust load-frequency controller can achieve good performance even in the presence of generation rate constraint [13].

K.Y.Lim, Y.Wang,R.Zhou itroduced a robust decentralized load-frequency controller. based on the Riccati-equation approach, is proposed for multi-area power Systems with -parametric uncertainties. It comprises N l&al robust load-frequency controllers for ***it*** N-area power system. ***N*** interlinked Riccati equations are produced initially which are separated via a decoupling technique. Bounds of system parametric uncertainties are included in these Riccati equations to improve the robustness of the intended controller. One local robust load frequency relationship is obtained by solving the corresponding decoupled Riccati equation. It operates on its local measurements; feedback from other areas is not needed. Overall system is asymptotically stable, for all admissible system parametric; uncertainties, when all the local load frequency controllers are working together. Good performance is reported from the studied three-area power systern even in the presence of the generation-rate constraint [14].

.

K.Sabahi,M.A.NEekoui M.Teshnehlab M.Aliyari power-system load-frequency control by fuzzy-PI (FPI) controller is proposed. During control, a fuzzy system is used to decide adaptively the proper proportional and integral gains of a PI controller according the area-control error and its change. To ease the design effort and improve the performance of the controller, design of the FPI controller by hybridizing a genetic algorithm and particle-swarm optimization, called FPI–HGAPSO, is proposed. FPI–HGAPSO is based on the hybrid of the genetic algorithm and particle-swarm optimization. In FPI–HGAPSO, elites in the population of GAs are enhanced by particle-swarm optimization and these enhanced elites are selected as parents for crossover and mutation operations. Simulations of the proposed evolutionary FPI-control approach on a multiarea interconnected power system with different kinds of perturbations are performed. The performance of the proposed approach is varied from simulations and comparisons[15].

Lianfang Cong and Le Xio introduced the new state contractive constraint-based predictive control (SCC-NPC) scheme was proposed in this paper. This model predictive control algorithm consists of a basic finite horizon NPC technique and an additional state contractive constraint. The crucial function of the additional state contractive constraint in SCC-NPC is to guarantee the stability of the control scheme. The nominal stability and robust stability of SCC-NPC scheme were analyzed and proved respectively. In our research, the new model predictive control scheme was applied to multi-area load frequency control (LFC). Simulation results show the performance of the scheme. When the areas suffer from load disturbances, the frequency variations and the deviation of power flow between areas go to zero by using SCCNPC scheme [16].

Richard D.Christie and aAnjan Bose introduced an open transmission access is a legal requirement in the United States, but is not fully implemented. Discussion of deregulation has so far focused principally on the tariff structure for transmission access, but operating the power system in this new environment will present significant problems of an almost purely technical nature. Something as simple as frequency control becomes challenging when implemented in the competitive, distributed control environment that true third party wheeling creates. This paper seeks to identify likely deregulation scenarios, identify the technical issues associated with Load Frequency Control, and identify technical solutions, such as standards and algorithms, needed for the operation of this key component of national infrastructure in the face of profound structural changes [17].

Jwad Talaqand Fadel Al Basri introduced an adaptive fuzzy gain scheduling scheme for conventional PI and optimal load frequency controllers has been proposed. A Sugeno type fuzzy inference system is used in the proposed controller. The Sugeno type fuzzy inference system is extremely well suited to the task of smoothly interpolating linear gains across the input space when a very non-linear system moves around in its operating space. The proposed adaptive controller requires much less training patterns than a neural net based adaptive scheme does and hence avoiding excessive training time.

Results of simulation show that the proposed adaptive fuzzy controller offers better performance than fixed gain controllers at different operating conditions [18].

Dulpichet Rerkpreedapong introduced two robust decentralized control design methodologies for load frequency control (LFC) are proposed. The first one is based on control design using linear matrix inequalities (LMI) technique in order to obtain robustness against uncertainties. The second controller has a simpler structure, which is more appealing from an implementation point of view, and it is tuned by a proposed novel robust control design algorithm to achieve the same robust performance as the first one. More specifically, genetic algorithms (GAs) optimization is used to tune the control parameters of the proportional-integral (PI) controller subject to the constraints in terms of LMI. Hence, the second control design is called GALMI. Both proposed controllers are tested on a three-area power system with three scenarios of load disturbances to demonstrate their robust performances [19].

C.-F. Juang and C.-F. Lu introduced power-system load-frequency control by fuzzy-PI (FPI) controller is proposed. During control, a fuzzy system is used to decide adaptively the proper proportional and integral gains of a PI controller according the area-control error and its change. To ease the design effort and improve the performance of the controller, design of the FPI controller by hybridizing a genetic algorithm and particle-swarm optimisation, called FPI–HGAPSO, is proposed. FPI–HGAPSO is based on the hybrid of the genetic algorithm and particle-swarm optimisation. In FPI–HGAPSO, elites in the population of GAs are enhanced by particle-swarm optimisation and these enhanced elites are selected as parents for crossover and mutation operations. Simulations of the proposed evolutionary FPI-control approach on a multiarea interconnected power system with different kinds of perturbations are performed. The performance of the proposed approach is verified from simulations and comparisons [20].

H.D. Mathur, and S. Ghosh introduced novel approaches of artificial intelligence (AI) techniques, like fuzzy logic, artificial neural network (ANN), hybrid fuzzy neural network (HFNN), genetic algorithm (GA) for the load frequency control of electrical

power system. The limitations of conventional controls such as proportional, integral and derivative are slow and lack of efficiency in handling system nonlinearities. Since high frequency deviation may lead to system collapse, this necessitates an accurate and fast acting controller to maintain the constant nominal system frequency. The intelligent controllers are used for load frequency control for the single area system, multi area interconnected system. The performance of intelligent controllers with the conventional controllers has been thoroughly compared and analyzed [21].

Kourosh Sedghisigarchi introduced a concept of a distribution system that has enough generation to track its load without the help of a substation. Specifically, it addresses the presence of solid oxide fuel cells in the distributed generation mix. Two control loops are proposed 1) to guarantee that the fuel cell is protected by maintaining its cell utilization within its admissible range and 2) to track load changes and regulate the frequency. The two respective loops are called primary and secondary loops. The primary loop maintains constant the ratio of stack current to input fuel flow, and the secondary loop tracks the load and regulates the frequency. A distribution area error (DSE) is introduced to formulate the frequency-control problem. The secondary loop feeds back this DSE signal. Tuning of the parameters is performed using genetic algorithms [22] .

CHAPTER 1

INTRODUCTION

Large scale power systems are normally composed of control areas or regions representing coherent groups of generators. The various areas are interconnected through tie-lines. The tie-lines are utilized for contractual energy exchange between areas and provide inter-area support in case of abnormal conditions. Area load changes and abnormal conditions, such as outages of generation, lead to mismatches in frequency and scheduled power interchanges between areas. These mismatches have to be corrected via supplementary control. Load Frequency Control (LFC) of interconnected systems is defined as the regulation of power output of generators within a prescribed area, in response to change in system frequency, tie-line loading, or the relation of these to each other; so as to maintain scheduled system frequency and/or established interchange with other areas within predetermined limits.

Many investigations in the area of LFC problem of interconnected power systems have been reported over the last six decades. A number of control strategies have been employed in the design of load frequency controllers in order to achieve better dynamic performance. Among the various types of load frequency controllers, the most widely employed is the conventional proportional integral controller. The PI controller is simple for implementation but generally gives large frequency deviations. A number of state feedback controllers based on linear optimal control theory have been proposed to achieve better performance. Alternative and more practical forms of feedback controllers such as output feedback and decentralized controllers have been the subject of extensive investigations.

Fixed gain controllers are designed at nominal operating conditions and fail to provide best control performance over a wide range of operating conditions. So, to keep system performance near its optimum, it is desirable to track the operating conditions and use updated parameters to compute the control. Adaptive controllers with self-adjusting gain settings have been proposed for LFC. Despite the promising results achieved by these adaptive controllers, the control algorithms are complicated and require on-line system

model identification. Artificial neural networks and fuzzy systems have been successfully applied to the LFC problem with rather promising results. The salient feature of these techniques is that they provide a model-free description of control systems and do not require model identification. In this thesis, an adaptive fuzzy gain scheduling scheme for conventional PI and optimal controllers has been proposed and tested for off-nominal operating conditions. The proposed controller offers better performance than fixed gain controllers.

Load frequency control (LFC) is the mechanism by which a balance between power generation and demand is satisfied. Usually, the load frequency controllers used in the industry are proportional-integral (PI) type and are tuned online based on trial-and-error approaches. Several optimization techniques have been proposed to tune the control parameters using simulation of the entire system rather than just the control area being studied. Some of them simply assume that all subsystems are identical, which is not the case of actual power systems. Subsequently, a number of decentralized load frequency controllers were developed to eliminate the above drawback. But most of them are complex state-feedback or high order dynamic controllers, which are not practical for industry practices. This thesis proposes two robust decentralized LFC controllers. The first one is based on theory, and results in a high order controller. The second controller is a PI controller tuned by a novel robust control design algorithm to achieve the same robust performance as the first one, but it is more appealing from an implementation point of view. Load-frequency control or called automatic generation control (AGC) is very important item in power system operation and control for supplying sufficient and reliable electric power with good quality. Many control strategies have been proposed to achieve better performance. Due to the nonlinearities of various components of power systems, a linear model obtained by linearization around an operating point is usually adopted for the controller design. However, because of the inherent characteristics of changing loads, the operating point of a power system may change very much during a daily cycle. As a result, a fixed controller which is optimal under one operating condition may no longer be suitable in another status. In view of this, some authors have applied the variable structure control to make the controller insensitive to the plant parameter

changes. However, this method requires the information of the system states which is not completely known generally. On the other hand, recently, various adaptive control techniques have been proposed for dealing with large parameter variations. Basically, adaptive control systems can be classified into two categories, namely the self-tuning regulators and the model reference control systems.

Load-frequency control (LFC) is of importance in electric power system design and operation. The loading in a power system is never constant. To ensure the quality of the power supply it is necessary to design a load-frequency control system which deals with the control of loading of the generator depending on the frequency. There has been continuing interest in designing load-frequency controllers with better performance during past 20 years. Many control strategies for LFC have been proposed since the 1970s**.** An industrial plant such as power systems always contains parametric uncertainties. In the design of a controller, the uncertainties have to be considered. Otherwise, if the real plant differs from the assumed plant model, a controller design based on classical approaches may not ensure the stability of the overall system.

In electric-power generation, disturbances caused by load .actuations will cause changes in the desired frequency value. Load-frequency control (LFC) is thus very important in power-system operation for supplying sufficient and reliable electrical power of good quality. Many types of control technique such as PI control, linear-state feedback optimal control and variable-structure control have been used. In, a robust linear controller was used. However, in this design process, a linear model is considered, and both the nonlinear terms of governor dead band and the generation-rate constraint (GRC) are ignored. In addition, for these controller-design methods, a mathematical model of the controlled system is usually required. Among the various types of load-frequency control, the PI controller is most widely applied to peed governor systems for LFC schemes. One advantage of the PI controller is that it reduces the steady state error to zero. However, since the conventional PI controller with .xed gains has been designed at nominal operating conditions, it fails to provide the best control performance over a wide range of operating conditions and exhibits poor dynamic performance. To solve this problem,

adaptive-gain-scheduling techniques have been proposed, most of which are based on the adoption of a fuzzy system for gain scheduling. In three parameters that represent system operating conditions are monitored and used as inputs to a fuzzy system whose output is the adaptive gain of the integral or PI controller. In fuzzy gain scheduling of a PI controller is proposed. The input to the fuzzy system is a newly denned area-control error and its change. In the aforementioned fuzzy-gain-scheduling approach, the precondition part of the designed fuzzy system is partitioned by grid type, resulting a large number of fuzzy rules. Furthermore, in these approaches, the selection of fuzzy if–then rules relies on a substantial quantity of eristic observations to express knowledge of proper strategy. Clearly, it is difficult for a human expert to search for a number of proper rules for the fuzzy system. To resolve this problem, many evolutionary fuzzy systems that automate the design of a fuzzy system by evolutionary algorithms have been proposed, of which the best known are genetic fuzzy systems. In this thesis, we will apply the idea of evolutionary fuzzy systems to the LFC problem.

Frequency is a major stability criterion for large-scale stability in multi area power systems. To provide the stability, active power balance and constant frequency are required. Frequency depends on active power balance. If any change occurs in active power demand/generation in power systems, frequency cannot be hold in its rated value. So oscillations increase in both power and frequency. Thus, system subjects to a serious instability problem. To improve the stability of the power networks, it is necessary to design a load frequency control (LFC) systems that control the power generation and active power at tie lines. Because of the relationship between active power and frequency, three level automatic generation controls have been proposed by power system researchers. In interconnected power networks with two or more areas, the generation within each area has to be controlled so as to maintain scheduled power interchange.

Load frequency control scheme have to be two main control loops. These are primary control and secondary control. This action is realized by turbine-governor system in the plant. In this control level, only active power is balanced. However, maintaining the

frequency at scheduled value (e.g. 50 Hz) cannot be provided. Therefore, steady state frequency error can occur forever. So this level does not enough for interconnected system. In interconnected power systems, frequency must be equal at all areas. The second level of generation control called as secondary or supplementary control is happened in large power systems which include two or more areas. Active power is controlled at the tie line between neighbor areas and there are central and local load control and distribution center.

The objective of the load frequency control (LFC) of a power system is to maintain the frequency of each area and tie-line power flow (in interconnected system) within specified tolerance by adjusting the MW outputs of LFC generators so as to accommodate fluctuating load demands. LFC is also simultaneous automatic generation control (AGC). Many investigations in the area of LFC problem of interconnected systems have been reported in the past six decades. A number of control schemes have been employed in the design of load frequency controllers in order to achieve better dynamic performance. Among the various types of load frequency controllers, the most widely conventional types used are the tie-line bias control and flat frequency control to achieve the above goals of the LFC; both schemes are based on the classical PID (Proportional, integral and differential) controls which work on some function made up of the frequency and tie-line power deviations. Nevertheless, these conventional systems have been successful to some extent. With the enlargement of power system size and capacity together with the strengthening of interconnections, abnormal phenomena have been frequently observed such as excessively large tie-line power deviations and/or sustained power oscillations under sudden system load changes. This fact suggests the necessity of more advanced control strategies to be incorporated for better control. With the recent technological innovations, intelligent controllers have been replacing conventional controllers in order to have fast and good dynamic response for load frequency control problem. Many intelligent techniques such as Fuzzy logic, artificial neural network, hybrid fuzzy neural network and genetic algorithm are being used extensively in isolated as well as interconnected power systems. Load frequency control is important in electrical power system design and operation. Moreover, to ensure the

quality of the power supply, it is necessary to design a LFC system, which deals with the control of loading of generator depending on the frequency. Conventional controllers are quite often impractical for the implementation due to following reasons. (i) The optimal control is a function of all the states of the system. In practice all the states may not be available. The inaccessible states or missing states are required to be estimated. (ii) It may not be economical to transfer all the information over long distances. (iii) The control, which is a function of the states in turn, is dependent on the load demand. Accurate prediction of load demand may be essential for realizing optimal controller.(iv) The optimal control is also dependent on the weighing matrices and is not unique. An attempt has been made in this thesis to a comprehensive analysis of Fuzzy logic and Hybrid fuzzy Neural Network (HFNN) based controllers for the Load frequency control which have been applied by researchers and performance of various controllers is compared with other controllers.

CHAPTER 2

FUZZY LOGIC

The concept of Fuzzy Logic (FL) was conceived by Lotfi Zadeh, a professor at the University of California at Berkley, and presented not as a control methodology, but as a way of processing data by allowing partial set membership rather than crisp set membership or non-membership. This approach to set theory was not applied to control systems until the 70's due to insufficient small-computer capability prior to that time. Professor Zadeh reasoned that people do not require precise, numerical information input, and yet they are capable of highly adaptive control. If feedback controllers could be programmed to accept noisy, imprecise input, they would be much more effective and perhaps easier to implement. Unfortunately, U.S. manufacturers have not been so quick to embrace this technology while the Europeans and Japanese have been aggressively building real products around it.

In this context, FL is a problem-solving control system methodology that lends itself to implementation in systems ranging from simple, small, embedded micro-controllers to large, networked, multi-channel PC or workstation-based data acquisition and control systems. It can be implemented in hardware, software, or a combination of both. FL provides a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy, or missing input information. FL's approach to control problems mimics how a person would make decisions, only much faster.

2.1 Difference between Fuzzy Logic and conventional control methods

FL incorporates a simple, rule-based IF X AND Y THEN Z approach to a solving control problem rather than attempting to model a system mathematically. The FL model is empirically-based, relying on an operator's experience rather than their technical understanding of the system. For example, rather than dealing with temperature control in terms such as "SP =500F", "T <1000F", or "210C <TEMP <220C", terms like "IF (process is too cool) AND (process is getting colder) THEN (add heat to the process)" or "IF (process is too hot) AND (process is heating rapidly) THEN (cool the process

quickly)" are used. These terms are imprecise and yet very descriptive of what must actually happen. Consider what you do in the shower if the temperature is too cold: you will make the water comfortable very quickly with little trouble. FL is capable of mimicking this type of behavior but at very high rate.

2.2 How does Fuzzy Logic work

FL requires some numerical parameters in order to operate such as what is considered significant error and significant rate-of-change-of-error, but exact values of these numbers are usually not critical unless very responsive performance is required in which case empirical tuning would determine them. For example, a simple temperature control system could use a single temperature feedback sensor whose data is subtracted from the command signal to compute "error" and then time-differentiated to yield the error slope or rate-of-change-of-error, hereafter called "error-dot". Error might have units of degs F and a small error considered to be 2F while a large error is 5F. The "error-dot" might then have units of degs/min with a small error-dot being 5F/min and a large one being 15F/min. These values don't have to be symmetrical and can be "tweaked" once the system is operating in order to optimize performance. Generally, FL is so forgiving that the system will probably work the first time without any tweaking.

2.3 Features of Fuzzy Logic

FL offers several unique features that make it a particularly good choice for many control problems.

1) It is inherently robust since it does not require precise, noise-free inputs and can be programmed to fail safely if a feedback sensor quits or is destroyed. The output control is a smooth control function despite a wide range of input variations.

2) Since the FL controller processes user-defined rules governing the target control system, it can be modified and tweaked easily to improve or drastically alter system performance. New sensors can easily be incorporated into the system simply by generating appropriate governing rules.

3) FL is not limited to a few feedback inputs and one or two control outputs, nor is it necessary to measure or compute rate-of-change parameters in order for it to be

implemented. Any sensor data that provides some indication of a system's actions and reactions is sufficient. This allows the sensors to be inexpensive and imprecise thus keeping the overall system cost and complexity low.

4) Because of the rule-based operation, any reasonable number of inputs can be processed (1-8 or more) and numerous outputs (1-4 or more) generated, although defining the rule base quickly becomes complex if too many inputs and outputs are chosen for a single implementation since rules defining their interrelations must also be defined. It would be better to break the control system into smaller chunks and use several smaller FL controllers distributed on the system, each with more limited responsibilities.

5) FL can control nonlinear systems that would be difficult or impossible to model mathematically. This opens doors for control systems that would normally be deemed unfeasible for automation.

2.4 Use of Fuzzy Logic

1) Define the control objectives and criteria: What am I trying to control? What do I have to do to control the system? What kind of response do I need? What are the possible (probable) system failure modes?

2) Determine the input and output relationships and choose a minimum number of variables for input to the FL engine (typically error and rate-of-change-of-error).

3) Using the rule-based structure of FL, break the control problem down into a series of IF X AND Y THEN Z rules that define the desired system output response for given system input conditions. The number and complexity of rules depends on the number of input parameters that are to be processed and the number fuzzy variables associated with each parameter. If possible, use at least one variable and its time derivative. Although it is possible to use a single, instantaneous error parameter without knowing its rate of change, this cripples the system's ability to minimize overshoot for a step inputs.

4) Create FL membership functions that define the meaning (values) of Input/Output terms used in the rules.

5) Create the necessary pre- and post-processing FL routines if implementing in S/W, otherwise program the rules into the FL H/W engine.

6) Test the system, evaluate the results, tune the rules and membership functions, and retest until satisfactory results are obtained.

2.5 Fuzzy expert system

A fuzzy expert system is an expert system that uses a collection of fuzzy membership functions and rules, instead of Boolean logic, to reason about data. The rules in a fuzzy expert system are usually of a form similar to the following:

if x is low and y is high then z = medium

where x and y are input variables (names for know data values), z is an output variable (a name for a data value to be computed), low is a membership function (fuzzy subset) defined on x, high is a membership function defined on y, and medium is a membership function defined on z. The antecedent (the rule's premise) describes to what degree the rule applies, while the conclusion (the rule's consequent) assigns a membership function to each of one or more output variables. Most tools for working with fuzzy expert systems allow more than one conclusion per rule. The set of rules in a fuzzy expert system is known as the rule base or knowledge base.

The general inference process proceeds in three (or four) steps.

1. Under FUZZIFICATION, the membership functions defined on the input variables are applied to their actual values, to determine the degree of truth for each rule premise.

2. Under INFERENCE, the truth value for the premise of each rule is computed, and applied to the conclusion part of each rule. This results in one fuzzy subset to be assigned to each output variable for each rule. Usually only MIN or PRODUCT are used as inference rules. In MIN inferencing, the output membership function is clipped off at a height corresponding to the rule premise's computed degree of truth (fuzzy logic AND). In PRODUCT inferencing, the output membership function is scaled by the rule premise's computed degree of truth.

3. Under COMPOSITION, all of the fuzzy subsets assigned to each output variable are combined together to form a single fuzzy subset for each output variable. Again, usually MAX or SUM are used. In MAX composition, the combined output fuzzy subset is constructed by taking the point wise maximum over all of the fuzzy subsets assigned to variable by the inference rule (fuzzy logic OR). In SUM composition, the combined output fuzzy subset is constructed by taking the point wise sum over all of the fuzzy subsets assigned to the output variable by the inference rule.

4. Finally is the (optional) DEFUZZIFICATION, which is used when it is useful to convert the fuzzy output set to a crisp number. There are more defuzzification methods than you can shake a stick at (at least 30). Two of the more common techniques are the CENTROID and MAXIMUM methods. In the CENTROID method, the crisp value of the output variable is computed by finding the variable value of the center of gravity of the membership function for the fuzzy value. In the MAXIMUM method, one of the variable values at which the fuzzy subset has its maximum truth value is chosen as the crisp value for the output variable.

2.6 Fuzzy Set

The very basic notion of fuzzy systems is a fuzzy (sub)set. In classical mathematics we are familiar with what we call *crisp sets*. For example, the possible interferometric coherence values are the set *X* of all real numbers between 0 and 1. From this set *X* a subset *A* can be defined, (e.g. all values 0< □< 0.2). The *characteristic function* of A, (i.e. this function assigns a number1 or 0 to each element in *X,* depending on whether the element is in the subset *A* or not) is shown in Figure 3.1.

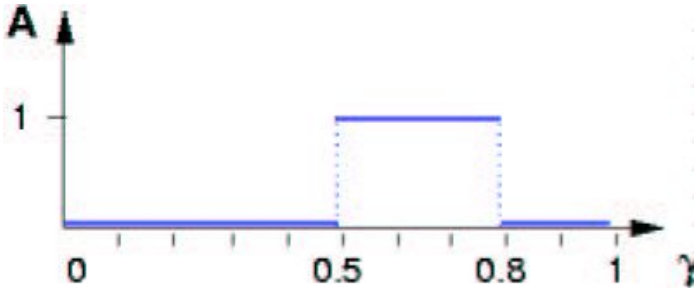

Figure 2.1: Characteristic Function of a Crisp Set

The elements which have been assigned the number 1 can be interpreted as the elements that are in the set *A* and the elements which have assigned the number 0 as the elements that are not in the set *A*. This concept is sufficient for many areas of applications, but it can easily be seen, that it lacks in flexibility for some applications like classification of remotely sensed data analysis. For example it is well known that water shows low interferometric coherence γ in SAR images. Since γ starts at 0, the lower range of this set ought to be clear. The upper range, on the other hand, is rather hard to define. As a first attempt, we set the upper range to 0.2. Therefore we get B as a crisp interval B=[0,0.2]. But this means that a γ value of 0.20 is low but a γ value of 0.21 not. Obviously, this is a structural problem, for if we moved the upper boundary of the range from γ =0.20 to an arbitrary point we can pose the same question. A more natural way to construct the set *B* would be to relax the strict separation between *low* and *not low*. This can be done by allowing not only the (*crisp*) decision *Yes/No*, but more flexible rules like “fairly low". A fuzzy set allows us to define such a notion. The aim is to use fuzzy sets in order to make computers more intelligent; therefore, the idea above has to be coded more formally. In the example, all the elements were coded with 0 or 1. A straight way to generalize this concept is to allow more values between 0 and 1. In fact infinitely many alternatives can be allowed between 0 and 1, namely the unit interval I =[0, 1]. The interpretation of the numbers, now assigned to all elements is much more difficult. Of course, again the number 1 assigned to an element means, which the element is in the set *B* and 0 means that the element is definitely not in the set *B*. All other values mean a gradual membership to the set *B*. This is shown in Figure 2.2. The *membership function* is a graphical representation of the magnitude of participation of each input. It associates a weighting with each of the inputs that are processed, define functional overlap between inputs, and ultimately determines an output response. The rules use the input membership values as weighting factors to determine their influence on the fuzzy output sets of the final output conclusion.

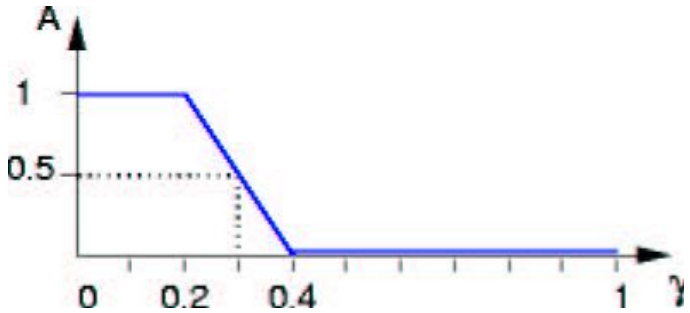

Figure 2.2: Characteristic Function of a Fuzzy Set

The membership function, operating in this case on the fuzzy set of interferometric coherence γ.returns a value between 0.0 and 1.0. For example, an interferometric coherence γ of 0.3 has a membership of 0.5 to the set *low coherence* (see Figure 2.2).

It is important to point out the distinction between fuzzy logic and probability. Both operate over the same numeric range, and have similar values: 0.0 representing *False* (or non membership), and 1.0 representing *True* (or full membership). However, there is a distinction to be made between the two statements: The probabilistic approach yields the natural language statement, "There is a 50% chance that γ is low," while the fuzzy terminology corresponds to "γ 's degree of membership within the set of low interferometric coherence is 0.50." The semantic difference is significant: the first view supposes that γ is or is not low; it is just that we only have an 50% chance of knowing which set it is in. By contrast, fuzzy terminology supposes that γ is "more or less" low, or in some other term corresponding to the value of 0.50.

2.6.1 Operations on Fuzzy Sets

We can introduce basic operations on fuzzy sets. Similar to the operations on crisp sets we also want to intersect, unify and negate fuzzy sets. In his very first paper about fuzzy sets [1], L. A. Zadeh suggested the minimum operator for the intersection and the maximum operator for the union of two fuzzy sets. It can be shown that these operators coincide with the crisp unification, and intersection if we only consider the membership degrees 0 and 1. For example, if *A* is a fuzzy interval between 5 and 8 and *B* be a fuzzy number about 4 as shown in the Figure below

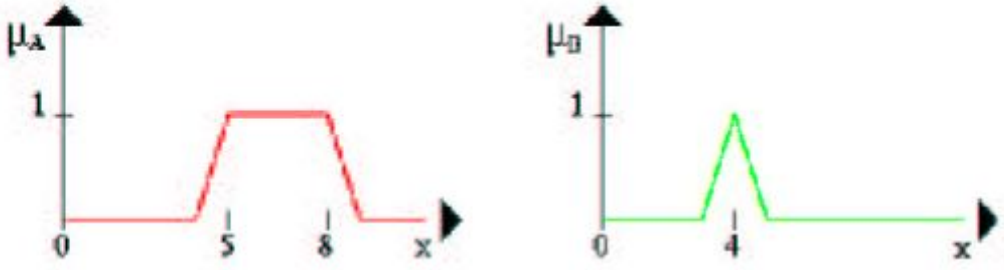

Figure 2.3: Example fuzzy sets

In this case, the fuzzy set between 5 and 8 *AND* about 4 is

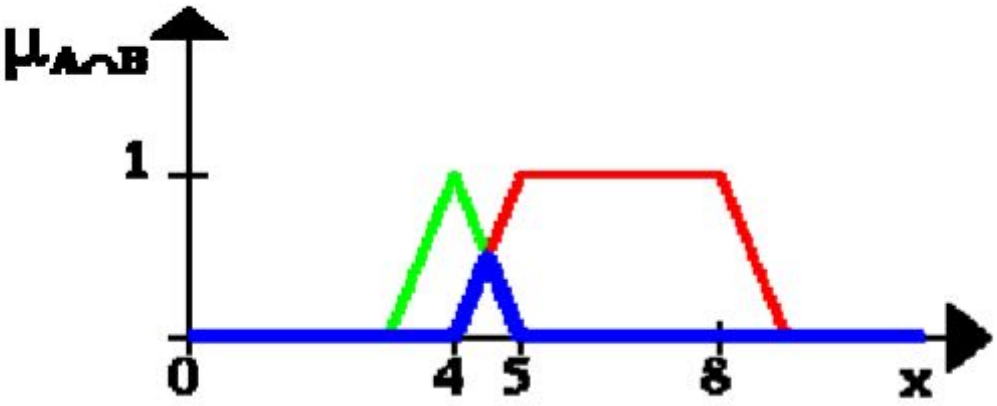

Figure 2.4: Fuzzy *AND*

set between 5 and 8 *OR* about 4 is shown in the next figure

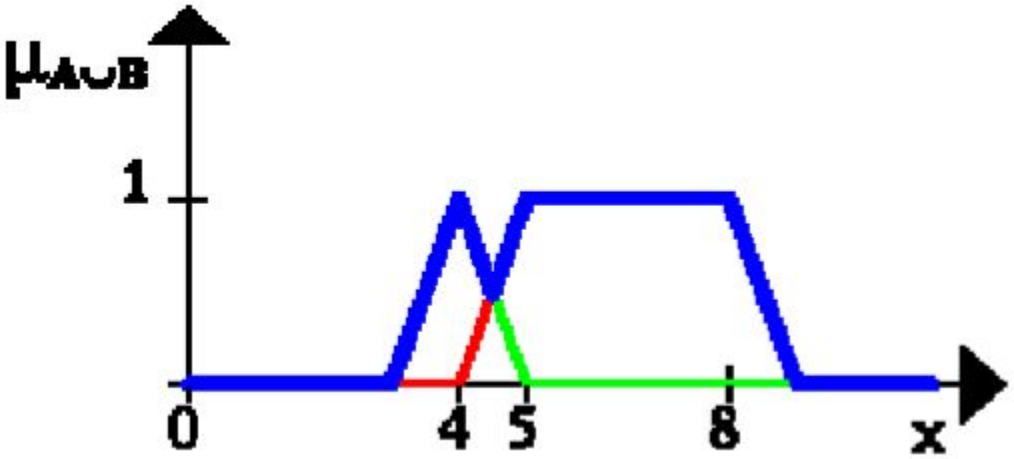

Figure 2.5: Fuzzy *OR*

The NEGATION of the fuzzy set A is shown below

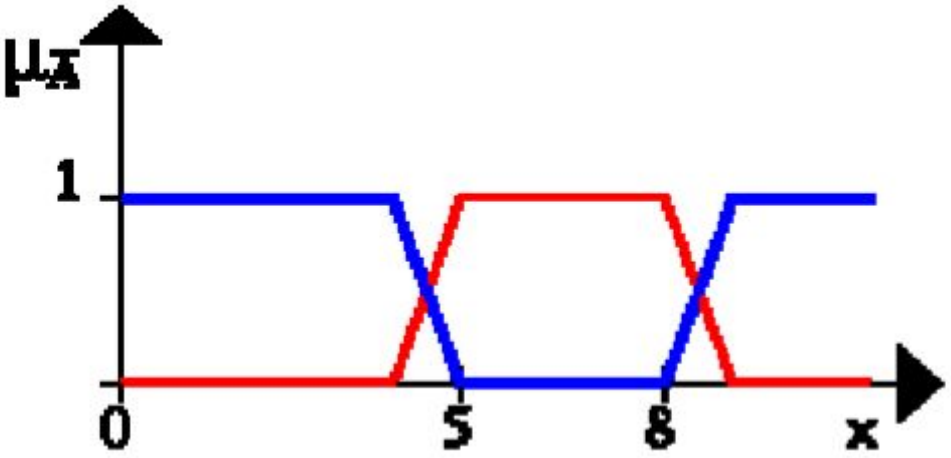

Figure 2.6: Fuzzy NEGATION

2.7 Fuzzy Control

Automatic control belongs to the application areas of fuzzy set theory that have attracted most attention. In 1974, the first successful application of fuzzy logic to the control of a

laboratory-scale process was reported (Mamdani and Assilian 1975). Control of cement kilns was an early industrial application (Holmblad and Ostergaard 1982). Since the first consumer product using fuzzy logic was marketed in 1987, the use of fuzzy control has increased substantially. A number of CAD environments for fuzzy control design have emerged together with VLSI hardware for fast execution. Fuzzy control is being applied to various systems in the process industry (Santhanam and Langari 1994, Tani et al. 1994), consumer electronics (Hirota 1993, Bonissone 1994), automatic train operation (Yasunobu and Miyamoto 1985), traffic systems in general (Hellendoorn 1993), and in many other fields (Hirota 1993, Terano et al. 1994).

A fuzzy logic controller describes a control protocol by means of if-then rules, such as "if temperature is low open heating valve slightly". The ambiguity (uncertainty) in the definition of the linguistic terms (e.g., *low temperature*) is represented by using *fuzzy sets*, which are sets with overlapping boundaries, see Figure 3.7. In the fuzzy set framework, a particular domain element can simultaneously belong to several sets (with different degrees of membership, μ). For instance, $t = 20°$ C belongs to the set of *High* temperatures with membership 0.4 and to the set of *Medium* temperatures with membership 0.2. This gradual transition from membership to non-membership facilitates a smooth outcome of the reasoning (deduction) with fuzzy if-then rules.

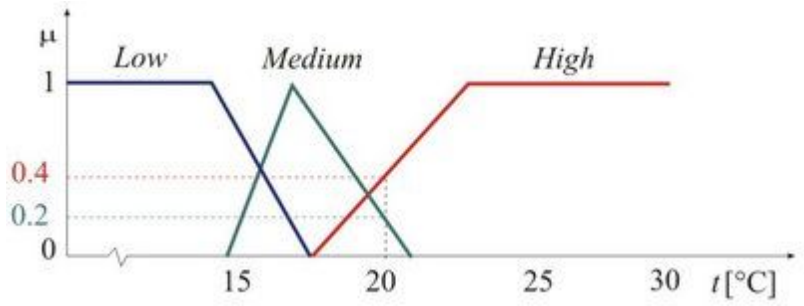

Figure 2.7: Partitioning of the temperature domain into three fuzzy sets.

2.7.1 Historical perspective

Fuzzy Control emerged from a doctorate thesis in Artificial Intelligence. The work was carried out around 1972 to 1974. The objective of the research was to see if computers

can learn to perform a task by observing a human carry it out. The task chosen was the control of a model steam engine. The controls available were "heat" which produced the steam pressure in a boiler; and, "throttle" which injected the steam into the single cylinder engine affecting its speed. The control objective was to maintain constant pressure in the boiler and speed of the engine. Several key issues about this research project have to be highlighted to gain a good understanding of how Fuzzy Control emerged from this research project.

2.7.2 The Computational Environment

A small hybrid computer was available built around a PDP-8S mini-computer with 8K words (12 bit words) of magnetic core RAM. Paper tape was the main form of back up store. The mouse had not been invented yet and the communication was via a teletype. The steam engine inputs and outputs were set and read via the hybrid computer. An individual run of the experiment would involve the teletype printing out the present speed and pressure and waiting for the operator to respond by typing in the values for heat and throttle settings. On pressing the return key, the computer carried out these settings and responded by typing the new readings of the speed and pressure. This went on until the boiler ran out of water whereupon a new run could be started.

The study was agnostic with respect to any particular mathematical formalism to be used. Given the limited resources of our computational environment, we felt it was easiest to begin by using a Bayesian learning approach. The idea was to update the probabilities of an action given the state of the steam engine. This was obviously a naïve approach and not surprisingly the learnt probabilities failed to converge. It was naïve because it failed to take into account the fact that we were controlling a dynamic system, and the human operator did not merely take the current state into account in determine his action but was aware of the system's previous state trajectory. The algorithm had to be revised in such a way that it needed to take the previous states into account but it was feared that this would make it more complex and could stress the available RAM.

2.7.3 Motivation for fuzzy control

Conventional control theory uses an explicit mathematical (analytical) model of a process to be controlled and specifications of the desired closed-loop behavior to design a controller. This approach may fall short if the model of the process is difficult to obtain, (partly) unknown, or highly nonlinear. The design of controllers for seemingly easy everyday tasks such as driving a car or grasping a fragile object continues to be a challenge for robotics, while these tasks are easily performed by human beings. Yet, humans do neither use mathematical models nor exact trajectories for controlling such processes.

Many processes controlled by human operators in industry cannot be automated using conventional control techniques, since the performance of these controllers is often inferior to that of the operators. One of the reasons is that linear controllers, which are commonly used in conventional control, are not appropriate for nonlinear plants. Another reason is that humans aggregate various kinds of information and combine control strategies, that cannot be integrated into a single analytic control law. The underlying principle of *knowledge-based (expert) control* is to capture and implement experience and knowledge available from experts (e.g., process operators). A specific type of knowledge-based control is the fuzzy rule-based control, where the control actions corresponding to particular conditions of the system are described in terms of fuzzy if-then rules. Fuzzy sets are used to define the meaning of qualitative values of the controller inputs and output such *small* error, *large* control action. Fuzzy logic can capture the continuous nature of human decision processes and as such is a definite improvement over methods based on binary logic (which are widely used in industrial controllers).

The early work in fuzzy control was motivated by a desire to mimic the control actions of an experienced human operator (knowledge-based part) obtain smooth interpolation between discrete outputs that would normally be obtained (fuzzy logic part).

Since then the application range of fuzzy control has widened substantially. However, the two main motivations still persevere. The linguistic nature of fuzzy control makes it possible to express process knowledge concerning how the process should be controlled

or how the process behaves. The interpolation aspect of fuzzy control has led to the viewpoint where fuzzy systems are seen as smooth function approximation schemes.

In most cases a fuzzy controller is used for direct feedback control. However, it can also be used on the supervisory level as, e.g., a self-tuning device in a conventional PID (Proportional-Integral-Differential) controller. Also, fuzzy control is no longer only used to directly express a priori process knowledge. For example, a fuzzy controller can be derived from a fuzzy model obtained through system identification. Most often used are :

- *Mamdani (linguistic) controller* with either fuzzy or singleton consequents. This type of controller is usually used as a *direct* closed-loop controller.
- *Takagi-Sugeno (TS) controller*, typically used as a *supervisory* controller.

2.8 Fuzzy Controller

When a person controls a process, perhaps as commonplace as a car engine or as specialized as a chemical plant, they describe their expertise in an apparently imprecise way. In starting a car a person may accelerate the engine "a little" before engaging the clutch and driving away. The chemical plant operator may reduce process heating "slightly" if the product temperature is rising "slowly". While imprecise, such rules appear to work well in practice. If such a process needs to be automated one approach is to attempt to emulate the human operator. The usual starting point in the development of such a control system is the process of knowledge elicitation in which a record is constructed of the human operator's expertise. In some cases such expertise is expressed in the form of rules which make use of linguistic variables. It is at this point that the system developers can make an implementation decision. The human's expertise could be applied directly. Alternatively, control technologists could be called in to do in-depth numerical analysis of the process and to recommend an algorithm for control.

In some cases the latter approach would turn out to be too expensive and time consuming, in others it might yield no results at all. In control technology terms there would be no process model obtainable within reasonable time and budget limitations. To control the process what is therefore needed is a computer based system which can use the control rules directly to implement an automatic control system. A fuzzy controller, which will make direct use of the control rules, is a strong candidate as a technical solution.

Developing a fuzzy control knowledge base is divided into two main tasks. The first is to choose a suitable set of linguistic variables to describe the values of the main control parameters. This selection plays an important role in the smoothness of control. When all the different labels have been selected, their membership function must be defined. This process is highly subjective. One might also use the neural networks to learn the membership function from examples. The second task is to state the rules in the control knowledge base with the aid of the chosen linguistic description. This can be done in several ways, for instance by interviewing an experienced human operator.

The basic architecture of a fuzzy controller is depicted in the figure. The decision making logic consists of the inference and composition sub process.

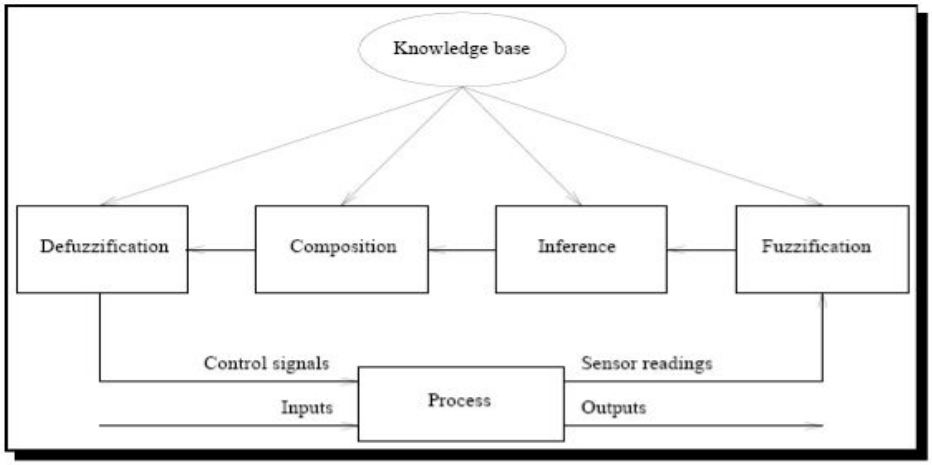

Figure 2.8: Topology of a typical fuzzy controller

2.8.1 Fuzzification

In the Fuzzification process, the membership functions defined on the input variables are applied to their actual values if the input variables are crisp. If the sensor is fuzzy (noisy), fuzzification refers to finding the intersection of the label's membership function and the distribution for the sensed data. Figure (3.9) shows how a sensor reading x_0 is matched with the membership function $\mu(x)$ to get $\mu(x_0)$ in both the crisp and fuzzy case. Usually the sensor reading is crisp.

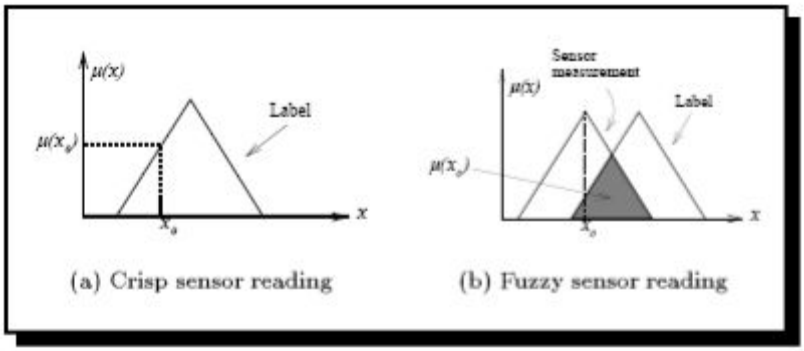

Figure 2.9: The fuzzification sub process

What fuzzification does is to turn the measurement into a degree of membership. Suppose a temperature measurement is made and corresponds to 80^0C. This measurement is required for an application in which the linguistic variable temperature might take on the values "high", "OK and "low". Fuzzification takes the measurement and decides to what degree it is "high", "OK" and "low". This matter of degree is decided on the basis of the framework suggested by the "expert" and is usually expressed as a membership function. The expert might consider that 80°C should be considered more "hot" than "OK". Where "1" represents full membership, the measurement might be

"high" to a degree 0.8 because it is well above normal operating temperature, but it could get higher

"OK to a degree 0.35 because operating temperature could be this high but would normally be lower

"low" to a degree 0 because this temperature could never be considered low

In a simple implementation the control rules will be applied to the degree implied by a single measurement. In this case the controller would apply rules applicable under high temperature conditions to a degree 0.8, and rule applying to "OK" temperature condition to a degree 0.35. The control action is in this way biased according to conditions.

The fuzzification process thus determines how applicable each component of a rule's premise is. The applicability of the premise (if the rule applies at all) is its truth value so to speak. If a rule's premise has a non-zero degree of truth then the rule is said to fire.

2.8.2 Inference

In the inference sub process, the truth value for the premise of each rule is computed and applied to the conclusion part of the rule. The result of this is assigning a fuzzy subset to each output variable of each rule. The truth value of the precondition of a rule is referred to as its strength and is denoted by α. The rule's strength is computed by the means of equations of complement, union and intersection.

Example: Assume we have the following rule:

R: IF x is A AND y is B THEN z is C

If we have the sensor readings x_0 and y_0 for x and y respectively, the strength of the rule R can be computed as $\alpha = \mu_A(x_0) \cap \mu_B(y_0)$.

Of course, the rule's premise may include disjunctions and negations as well as conjunctions. The premise is then computed on the basis of the previous given definitions of union, intersection and complement of fuzzy sets. We also apply more complex operations than min and max for fuzzy inferencing.

There are two widely used inference methods: Min-inferencing and Product-inferencing. In MIN-inferencing, the output membership function is clipped off at a height corresponding to rule's degree of truth. In Product-inferencing, the output membership function is scaled by the rule's computed degree of truth. A graphical illustration of the two inferencing methods is shown in figure

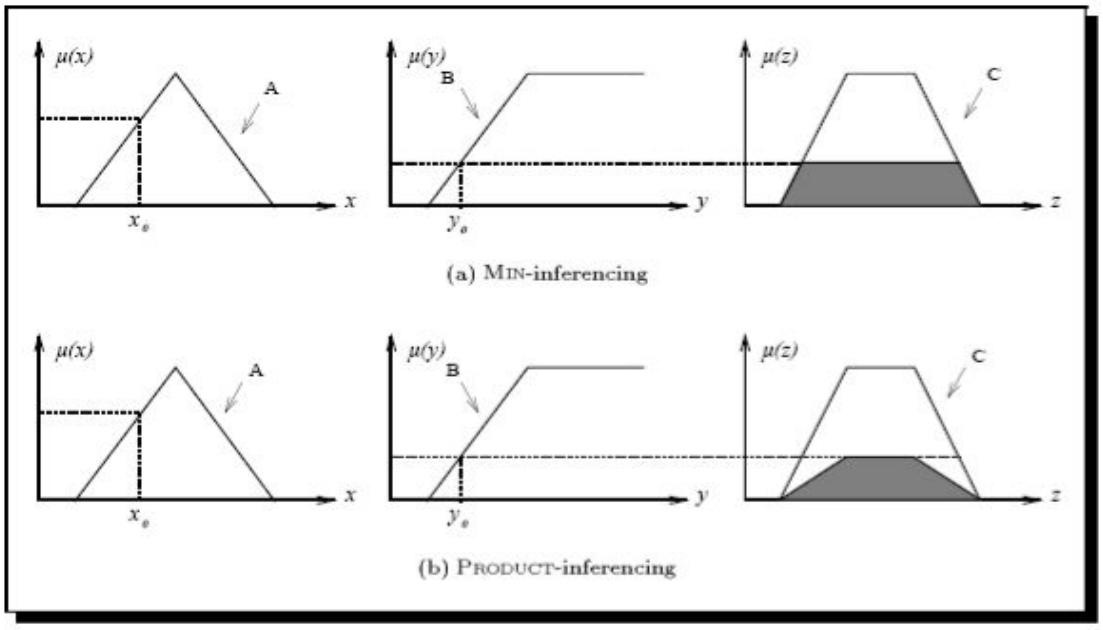

Figure 2.10: Two inferencing methods

EXAMPLE: Let the rule *R* be defined as in the previous example. In Min-inferencing, the variable *z* would be assigned the fuzzy set C' defined by $\mu_{C'}(w)=\alpha \cap \mu_C(w)$ where *w* ranges over all the values the rule conclusion can take. In other words, *z* is assigned an entire set and not a crisp value. In Product-inferencing, the output variable *z* would be assigned the fuzzy set C' defined by $\mu_{C'}(w)=\alpha \times \mu_C(w)$.

2.8.3 Composition

Many rules in the rule base may fire at the same time. If they involve the same output variable, some sort of conflict resolution has to be made. Consider the following rule base:

R_1: IF *x* is A_1 AND *y* is B_1 THEN *z* is C_1

R_2: IF *x* is A_2 AND *y* is B_2 THEN *z* is C_2

In the composition subprocess, all the fuzzy sets assigned to each output variable are combined together to form a single fuzzy set for each output variable

The most common rule of composition is Max-composition. In Max-composition, the combined output fuzzy subset is constructed by taking the point-wise maximum over all the fuzzy subsets assigned to the output by the inference rule. Assuming MIN-inferencing

was used, the fuzzy set C assigned to the output variable z in the rule base above would be defined through the membership function $\mu_C(w)$ given below.

$$\begin{aligned}\mu_C(w) &= \mu_{C_1'}(w) \cup \mu_{C_2'}(w) \\ &= (\alpha_1 \cap \mu_{C_1}(w)) \cup (\alpha_2 \cap \mu_{C_2}(w))\end{aligned} \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

The figure illustrates Max-composition graphically.

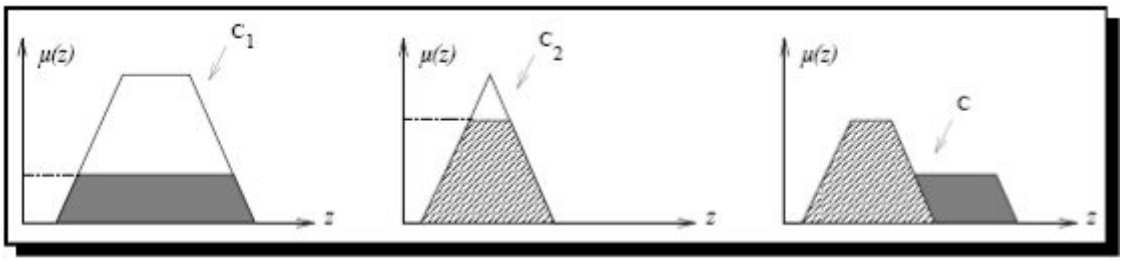

Figure 2.11: An example of MIN-inferencing followed by MAX-composition

Another composition rule is Sum-composition. In Sum-composition the combined output fuzzy subset is constructed by taking the point-wise sum over all the fuzzy subsets assigned to the output variable by the inference rule. As this can result in truth values greater than one, Sum-composition is only used when followed by some kind of defuzzification method that does not have with this odd case.

2.8.4 Defuzzification

Ordinarily, the control action must be in the form of a crisp value. Defuzzification is the process of transforming the fuzzy set assigned to a control output variable into such a crisp value.

There are various methods for defuzzification. The following two are the most prominent in fuzzy control.

Centre of Area Method

Mean of Maxima Method

The centre of area method calculates the centre of gravity of the distribution for the control action. Assuming a discrete universe, this can be written as

$$z^* = (\sum_{j=1}^{q} z_j \mu_C(z_j)) / (\sum_{j=1}^{q} \mu_C(z_j)) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.2)$$

In the formula above, q is the number of quantization levels of the output; z_j is the amount of control output at level j and $\mu_C(z_j)$ represents its membership value in C.

The mean of maxima method generates a crisp control action by averaging the support values whose membership values reaches the maximum. Again assuming a discrete universe, this can be written as

$$z^* = \frac{1}{l} \sum_{j=1}^{l} z_j \quad \text{..... (2.3)}$$

Here l is the number of quantized z values which each their maximum m

CHAPTER 3

ARTIFICIAL NEURAL NETWORK

3.1 Introduction

An artificial neural network (ANN), often just called a "neural network" (NN), is an interconnected group of artificial neurons that uses a mathematical model or computational model for information processing based on a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network.

In more practical terms neural networks are non-linear statistical data modeling tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data.

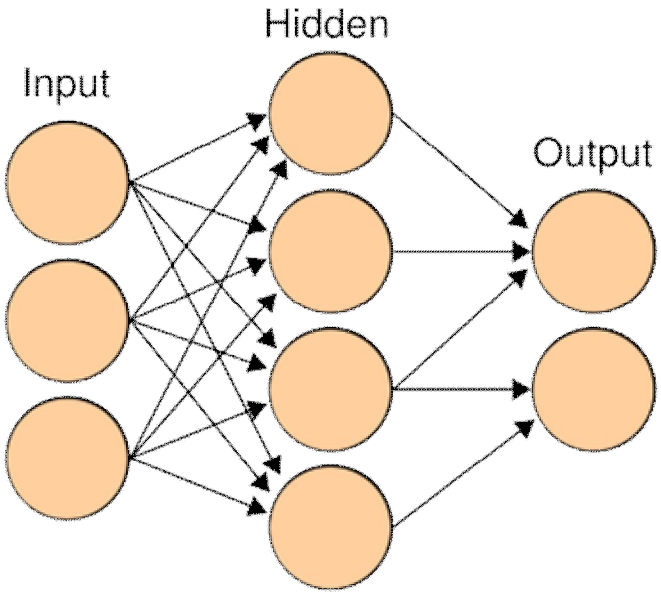

Figure 3.1: Neural network

There is no precise agreed definition among researchers as to what a neural network is, but most would agree that it involves a network of simple processing elements (neurons) which can exhibit complex global behavior, determined by the connections between the processing elements and element parameters. The original inspiration for the technique was from examination of the central nervous system and the

neurons (and their axons, dendrites and synapses) which constitute one of its most significant information processing elements (see Neuroscience). In a neural network model, simple nodes (called variously "neurons", "neurodes", "PEs" ("processing elements") or "units") are connected together to form a network of nodes — hence the term "neural network." While a neural network does not have to be adaptive per se, its practical use comes with algorithms designed to alter the strength (weights) of the connections in the network to produce a desired signal flow.

These networks are also similar to the biological neural networks in the sense that functions are performed collectively and in parallel by the units, rather than there being a clear delineation of subtasks to which various units are assigned (see also connectionism). Currently, the term Artificial Neural Network (ANN) tends to refer mostly to neural network models employed in statistics, cognitive psychology and artificial intelligence. Neural network models designed with emulation of the central nervous system (CNS) in mind are a subject of theoretical neuroscience.

In modern software implementations of artificial neural networks the approach inspired by biology has more or less been abandoned for a more practical approach based on statistics and signal processing. In some of these systems neural networks, or parts of neural networks (such as artificial neurons) are used as components in larger systems that combine both adaptive and non-adaptive elements. While the more general approach of such adaptive systems is more suitable for real-world problem solving, it has far less to do with the traditional artificial intelligence connectionist models. What they do however have in common is the principle of non-linear, distributed, parallel and local processing and adaptation.

3.2 Neural Network Models

Neural network models in artificial intelligence are usually referred to as artificial neural networks (ANNs); these are essentially simple mathematical models defining a function $f : X \to Y$. Each type of ANN model corresponds to a *class* of such functions.

3.2.1 The Network In Artificial Neural Network

The word *network* in the term 'artificial neural network' arises because the function $f(x)$ is defined as a composition of other functions $g_i(x)$, which can further be defined as a composition of other functions. This can be conveniently represented as a network structure, with arrows depicting the dependencies between variables. A widely used type of composition is the *nonlinear weighted sum*, where

$$f(x) = K\left(\sum_i w_i g_i(x)\right),$$

where K is some predefined function, such as the hyperbolic tangent. It will be convenient for the following to refer to a collection of functions g_i as simply a vector $g = (g_1, g_2, \ldots, g_n)$.

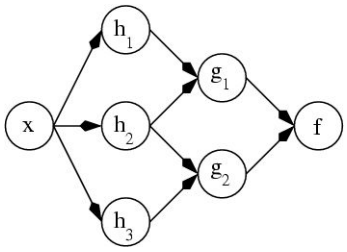

Figure 3.2: ANN dependency graph

This figure depicts such a decomposition of f, with dependencies between variables indicated by arrows. These can be interpreted in two ways.

The first view is the functional view: the input x is transformed into a 3-dimensional vector h, which is then transformed into a 2-dimensional vector g, which is finally transformed into f. This view is most commonly encountered in the context of optimization.

The second view is the probabilistic view: the random variable $F = f(G)$ depends upon the random variable $G = g(H)$, which depends upon $H = h(X)$, which depends upon the random variable X. This view is most commonly encountered in the context of graphical models.

The two views are largely equivalent. In either case, for this particular network architecture, the components of individual layers are independent of each other (e.g., the components of *g* are independent of each other given their input *h*). This naturally enables a degree of parallelism in the implementation.

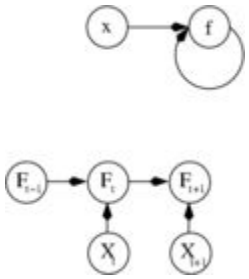

Figure 3.3: Recurrent ANN dependency graph

Networks such as the previous one are commonly called feed forward, because their graph is a directed acyclic graph. Networks with cycles are commonly called recurrent. Such networks are commonly depicted in the manner shown at the top of the figure, where *f* is shown as being dependent upon itself. However, there is an implied temporal dependence which is not shown. What this actually means in practice is that the value of *f* at some point in time *t* depends upon the values of *f* at zero or at one or more other points in time. The graphical model at the bottom of the figure illustrates the case: the value of *f* at time *t* only depends upon its last value. Models such as these, which have no dependencies in the future, are called causal models.

3.3 Learning

However interesting such functions may be in themselves, what has attracted the most interest in neural networks is the possibility of *learning*, which in practice means the following:

Given a specific task to solve, and a class of functions F, learning means using a set of observations, in order to find $f^* \in F$ which solves the task in an optimal sense.

This entails defining a cost function $C : F \rightarrow \mathbb{R}$ such that, for the optimal solution f^*, $C(f^*) \leq C(f) \forall f \in F$ (no solution has a cost less than the cost of the optimal solution).

The cost function *C* is an important concept in learning, as it is a measure of how far away we are from an optimal solution to the problem that we want to solve. Learning algorithms search through the solution space in order to find a function that has the smallest possible cost.

For applications where the solution is dependent on some data, the cost must necessarily be a *function of the observations*; otherwise we would not be modeling anything related to the data. It is frequently defined as a statistic to which only approximations can be made. As a simple example consider the problem of finding the model *f* which minimizes $C = E\left[(f(x) - y)^2\right]$, for data pairs (*x*,*y*) drawn from some distribution $\mathcal{D}$. In practical situations we would only have *N* samples from $\mathcal{D}$and thus, for the above example, we would only minimize $\hat{C} = \frac{1}{N}\sum_{i=1}^{N}(f(x_i) - y_i)^2$. Thus, the cost is minimized over a sample of the data rather than the true data distribution.

When $N \rightarrow \infty$ some form of online learning must be used, where the cost is partially minimized as each new example is seen. While online learning is often used when $\mathcal{D}$is fixed, it is most useful in the case where the distribution changes slowly over time. In neural network methods, some form of online learning is frequently also used for finite datasets.

3.4 Choosing A Cost Function

While it is possible to arbitrarily define some ad hoc cost function, frequently a particular cost will be used either because it has desirable properties (such as convexity) or because it arises naturally from a particular formulation of the problem (i.e., In a probabilistic formulation the posterior probability of the model can be used as an inverse cost). Ultimately, the cost function will depend on the task we wish to perform. The three main categories of learning tasks are overviewed below.

3.5 Learning Paradigms

There are three major learning paradigms, each corresponding to a particular abstract learning task. These are supervised learning, unsupervised learning and reinforcement learning. Usually any given type of network architecture can be employed in any of those tasks.

3.5.1 Supervised Learning

In supervised learning, we are given a set of example pairs $(x, y), x \in X, y \in Y$ and the aim is to find a function f in the allowed class of functions that matches the examples. In other words, we wish to *infer* the mapping implied by the data; the cost function is related to the mismatch between our mapping and the data and it implicitly contains prior knowledge about the problem domain.

A commonly used cost is the mean-squared error which tries to minimize the average error between the network's output, f(x), and the target value y over all the example pairs. When one tries to minimize this cost using gradient descent for the class of neural networks called Multi-Layer Perceptions, one obtains the well-known back propagation algorithm for training neural networks.

Tasks that fall within the paradigm of supervised learning are pattern recognition (also known as classification) and regression (also known as function approximation). The supervised learning paradigm is also applicable to sequential data (e.g., for speech and gesture recognition). This can be thought of as learning with a "teacher," in the form of a function that provides continuous feedback on the quality of solutions obtained thus far.

3.5.2 Unsupervised Learning

In unsupervised learning we are given some data *x*, and the cost function to be minimized can be any function of the data *x* and the network's output, *f*.

The cost function is dependent on the task (what we are trying to model) and our *a priori* assumptions (the implicit properties of our model, its parameters and the observed variables).

As a trivial example, consider the model $f(x) = a$, where a is a constant and the cost $C = (E[x] - f(x))^2$. Minimizing this cost will give us a value of a that is equal to the mean of

the data. The cost function can be much more complicated. Its form depends on the application: For example in compression it could be related to the mutual information between x and y. In statistical modeling, it could be related to the posterior probability of the model given the data. (Note that in both of those examples those quantities would be maximized rather than minimized)

Tasks that fall within the paradigm of unsupervised learning are in general estimation problems; the applications include clustering, the estimation of statistical distributions, compression and filtering.

3.5.3 Reinforcement Learning

In reinforcement learning, data x is usually not given, but generated by an agent's interactions with the environment. At each point in time t, the agent performs an action y_t and the environment generates an observation x_t and an instantaneous cost c_t, according to some (usually unknown) dynamics. The aim is to discover a *policy* for selecting actions that minimizes some measure of a long-term cost, i.e. the expected cumulative cost. The environment's dynamics and the long-term cost for each policy are usually unknown, but can be estimated.

More formally, the environment is modeled as a Markov decision process (MDP) with states $s_1, ..., s_n \in S$ and actions $a_1, ..., a_m \in A$ with the following probability distributions: the instantaneous cost distribution $P(c_t \mid s_t)$, the observation distribution $P(x_t \mid s_t)$ and the transition $P(s_{t+1} \mid s_t, a_t)$, while a policy is defined as conditional distribution over actions given the observations. Taken together, the two define a Markov chain (MC). The aim is to discover the policy that minimises the cost, i.e. the MC for which the cost is minimal.

ANNs are frequently used in reinforcement learning as part of the overall algorithm.

Tasks that fall within the paradigm of reinforcement learning are control problems, games and other sequential decision making tasks.

3.6 Learning Algorithms

Training a neural network model essentially means selecting one model from the set of allowed models (or, in a Bayesian framework, determining a distribution over the set of allowed models) that minimizes the cost criterion. There are numerous algorithms available for training neural network models; most of them can be viewed as a straightforward application of optimization theory and statistical estimation.

Most of the algorithms used in training artificial neural networks are employing some form of gradient descent. This is done by simply taking the derivative of the cost function with respect to the network parameters and then changing those parameters in a gradient-related direction.

Evolutionary methods, simulated annealing, and Expectation-maximization and non-parametric methods are among other commonly used methods for training neural networks.

3.7 Employing Artificial Neural Networks

Perhaps the greatest advantage of ANNs is their ability to be used as an arbitrary function approximation mechanism which 'learns' from observed data. However, using them is not so straightforward and a relatively good understanding of the underlying theory is essential.

Choice of model: This will depend on the data representation and the application. Overly complex models tend to lead to problems with learning.

3.8 Learning Algorithm

There are numerous tradeoffs between learning algorithms. Almost any algorithm will work well with the *correct hyper parameters* for training on a particular fixed dataset. However selecting and tuning an algorithm for training on unseen data requires a significant amount of experimentation.

3.9 Robustness

If the model, cost function and learning algorithm are selected appropriately the resulting ANN can be extremely robust.

With the correct implementation ANNs can be used naturally in online learning and large dataset applications. Their simple implementation and the existence of mostly local dependencies exhibited in the structure allows for fast, parallel implementations in hardware.

3.10Applications

The utility of artificial neural network models lies in the fact that they can be used to infer a function from observations. This is particularly useful in applications where the complexity of the data or task makes the design of such a function by hand impractical.

3.10.1 Real Life Applications

The tasks to which artificial neural networks are applied tend to fall within the following broad categories:

- Function approximation, or regression analysis, including time series prediction and modeling.
- Classification, including pattern and sequence recognition, novelty detection and sequential decision making.
- Data processing, including filtering, clustering, blind source separation and compression.

Application areas include system identification and control (vehicle control, process control), game-playing and decision making (backgammon, chess, racing), pattern recognition (radar systems, face identification, object recognition and more), sequence recognition (gesture, speech, handwritten text recognition), medical diagnosis, financial applications, data mining (or knowledge discovery in databases, "KDD"), visualization and e-mail spam filtering.

3.11 Neural Network Software

Neural network software is used to simulate, research, develop and apply artificial neural networks, biological neural networks and in some cases a wider array of adaptive systems.

3.12 Types of Neural Networks:

Some type of neural networks given below:

3.12.1 Feed Forward Neural Network

The feed forward neural network was the first and arguably simplest type of artificial neural network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network.

3.12.2 Radial Basis Function (RBF) Network

Radial Basis Functions are powerful techniques for interpolation in multidimensional space. A RBF is a function which has built into a distance criterion with respect to a centre. Radial basis functions have been applied in the area of neural networks where they may be used as a replacement for the sigmoid hidden layer transfer characteristic in multi-layer perceptions. RBF networks have two layers of processing: In the first, input is mapped onto each RBF in the 'hidden' layer. The RBF chosen is usually a Gaussian. In regression problems the output layer is then a linear combination of hidden layer values representing mean predicted output. The interpretation of this output layer value is the same as a regression model in statistics. In classification problems the output layer is typically a sigmoid function of a linear combination of hidden layer values, representing a posterior probability. Performance in both cases is often improved by shrinkage techniques, known as ridge regression in classical statistics and known to correspond to a prior belief in small parameter values (and therefore smooth output functions) in a Bayesian framework.

RBF networks have the advantage of not suffering from local minima in the same way as multi-layer perceptrons. This is because the only parameters that are adjusted in the learning process are the linear mapping from hidden layer to output layer. Linearity ensures that the error surface is quadratic and therefore has a single easily found minimum. In regression problems this can be found in one matrix operation. In classification problems the fixed non-linearity introduced by the sigmoid output function is most efficiently dealt with using iteratively re-weighted least squares.

RBF networks have the disadvantage of requiring good coverage of the input space by radial basis functions. RBF centers are determined with reference to the distribution of the input data, but without reference to the prediction task. As a result, representational resources may be wasted on areas of the input space that are irrelevant to the learning task. A common solution is to associate each data point with its own centre, although this can make the linear system to be solved in the final layer rather large, and requires shrinkage techniques to avoid over fitting.

Associating each input datum with an RBF leads naturally to kernel methods such as Support Vector Machines and Gaussian Processes (the RBF is the kernel function). All three approaches use a non-linear kernel function to project the input data into a space where the learning problem can be solved using a linear model. Like Gaussian Processes, and unlike SVMs, RBF networks are typically trained in a Maximum Likelihood framework by maximizing the probability (minimizing the error) of the data under the model. SVMs take a different approach to avoiding over fitting by maximizing instead a margin. RBF networks are outperformed in most classification applications by SVMs. In regression applications they can be competitive when the dimensionality of the input space is relatively small.

3.12.3 Kohonen Self-Organizing Network

The *self-organizing map* (SOM) invented by Teuvo Kohonen uses a form of unsupervised learning. A set of artificial neurons learn to map points in an input space to coordinates in an output space. The input space can have different dimensions and topology from the output space and the SOM will attempt to preserve these.

3.12.4 Recurrent Network

Contrary to feed forward networks, recurrent neural networks (RNs) are models with bi-directional data flow. While a feed forward network propagates data linearly from input to output, RNs also propagate data from later processing stages to earlier stages.

3.12.5 Simple Recurrent Network

A *simple recurrent network* (SRN) is a variation on the multi-layer perceptron, sometimes called an "Elman network" due to its invention by Jeff Elman. A three-layer network is used, with the addition of a set of "context units" in the input layer. There are connections from the middle (hidden) layer to these context units fixed with a weight of one. At each time step, the input is propagated in a standard feed-forward fashion, and then a learning rule (usually back-propagation) is applied. The fixed back connections result in the context units always maintaining a copy of the previous values of the hidden units (since they propagate over the connections before the learning rule is applied). Thus the network can maintain a sort of state, allowing it to perform such tasks as sequence-prediction that is beyond the power of a standard multi-layer perceptron.

In a *fully recurrent network*, every neuron receives inputs from every other neuron in the network. These networks are not arranged in layers. Usually only a subset of the neurons receive external inputs in addition to the inputs from all the other neurons, and another disjunct subset of neurons report their output externally as well as sending it to all the neurons. These distinctive inputs and outputs perform the function of the input and output layers of a feed-forward or simple recurrent network, and also join all the other neurons in the recurrent processing.

3.12.6 Hopfield Network

The Hopfield network is a recurrent neural network in which all connections are symmetric. Invented by John Hopfield in 1982, this network guarantees that its dynamics will converge. If the connections are trained using Hebbian learning then the Hopfield network can perform as robust content-addressable memory, resistant to connection alteration.

3.12.7 Echo State Network

The Echo State Network (ESN) is a recurrent neural network with a sparsely connected random hidden layer. The weights of output neurons are the only part of the network that can change and be learned. ESN are good to (re)produce temporal patterns.

3.12.8 Long Short Term Memory Network

The Long short term memory is an artificial neural net structure that unlike traditional RNNs doesn't have the problem of vanishing gradients. It can therefore use long delays and can handle signals that have a mix of low and high frequency components.

3.12.9 Stochastic Neural Networks

A stochastic neural network differs from a typical neural network in the fact that it introduces random variations into the network. In a probabilistic view of neural networks, such random variations can be viewed as a form of statistical sampling, such as Monte Carlo sampling.

3.12.10 Boltzmann Machine

The Boltzmann machine can be thought of as a noisy Hopfield network. Invented by Geoff Hinton and Terry Sejnowski in 1985, the Boltzmann machine is important because it is one of the first neural networks to demonstrate learning of latent variables (hidden units). Boltzmann machine learning was at first slow to simulate, but the contrastive divergence algorithm of Geoff Hinton (circa 2000) allows models such as Boltzmann machines and *products of experts* to be trained much faster.

3.12.11 Modular Neural Networks

Biological studies showed that the human brain functions not as a single massive network, but as a collection of small networks. This realisation gave birth to the concept of modular neural networks, in which several small networks cooperate or compete to solve problems.

A committee of machines (CoM) is a collection of different neural networks that together "vote" on a given example. This generally gives a much better result compared to other

neural network models. In fact in many cases, starting with the same architecture and training but using different initial random weights gives vastly different networks. A CoM tends to stabilize the result.

The CoM is similar to the general machine learning *bagging* method, except that the necessary variety of machines in the committee is obtained by training from different random starting weights rather than training on different randomly selected subsets of the training data.

3.12.12 Associative Neural Network (ASNN)

The ASNN is an extension of the *committee of machines* that goes beyond a simple/weighted average of different models. ASNN represents a combination of an ensemble of feed-forward neural networks and the k-nearest neighbor technique (kNN). It uses the correlation between ensemble responses as a measure of **distanc**e amid the analysed cases for the kNN. This corrects the bias of the neural network ensemble. An associative neural network has a memory that can coincide with the training set. If new data becomes available, the network instantly improves its predictive ability and provides data approximation (self-learn the data) without a need to retrain the ensemble. Another important feature of ASNN is the possibility to interpret neural network results by analysis of correlations between data cases in the space of models.

3.13 Other Types Of Networks

These special networks do not fit in any of the previous categories.

3.13.1 Holographic Associative Memory

Holographic associative memory represents a family of analog, correlation-based, associative, stimulus-response memories, where information is mapped onto the phase orientation of complex numbers operating.

3.13.2 Instantaneously Trained Networks

Instantaneously trained neural networks (ITNNs) were inspired by the phenomenon of short-term learning that seems to occur instantaneously. In these networks the weights of the hidden and the output layers are mapped directly from the training vector data. Ordinarily, they work on binary data, but versions for continuous data that require small additional processing are also available.

3.13.3 Spiking Neural Networks

Spiking neural networks (SNNs) are models which explicitly take into account the timing of inputs. The network input and output are usually represented as series of spikes (delta function or more complex shapes). SNNs have an advantage of being able to continuously process information. They are often implemented as recurrent networks.

Networks of spiking neurons -- and the temporal correlations of neural assemblies in such networks -- have been used to model figure/ground separation and region linking in the visual system (see e.g. Reitboeck et.al.in Haken and Stadler: Synergetics of the Brain. Berlin, 1989).

Gerstner and Kistler have a freely-available online textbook on Spiking Neuron Models.

Spiking neural networks with axonal conduction delays exhibit polychronization, and hence could have a potentially unlimited memory capacity.

In June 2005 IBM announced construction of a Blue Gene supercomputer dedicated to the simulation of a large recurrent spiking neural network.

3.13.4 Dynamic Neural Networks

Dynamic neural networks not only deal with nonlinear multivariate behaviour, but also include (learning of) time-dependent behaviour such as various transient phenomena and delay effects.

3.13.5 Cascading Neural Networks

Cascade-Correlation is an architecture and supervised learning algorithm developed by Scott Fahlman and Christian Lebiere. Instead of just adjusting the weights in a network of fixed topology, Cascade-Correlation begins with a minimal network, then automatically

trains and adds new hidden units one by one, creating a multi-layer structure. Once a new hidden unit has been added to the network, its input-side weights are frozen. This unit then becomes a permanent feature-detector in the network, available for producing outputs or for creating other, more complex feature detectors. The Cascade-Correlation architecture has several advantages over existing algorithms: it learns very quickly, the network determines its own size and topology, it retains the structures it has built even if the training set changes, and it requires no back-propagation of error signals through the connections of the network.

3.13.6 Neuro-Fuzzy Networks

A neuro-fuzzy network is a fuzzy inference system in the body of an artificial neural network. Depending on the *FIS* type, there are several layers that simulate the processes involved in a *fuzzy inference* like fuzzification, inference, aggregation and defuzzification. Embedding an *FIS* in a general structure of an *ANN* has the benefit of using available *ANN* training methods to find the parameters of a fuzzy system.

3.14 Theoretical Properties

Some theoretical properties are given below

3.14.1 Capacity

Artificial neural network models have a property called 'capacity', which roughly corresponds to their ability to model any given function. It is related to the amount of information that can be stored in the network and to the notion of complexity.

3.14.2 Convergence

Nothing can be said in general about convergence since it depends on a number of factors. Firstly, there may exist many local minima. This depends on the cost function and the model. Secondly, the optimization method used might not be guaranteed to converge when far away from a local minimum. Thirdly, for a very large amount of data or parameters, some methods become impractical. In general, it has been found that

theoretical guarantees regarding convergence are not always a very reliable guide to practical application.

3.14.3 Generalization and Statistics

In applications where the goal is to create a system that generalizes well in unseen examples, the problem of overtraining has emerged. This arises in over complex or over specified systems when the capacity of the network significantly exceeds the needed free parameters. There are two schools of thought for avoiding this problem: The first is to use cross-validation and similar techniques to check for the presence of overtraining and optimally select hyper parameters such as to minimize the generalization error. The second is to use some form of *regularization*. This is a concept that emerges naturally in a probabilistic (Bayesian) framework, where the regularization can be performed by putting a larger prior probability over simpler models; but also in statistical learning theory, where the goal is to minimize over two quantities: the 'empirical risk' and the 'structural risk', which roughly correspond to the error over the training set and the predicted error in unseen data due to over fitting.

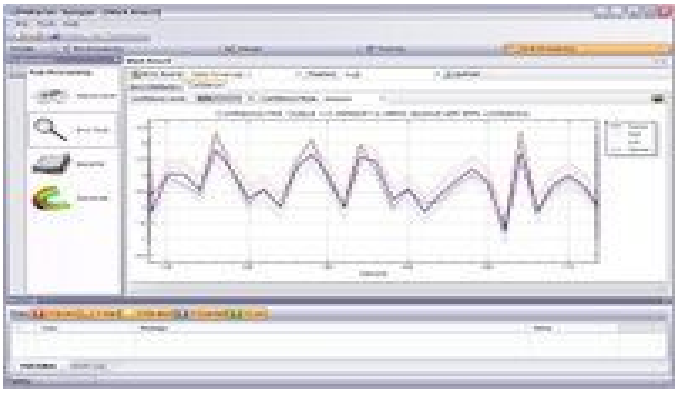

Figure: 3.4 Confidence analysis of a neural network

Supervised neural networks that use an MSE cost function can use formal statistical methods to determine the confidence of the trained model. The MSE on a validation set can be used as an estimate for variance. This value can then be used to calculate the confidence interval of the output of the network, assuming a normal distribution. A

confidence analysis made this way is statistically valid as long as the output probability distribution stays the same and the network is not modified.

By assigning a softmax activation function on the output layer of the neural network (or a softmax component in a component-based neural network) for categorical target variables, the outputs can be interpreted as posterior probabilities. This is very useful in classification as it gives a certainty measure on classifications.

CHAPTER: 4

NEURO FUZZY MODELS

4.1 Introduction

It immediately comes to mind, when looking at a neural network that the activation functions look like fuzzy membership functions. Consider a standard rule base for a fuzzy proportional controller with the error h as input and a control signal x with singleton membership functions as the output, If h is Pos then x is -100 If h is Zero then x is 0 If h is Nag then x is -100. The network has an input layer, one hidden layer, and one output layer. The input node connects to the neurons in the hidden layer; this corresponds to the if-part of the rules. Each neuron only consists of an activation function; there is no summation, because each neuron has only one input. The singleton control signals appear as weights on the outputs from the neurons.

The one neuron in the output layer, with a rather odd appearance, calculates the weighted average corresponding to the if part of the rules. Each neuron only consist of an activation function, there is no summation, because each neuron has only one input. The singleton control signal appear as weight on the outputs from the neurons. The one neuron in the output layer, with a rather old appearance, calculate the weight av corresponds to the centre of gravity the network can be generalized to multi-input-multi-output control, but then the diagram becomes very busy. Back propagation applies to this network since all layers are differentiable. Two possibilities for learning are apparent. One is to adjust the weights in the output layer, l=h=, all the singletons zl until the error is minimized. The other is to adjust the shape of the membership functions, provided they are parametric. The network can be described as a feed forward network with an input layer, a single hidden layer, and an output layer consisting of a single unit. The network performs a nonlinear mapping from the input layer to the hidden layer, followed by a linear mapping from the hidden layer to the output layer. Exactly such a topology occurs in radial bases function networks; the hidden unit provides a basis for the input patterns and ther functions radialy surround a particular data point. Radial basis function networks are used for curve-fitting in a multi-dimensional space. This is also called function

approximation, and learning is equivalent to finding a function that best fits the training data. In its *strict* sense the function is constrained to pass through all the training data points. The radial basis functions technique consists of choosing a function F,

$$F(u) = W^T f(\| u - u_\lambda \|)$$

$$= \begin{bmatrix} w1 & w2 & .... & w_\lambda \end{bmatrix} \begin{bmatrix} f(\| u - u1 \|) \\ f(\| u - u2 \|) \\ ..... \\ f(\| u - u_K \|) \end{bmatrix} \qquad (1)$$

Here u, $u \in R$is a vector of inputs, $u_k \in R$(k=1,2,…….K) are vectors of training data, $w \in R$ is the vector of weights, $f(\| u - u_k \|)$ is a set of (nonlinear) radial basis functions, and ||.|| is a norm, usually the Euclidean. The known data points u_k are taken to be the centers of the radial basis functions. The activation level of a function f (u, u_k) is maximum when the input u is at the center u(k), of the function, as demonstrated by the next example.

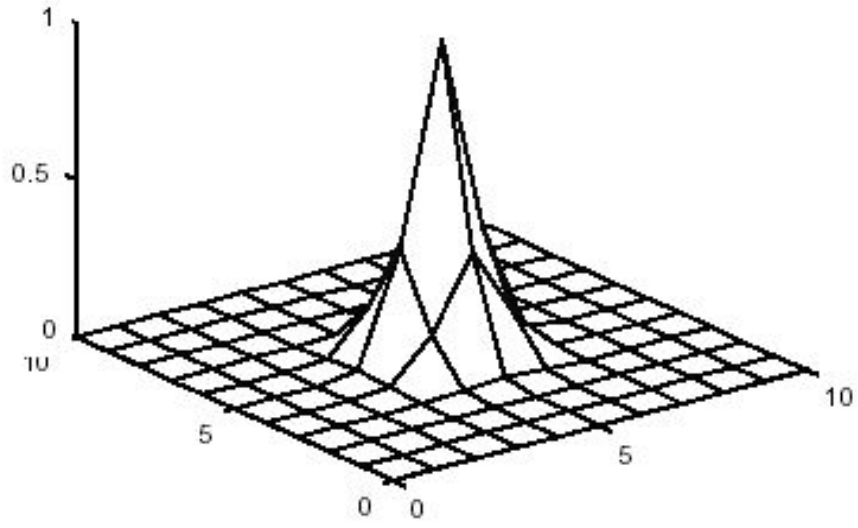

Figure 4.1: A Gaussian radial basis function

must satisfy the following set of equations

$$\begin{bmatrix} w_1 & w_2 & .... & w_k \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1k} \\ f_{21} & f_{22} & \cdots & f_{2k} \\ \cdots & \cdots & \cdots & \cdots \\ f_{k1} & f_{k2} & \cdots & f_{kk} \end{bmatrix} = \begin{bmatrix} d_1 & d_2 & \ldots & d_k \end{bmatrix} \quad (2)$$

In matrix notation

$$w^T \phi = d^T \quad (3)$$

where the vector d represents the desired response vector, and

$$f_{jk} = f(\|u_j - u_k\|), j = 1,2....K \quad (4)$$

The matrix $\Phi = \{f_{jk}\}$ is called the interpolation matrix. For a class of radial basis functions, Gaussian functions for instance, the interpolation matrix is invertible (it is positive definite). Provided the data points are all distinct, then we can solve for the weights directly, obtaining

$$w^T = d^{T\phi^{-1}} \quad (5)$$

Although in theory this means we can solve the strict interpolation problem where the function passes through all training points Xn, in practice we cannot, if the interpolation matrix is close to singular. The performance of the network depends only little on the type of function f(.) according to theoretical investigations and practical experiences (Powell in Haykin, 1994). The performance may be improved by adjustments of the centre and the shape of the activation functions. Generally the radial basis function networks enjoy faster convergence than back-propagation networks.

4.1 Adaptive Neurofuzzy Inference System

ANFIS (Adaptive Neuro Fuzzy Inference System) is an architecture which is functionally quivalent to a Sugeno type fuzzy rule base. Under certain minor constraints the ANFIS architecture is also equivalent to a radial basis function network. Loosely speaking ANFIS is a method for using an existing rule base with a learning algorithm based on a collection of training data. This allows the rule base to adapt. The network in Fig. 8 may be extended by assigning a linear function to the output weight of each neuron,

$$w_k = a_k^T U = B_K, k = 1,2.K \tag{6}$$

Where $a_k \in R^m$ is a parameter vector and en is a scalar parameter. The network is then equivalent to a first order Sugeno type fuzzy rule base. The requirements for the radial basis function network to be equivalent to a fuzzy rule base are summarized in the following:

- Both must use the same aggregation method (weighted average or weighted sum) to derive their overall outputs.
- The number of activation functions must be equal to the number of fuzzy if-then rules. When there are several inputs in the rule base, each activation function must be equal to a composite input membership function. One way to achieve this is to employ Gaussian membership functions with the same variance in the rule base, and apply product for the DQG operation. The multiplication of the Gaussian membership functions becomes a multi-dimensional Gaussian radial basis function.
- Corresponding activation functions and fuzzy rules should have the same functions on the output side of the neurons and rules respectively.

If the training data are contained in a small region of the input space, the centers of the neurons in the hidden layer can be concentrated within the region and sparsely cover the remaining area. Thus only a local model will be formed and if the test data lie outside the

region, the performance of the network will be poor. On the other hand, if one distributes the basis function centers evenly throughout the input space, the number of neurons depends exponentially on the dimension of the input space.

4.1.2 ANFIS Architecture

Without loss of generality we assume two inputs, x4 and x5> and one output, |. Assume for now a first order Sugeno type of rule base with the following two rules

$$\text{If } u_1 \text{ is } A_1 \text{ and } u_2 \text{ is } B_1 \text{ then } y_1 = c_{11}u_1 + c_{12}u_2 + c_{10}$$
$$\text{If } u_1 \text{ is } A_2 \text{ and } u_2 \text{ is } B_2 \text{ then } y_2 = c_{21}u_1 + c_{22}u_2 + c_{20}$$

Incidentally, this fuzzy controller could interpolate between two linear controllers depending on the current state. If the firing strengths of the rules are α_1 and α_2 respectively, for two particular values of the inputs x4 and x5> then the output is computed as a weighted average.

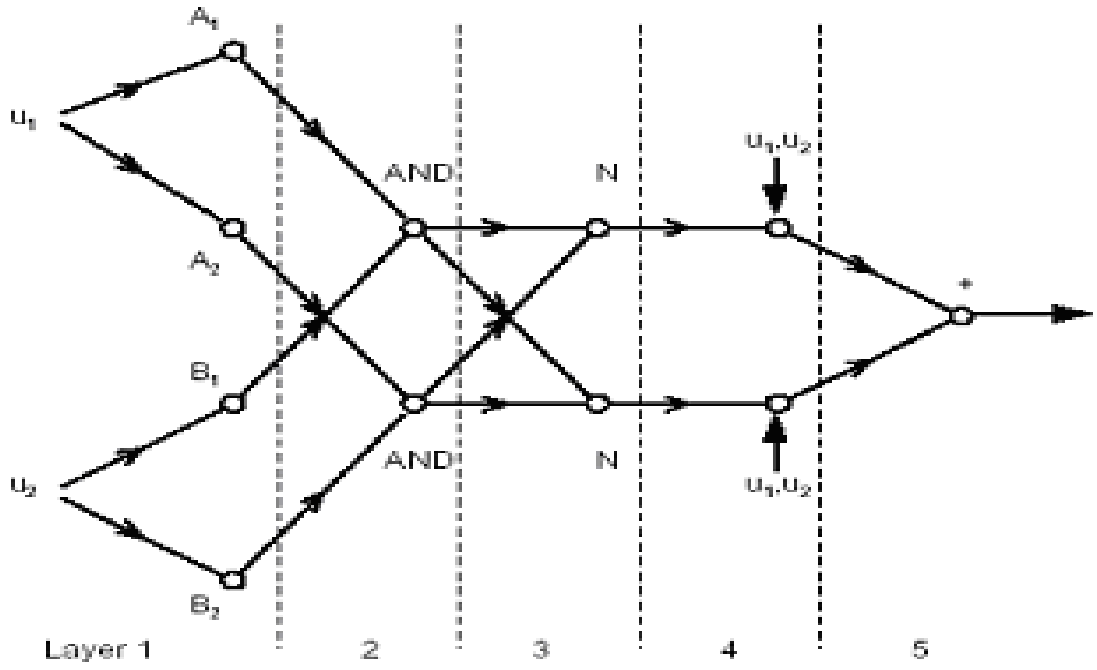

Figure 4.2: ANFIS Network

$$y = \frac{\alpha_1 y_1 + \alpha_2 y_2}{\alpha_1 + \alpha_2} = \overline{\alpha}_1 y_1 + \overline{\alpha}_2 y_2 \qquad (7)$$

The corresponding ANFIS network is shown in Fig. 4.2. A description of the layers in the network follows.

1. Each neuron l in layer 1 is adaptive with a parametric activation function. Its output is he grade of membership to which the given input satisfies the membership function, i.e., _$4 (x4), _%4(x5), _$5 (x4), or _%5(x5). An example of membership unction is the generalized bell function.

$$\mu(x) = \frac{1}{1 + \left| \frac{x - c}{a} \right|^{2b}} \qquad (8)$$

where {a,b,c)is the parameter set. As the values of the parameters change, the shape of the bell-shaped function varies. Parameters in that layer are called premise parameters.

2. Every node in layer 2 is a fixed node, whose output is the product of all incoming signals. In general, any other fuzzy AND operation can be used. Each node output represents the firing strength of the rule.
3. Every node in layer 3 is a fixed node which calculates the ratio of the lth rule's firing strength relative to the sum of all rule's firing strengths,

$$\overline{\alpha}_i = \frac{\alpha_i}{\alpha_i + \alpha_2} \quad , \quad \text{i} = 1,2 \qquad (9)$$

The result is a normalized firing strength.

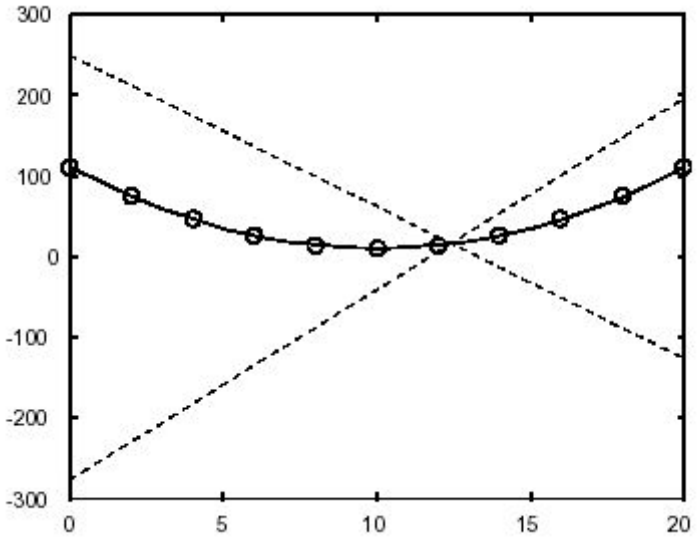

Figure 4.3: Approximation of data points (o) by an ANFIS network (solid).Two rules interpolate between two lines (dotted**)**

4. Every node in layer 4 is an adaptive node with a node output

$$\bar{\alpha}_i y_i = \bar{\alpha}_i (c_{i1} u_1 + c_{i2} u_2 + c_{i0}) \quad \text{i= 1,2} \qquad (10)$$

Where _1 is the normalized firing strength from layer 3 and ifl4> fl5> fl3j is the parameter set of this node. Parameters in this layer are called consequent parameters.

5. Every node in layer 5 is a fixed node which sums all incoming signals.2

It is straight forward to generalize the ANFIS architecture in Fig. 4.3 to a rule base with more than two rules.

4.1.3 The ANFIS learning algorithms

When the premise parameters are fixed, the overall output is a linear combination of the consequent parameters. In symbols, the output | can be written as

$$y = \frac{\alpha_1}{\alpha_1 + \alpha_2} y_1 + \frac{\alpha_2}{\alpha_1 + \alpha_2} y_2$$
$$= \overline{\alpha_1}(c_{11}u_1 + c_{12}u_2 + c_{10}) + \overline{\alpha_2}c_{21}u_1 + c_{22}u_2 + c_{20})$$
$$= (\overline{\alpha_1}u_1)c_{11} + (\overline{\alpha_1}u_2)c_{12} + \overline{\alpha_1}c_{10} + (\overline{\alpha_2}u_2)c_{21} + (\overline{\alpha_2}u_2)c_{22} + \overline{\alpha_2}c_{20}$$

(11)

Which is linear in the consequent parameters ii(i=1,2;j=0,1,2). A hybrid algorithm adjusts the consequent parameters flm in a forward pass and the premise parameters $\{a_i, b_i, c_i\}$ in a backward pass. In the forward pass the network inputs propagate forward until layer 4, where the consequent parameters are identified by the least-squares method. In the backward pass, the error signals propagate backwards and the premise parameters are updated by gradient descent.

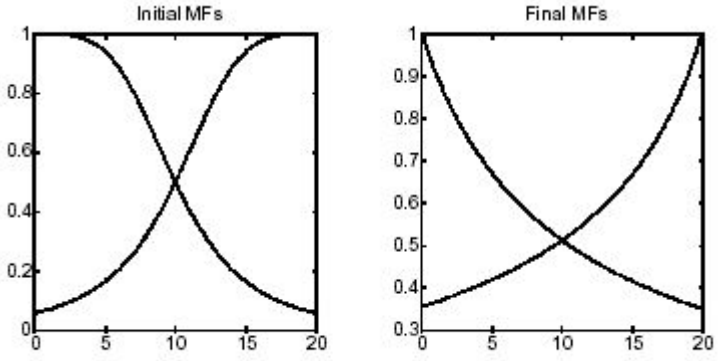

Figure 4.4 Membership functions before (left) and after (right) learning

The hybrid learning rule, a computational speedup may be possible by using variants of the gradient method or other optimization techniques on the premise parameters. Since ANFIS and radial basis function networks (RBFNs) are functionally equivalent under some minor conditions (p 17), a variety of learning methods can be used for both of them.

CHAPTER 5

PROBLEM FORMULATION

5.1 The Two-Area Model

A block diagram of a two-area interconnected system for the uncontrolled case is shown in Figure 1. The state variables model for the system is

$$\dot{x}(t) = Ax(t) + Bu(t) + Ld(t) \tag{1}$$

where A is system matrix, B is input distribution matrix, L is disturbance distribution matrix, x(t) is state vector, u(t) is control vector and d(t) is disturbance vector of load changes.

$$x(t) = [\triangle f1, \triangle Pg1, \triangle Pv1, \triangle Ptie, 12, \triangle f2, \triangle Pg2, \triangle Pv2]^T$$

$$u(t) = [u1, u2]^T = [\triangle Pc1, \triangle Pc2]^T$$

$$d(t) = [\triangle Pd1, \triangle Pd2]^T$$

Since, the effectiveness of LFC is judged in terms of area control errors, ACE, the system output is given by

$$y(t) = \begin{bmatrix} y1(t) \\ y2(t) \end{bmatrix} = \begin{bmatrix} ACE_1 \\ ACE_2 \end{bmatrix} = Cx(t) \tag{2}$$

$$ACEi = \triangle Ptic, i + bi \triangle fi \tag{3}$$

where y(t) is the output vector, ACEi is area i control error, bi is area **i** frequency bias constant, Afi is area i frequency change, APtie,i is the change in tie-line power and C is the output matrix.

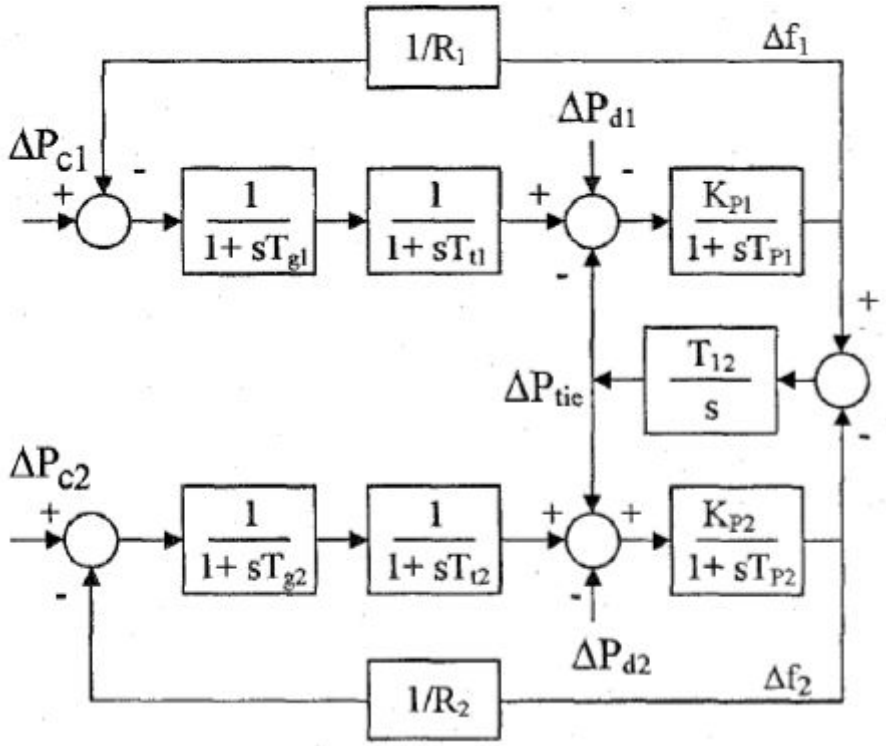

Figure.5.1: Two Area Interconnected System

5.2 Conventional PI Controller

The task of load frequency controller is to generate a control signal u that maintains system frequency and tie-line interchange power at predetermined values. The block diagram of the PI controller is shown in Figure 2. The control inputs uI and u2 are constructed as follows

$$ui = -Ki\int_0^{\tau}(ACEi)dt \quad -Ki\int_0^{\tau}(\triangle Ptie,i + bi\triangle fi)dt \tag{4}$$

Taking the derivative of equation (4) yields

$$\dot{u}i = Ki\int_0^{\tau}(ACEi)dt \quad -Ki\int_0^{\tau}(\triangle Ptie,i + bi\triangle fi)dt \tag{5}$$

or in matrix form

$$\dot{u} = -K_I Cx \tag{6}$$

Where

$$K_I = \begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix} \tag{7}$$

combining equations (1) and (6) yields

$$\dot{x}_c = A_c x_c + L_c d \tag{8}$$

Where

$$x_c = \begin{bmatrix} x \\ u \end{bmatrix}; A_c \quad \begin{bmatrix} A & B \\ -K_1 C & 0 \end{bmatrix}; L_c \quad \begin{bmatrix} L \\ 0 \end{bmatrix} \quad = \tag{9}$$

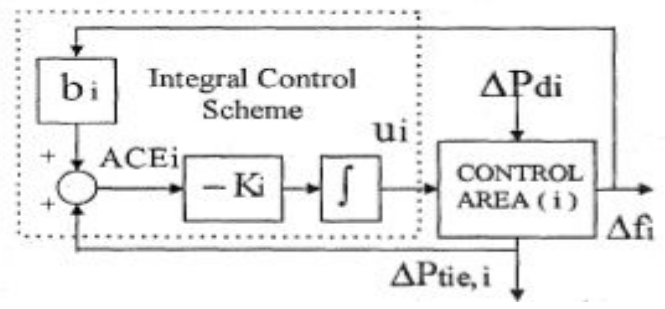

Figure 5.2: Conventional PI Controller Installed on i" Area

5.3 Optimal PI Controller

The integral of area control errors are considered as states. This will drive the area control errors to zero at steady state.

Defining these new states as vector q(t) yields

$$\dot{x}_a = A_a x_a + B_a u + L_a d \tag{10}$$

$$y_a(t) = C_a x_a(t) \tag{11}$$

Where

$$x_a(t) = \begin{bmatrix} x(t) \\ q(t) \end{bmatrix}; y_a(t) = \begin{bmatrix} y(t) \\ q(t) \end{bmatrix}; x_a(0) = \begin{bmatrix} x(0) \\ q(0) \end{bmatrix} \tag{12}$$

$$A_a = \begin{bmatrix} A & 0 \\ C & 0 \end{bmatrix}; B_a \begin{bmatrix} B \\ 0 \end{bmatrix}; L_a \begin{bmatrix} L \\ 0 \end{bmatrix}; C_a \begin{bmatrix} C & 0 \\ 0 & 1 \end{bmatrix} \tag{13}$$

The control vector u is generated by feeding back all states through a constant gain matrix K.

$$u = -Kxa(t) \tag{14}$$

The gain matrix K is determined by minimizing the following quadratic performance index

$$J = \frac{1}{2}\int_0^\infty (x_a{}^T Q x_a + u^T R u) dt \tag{15}$$

Where Q and R are positive semi definite and positive definite constant matrices defined through the minimization of ACEi, ACEi.dt and control vector u excursions**.** This is the state-space optimal regulator problem [lo]. K is obtained from the solution of the algebraic matrix Riccati equation. The control law of equation (14) may be partitioned into its proportional and integral components as follows:

.

$$u = K_p x + K_1 q \tag{16}$$

where Kp and KI are the proportional and integral feedback gain matrices, respectively That is why this LFC is referred to as Optimal PI State Feedback Controller**.**

5.4 Fuzzy Control Schemes

Fuzzy set theory has been recently applied to the LFC problem. In this thesis it has been proposed a fuzzy controller in which the change of frequency and its rate have been used as inputs. Indulkar et. proposed another fuzzy LFC in which he used the area control error and its change as inputs. The block diagram of Hsu fuzzy controller is shown in Figure 3. The input signals are first expressed in some linguistic variables using fuzzy set notations such as large positive (LP), medium positive *(MP),* small positive (SP), very small (VS), small negative (SN), medium negative (MN), and large negative(LN). A set of decision rules, also expressed in linguistic variables, are established to relate input signals to the output (control) signal. These decision rules, which are summarized by a fuzzy relation matrix using membership functions, form the basis of the fuzzy logic operations performed by the fuzzy controller. Table 1 shows the decision rules that has been proposed by Hsu .

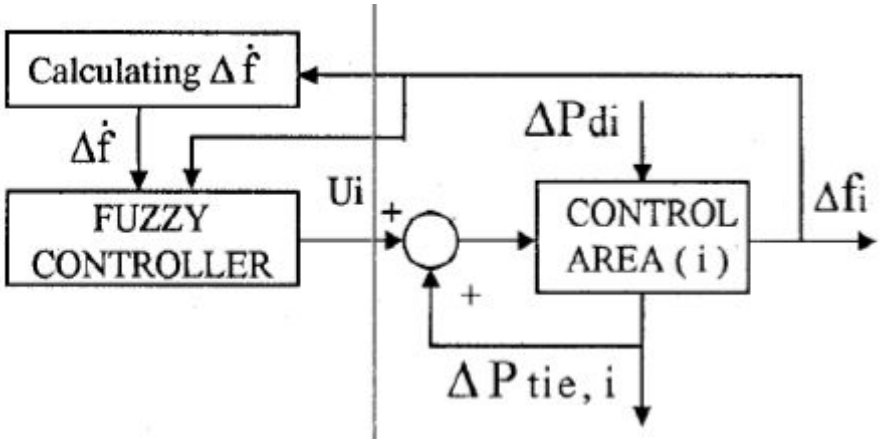

Figure5.3: Fuzzy Load Frequency Controller Proposed by Hsu

		Frequency Deviation Rate ($\dot{\Delta f}$)						
		LN	MN	SN	VS	SP	MP	LP
Frequency Deviation (Δf)	LP	VS*	SP	MP	LP	LP	LP	LP
	MP	SN	VS	SP	MP	MP	LP	LP
	SP	MN	SN	VS	SP	SP	MP	LP
	VS	MN	MN	SN	VS	SP	MP	MP
	SN	LN	MN	SN	SN	VS	SP	MP
	MN	LN	LN	MN	MN	SN	VS	SP
	LN	LN	LN	LN	LN	MN	SN	VS

Table 5.1: Decision rule of Hsu

* Rule 1

For example, rule 1 in 'ble 1 is:

IF Δf is LP and Δf is LN THEN U is VS (17)

Once the membership values for the controller output have been computed, a suitable algorithm must be employed to determine the controller output signal from these membership values.

5.5 Neural Network Based Adaptive Gain Scheduling

The main drawback of fixed gain controllers is that their performance deteriorates as a result of changes in system operating conditions. In order to maintain desired quality of the system dynamics over a wide range of operating conditions, it is highly desirable to be able to adapt the controller gains according to the on-line information. Djukanovic et. al used Artificial Neural Networks (ANN) for the adaptation of elements of the gain matrix K of the state space optimal load frequency controller. In his approach, three parameters that represent system operating conditions have been monitored and used as inputs to the neural net based adaptive controller. These parameters are power system time constant, Tp, synchronizing power coefficient, T12 and frequency bias setting b. The parameters depend on the load frequency characteristic, D *(see* Appendix), which is load dependent that must be determined at every load pattern to precisely monitor the

parameters. These parameters activate many individual neural-nets trained with supervised learning. The training set of patterns are generated by solving the state-space optimal regulator problem for different values of parameters Tp, T12 and b**.** Network training is slow if it contains a large number of patterns. Furthermore, a criterion for selecting the optimum number of nodes in the hidden layers of the net should be developed, which significantly affects learning rate and classification performance.

5.6 Adaptive Fuzzy Gain Scheduling

A Sugeno type fuzzy inference system to replace the neural nets is proposed in this thesis. The advantage is that the design is simpler and fewer training patterns are needed. Sugeno type fuzzy inference system **is** extremely well suited to the task of smoothly interpolating linear gains across the input space. So, we could build a fuzzy system that switches smoothly between several linear controllers as a non-linear system moves around in its operating space. Consequently, two fuzzy based adaptive load frequency controllers are proposed in this paper. The first is an optimal load frequency controller with fuzzy gain scheduling and the second is a conventional integral load frequency controller with fuzzy gain scheduling. Optimal Controller with Fuzzy Gain Scheduling The linear optimal PI controller is modified in such a way that its feedback gain matrix K elements are adapted by means of a fuzzy gain scheduler according to on-line system information (Tp, T12 and b) as shown in Figure 4. Unlike the neural-net approach wherein a large number of training patterns are required, we need fewer numbers of patterns in the fuzzy approach (only 27 combinations of system monitored parameters are used). Each monitored parameter 148 is divided into three fuzzy sets, Small, Medium and Large, represented by the membership functions shown in Figure *5*.

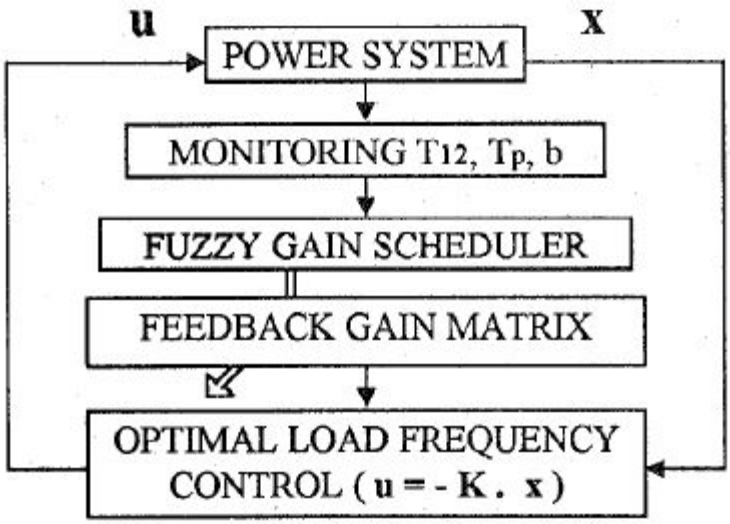

Figure5.4: Optimal controller with fuzzy gain scheduler

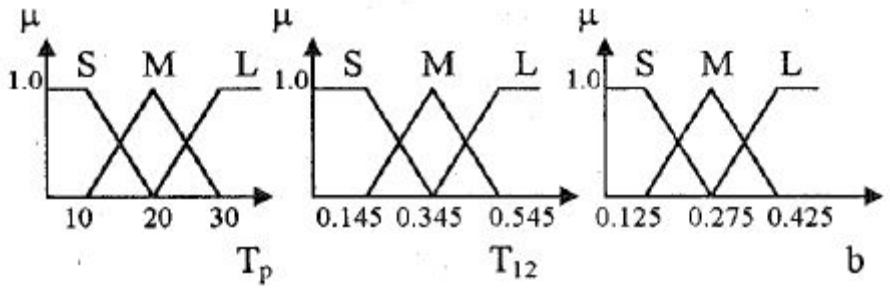

Figure 5.5: Membership functions of the monitored parameters

Patterns are represented by Sugeno w e fuzzy rules such as: IF Tp is AI' and T12 is A i and b is A3' THEN optimal gain = K' (18) where T ~T, ~b: ~monitored parameters representing a particular operating condition. AI' , A i , A3': fuzzy sets of the ith rule. K': optimal gain matrix of the ith rule. The fuzzy gain scheduler output is determined as follows:

$$K_{scheduled} = \frac{\sum_{i=1}^{27} w^i K^i}{\sum_{i=1}^{27} w^i} \tag{18}$$

$$w^i = \min(\mu(A_1^i), \mu(A_2^i), \mu(A_3^i)) \tag{19}$$

where w' is the firing strength of the i'" rule calculated as the minimum of membership values associated with that particular rule. p (AI'), p (A i) and p (A3') are the M membership values of the fuzzy variables T,, T12 and b respectively Conventional Integral Control With Fuzzy Gain Scheduling The same concept of fuzzy gain scheduling of the optimal controller is applied for adaptation of the conventional integral controller gain according to changes in system operating conditions. The design of the fuzzy gain scheduler block of this controller is exactly similar to that designed for the optimal controller except that the calculated optimal gain matrices are replaced by integral gains. The calculated integral gains for the conventional integral controller corresponding to the 27 patterns of operating conditions are listed in Table 2. To demonstrate the method of calculating the integral gain using equation (19), consider the following example. Assume an off-nominal operating condition represented by the parameters Tp-16s, T12-0.2~~a nd b-O.4pdHz. If we consider rule 12, that is 'If T, is M, T12 is S and b is L THEN K=0.52' as can be seen from Table 2, then the membership values calculated from Figure *5* for this particular case are p ("T, is W)=0.6, p ("T12 is s")=0.725, and p ("b is L")=0.834. The firing strength in this case is: w12 = min (0.6, 0.725,0.834) = 0.6 If we complete the process for the 27 rules, then the scheduled integral gain using equation (19) is 0.5458.

CHAPTER 6

SIMULATION AND TESTING

6.1 Case I: Using Mamdani Controller

Below figures shows the fuzzy sets using the Mamdani controller for the Gain Scheduling.

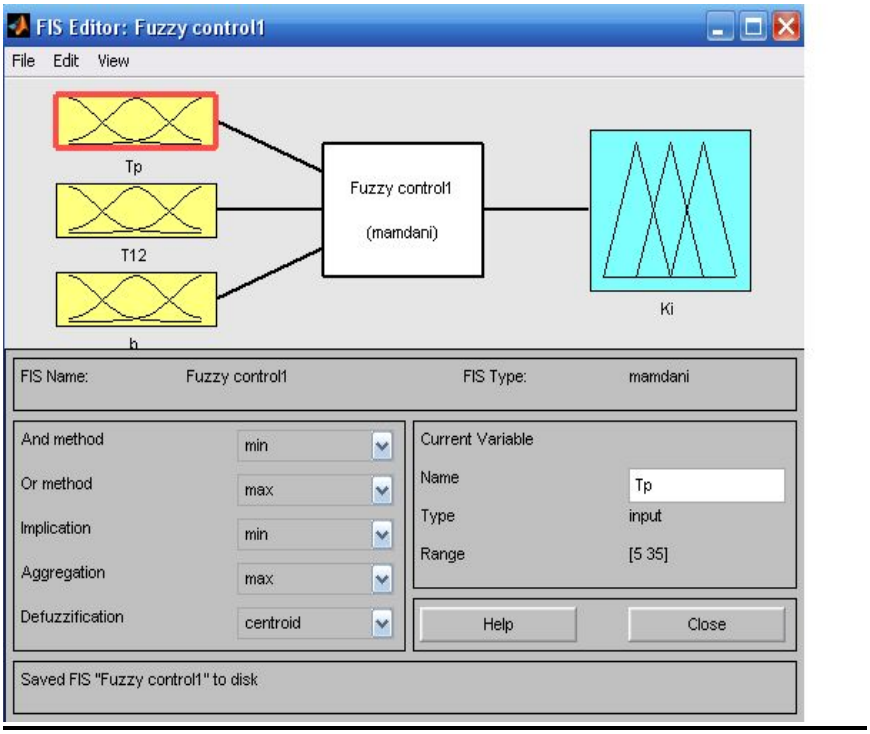

Figure 6.1: Fuzzy Mamdani Controller

The above fig shows the Fuzzy Mamdani Controller. There are three inputs T_p, T_{12}, b and the output is K_i.

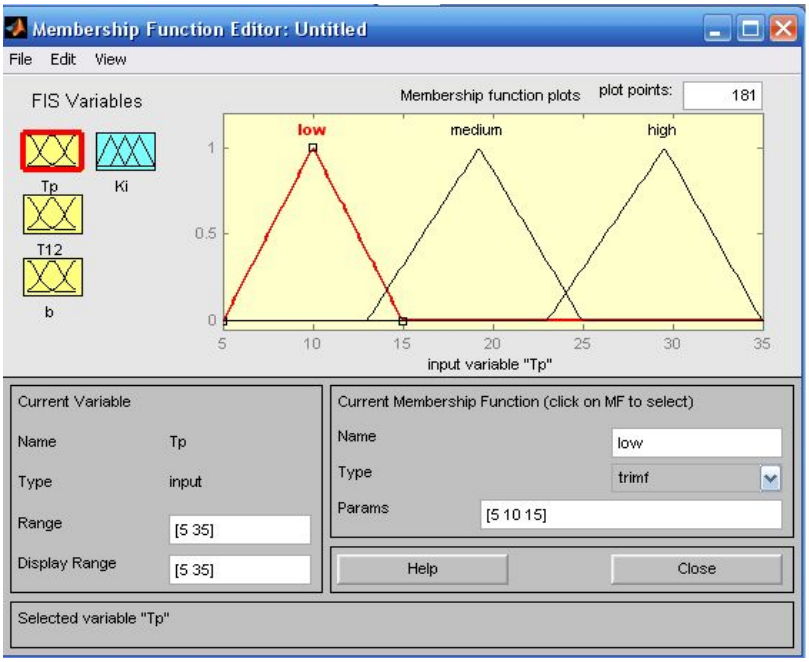

Figure 6.2: Fuzzy sets for input T_p

The above figure shows the input fuzzy sets for input T_p.

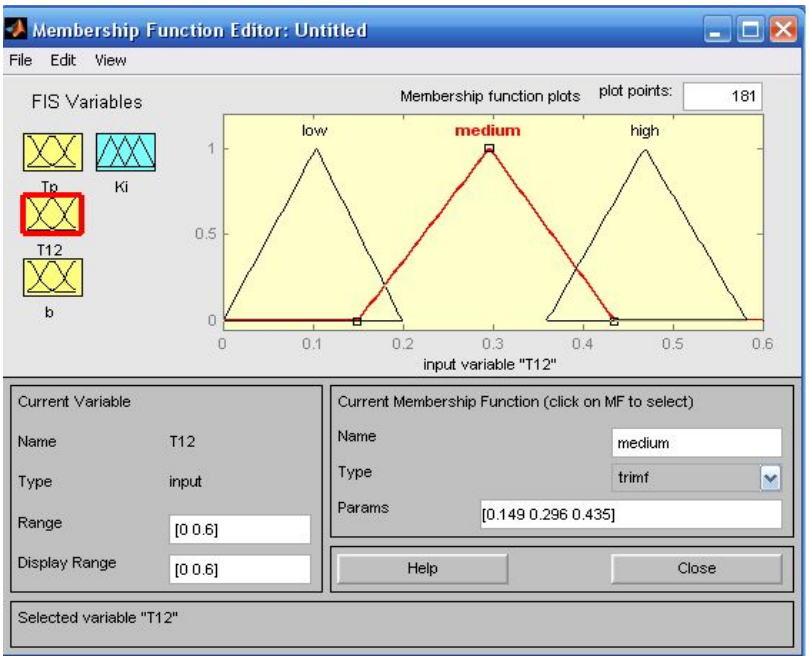

Figure 6.2: Fuzzy sets for input T_{12}

The above figure shows the input fuzzy sets for input T_{12}

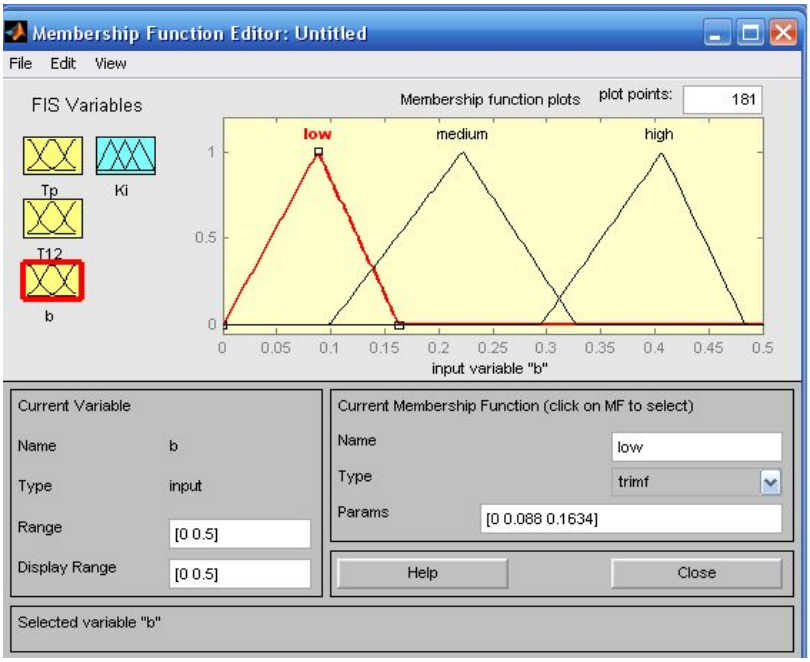

Figure 6.4: Fuzzy input sets for b

The above figure shows the input fuzzy sets for input $T_{12.}$

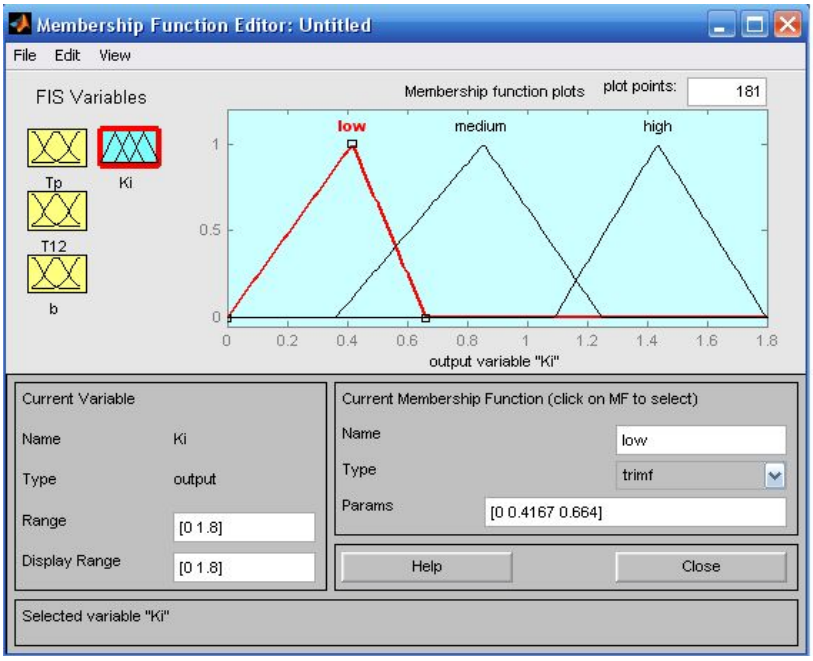

Figure 6.5: Fuzzy output sets for $K_{i.}$

The above figure shows the input fuzzy sets for input $T_{12.}$

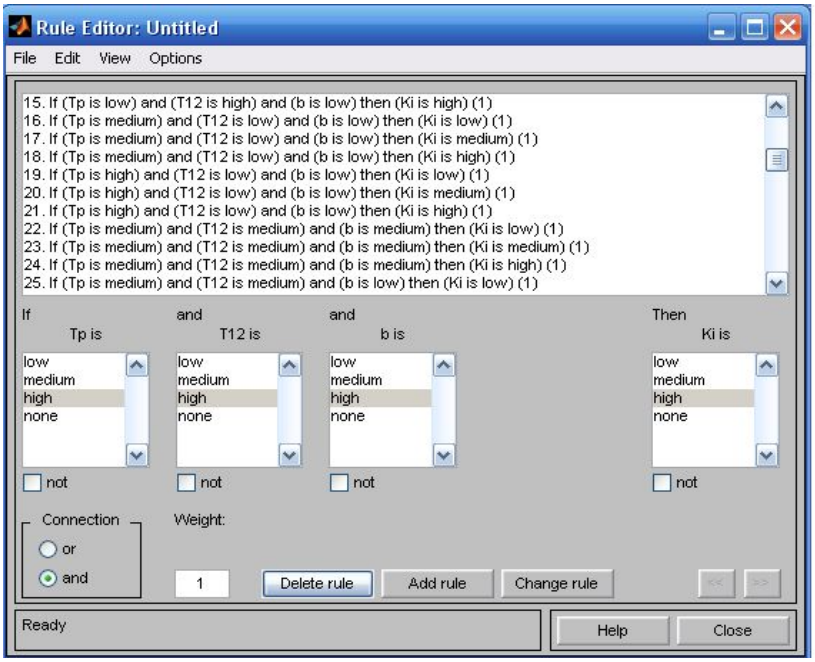

Figure 6.6: Fuzzy Rules

The above figure shows the fuzzy rules made for the inputs.

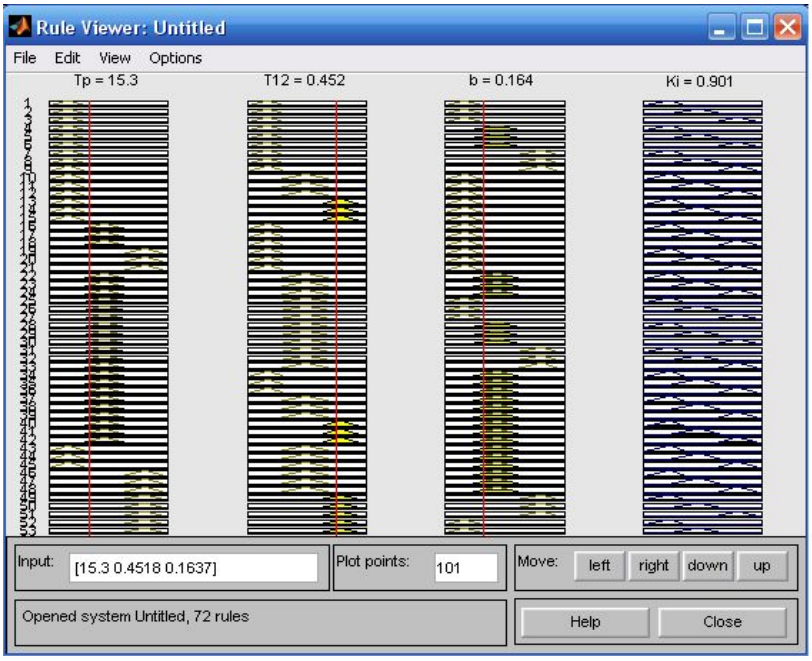

Figure 6.7: Fuzzy Rule Viewer

The above figure shows the fuzzy rules viewer to know which of the rules are fired.

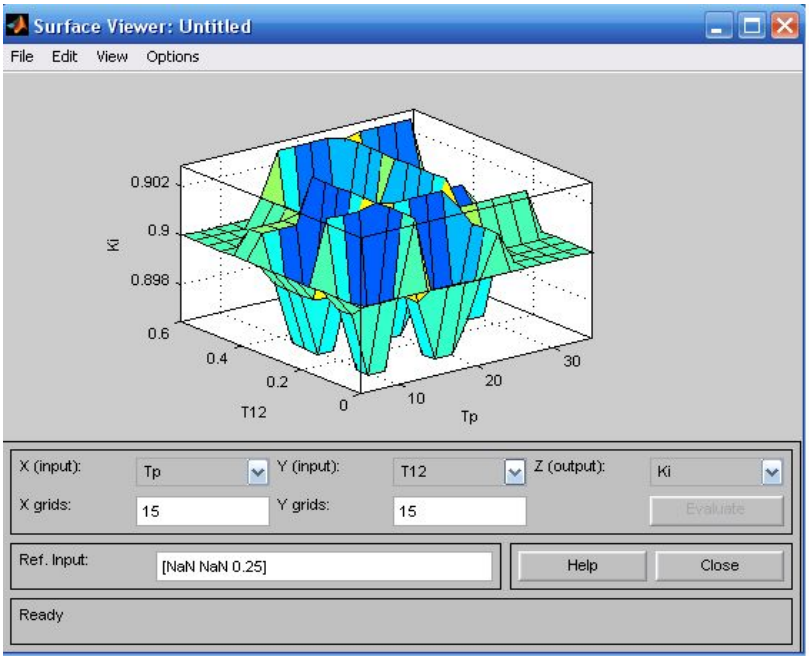

Figure 6.8: Fuzzy surface

The above figure shows the fuzzy rule surface.

6.2 Case II: Using Mamdani Controller

Below figures shows the fuzzy sets using the Sugeno controller for the Gain Scheduling.

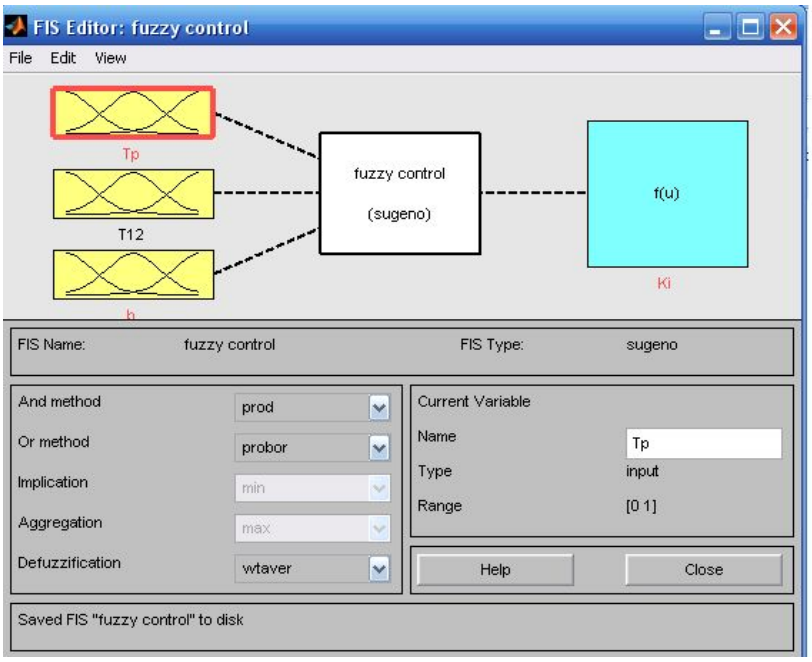

Figure 6.9: Fuzzy Sugeno Controller

The above fig shows the Fuzzy Mamdani Controller.There are three inputs T_p , T_{12}, b and the output is K_i.

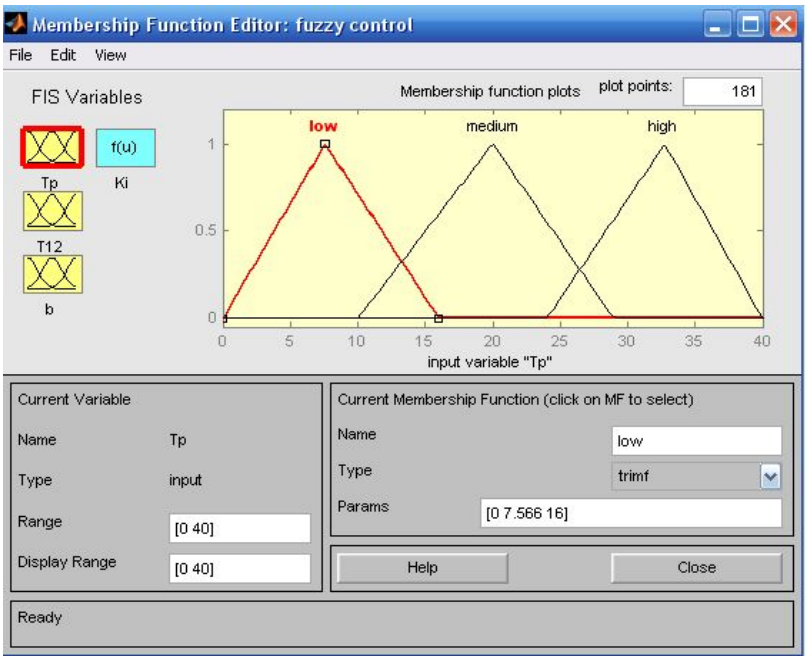

Figure 6.10: Fuzzy sets for input T_p

The above figure shows the input fuzzy sets for input T_p.

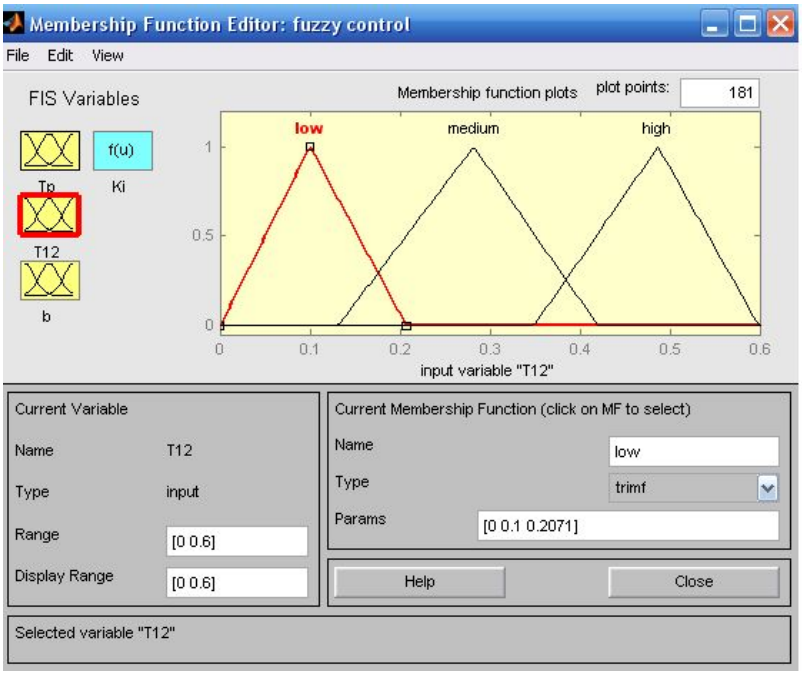

Figure 6.11: Fuzzy input sets for T_{12}

The above figure shows the input fuzzy sets for input T_{12}.

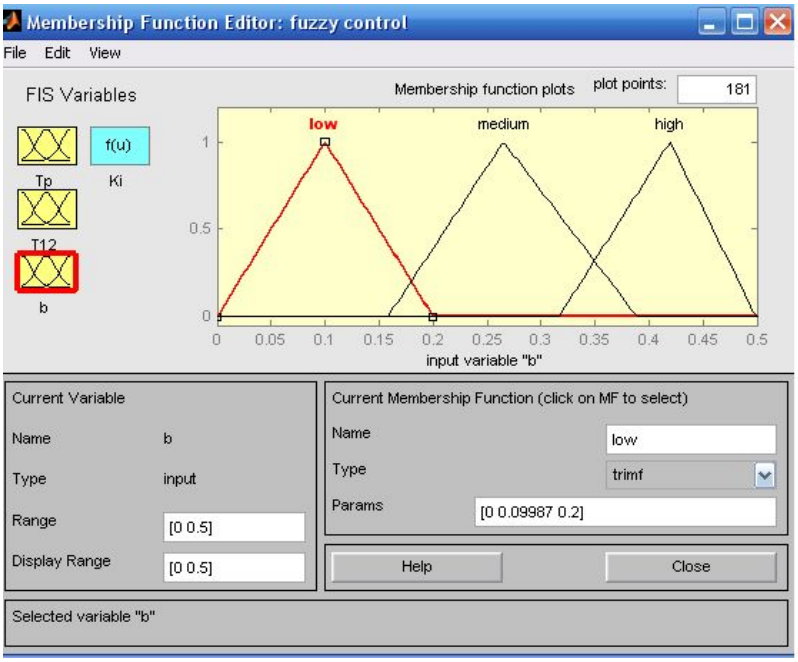

Figure 6.12: Fuzzy input sets for b

The above figure shows the input fuzzy sets for input $T_{12.}$

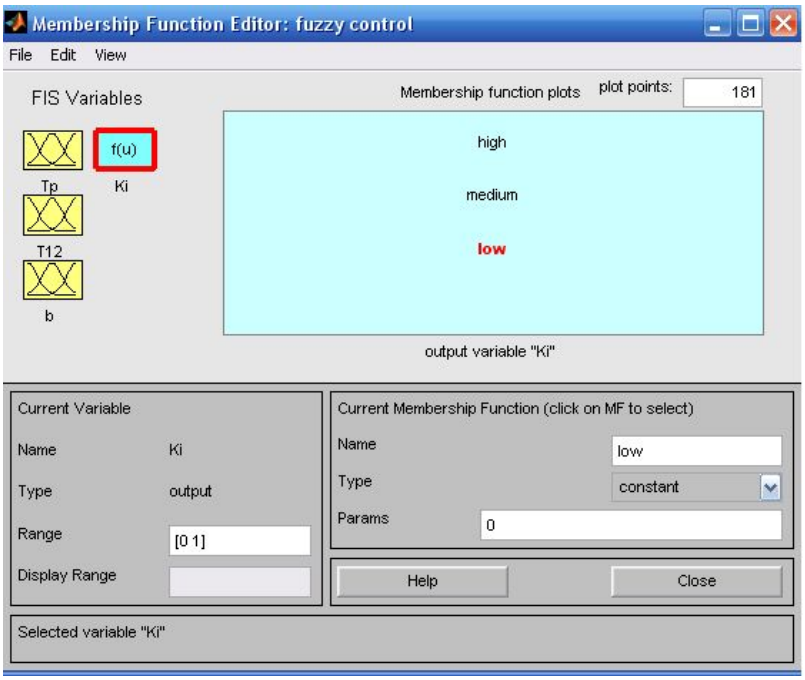

Figure 6.13: Fuzzy output sets for $K_{i.}$

The above figure shows the input fuzzy sets for input $T_{12.}$

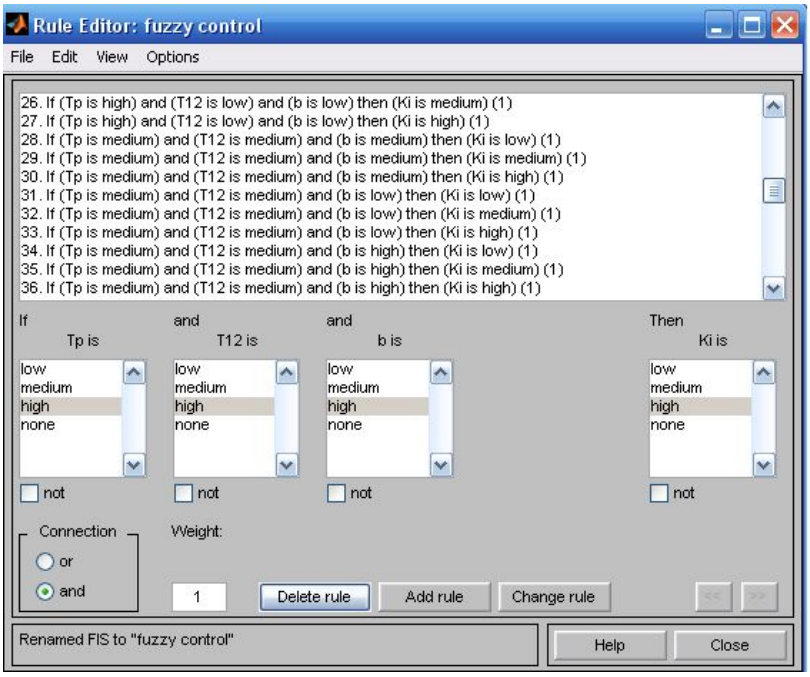

Figure 6.14: Fuzzy Rules

The above figure shows the fuzzy rules made for the inputs.

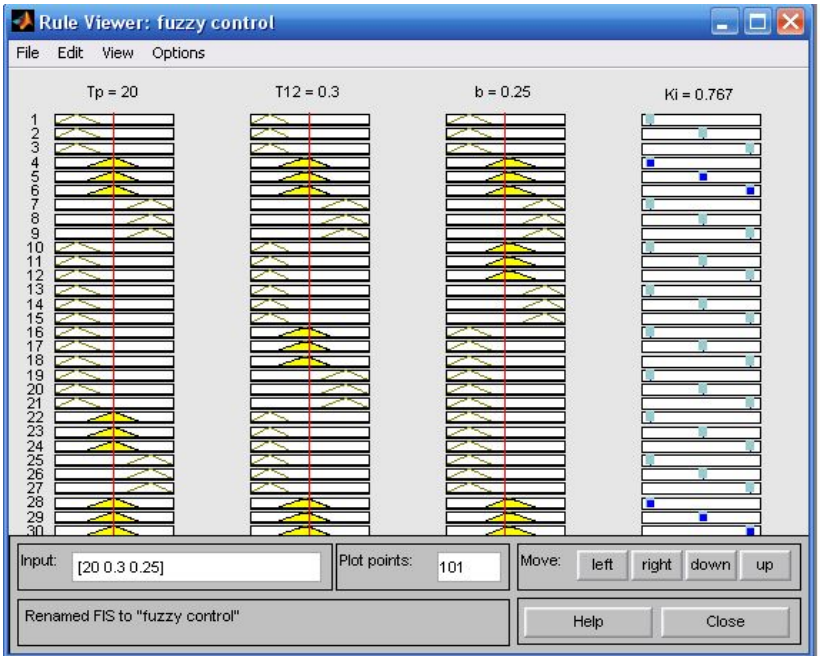

Figure 6.15: Fuzzy Rule Viewer

The above figure shows the fuzzy rules viewer to know which of the rules are fired.

CHAPTER 7

RESULT AND DISCUSSION

The following table shows the different values of gain.

Table 7.1: Different values of gain using FIS and ANFIS

S.NO.	Input(T_p)	Input(T_{12})	Input(b)	Ideal output gain (k_i)	Fuzzy Output Gain By Using Mmdani Controler	Absolute Error (e_1)	Fuzzy Output Gain Using Sugeno Contoler	Absolute Error (e_2)
1	10	0.145	0.125	1.46	1.34	.082	1.48	.013
2	10	0.145	0.275	0.65	0.75	.153	0.70	.076
3	10	0.145	0.425	0.46	0.54	.173	0.49	.065
4	10	0.345	0.125	1.41	1.51	.070	1.46	.035
5	10	0.345	0.275	0.63	0.72	.142	0.69	.095
6	10	0.345	0.425	0.41	0.53	.292	0.36	.121
7	10	0.545	0.125	1.12	1.02	.081	1.17	.147
8	10	0.545	0.275	0.51	0.42	.176	0.46	.098
9	10	0.545	0.425	0.33	0.43	.303	0.38	.151
10	20	0.145	0.125	1.33	1.43	.075	1.28	.037
11	20	0.145	0.275	0.72	0.63	.125	0.77	.069
12	20	0.145	0.425	0.52	0.48	.076	0.57	.096
13	20	0.345	0.125	1.37	1.46	.061	1.43	.043
14	20	0.345	0.275	0.78	0.70	.102	0.72	.076
15	20	0.345	0.425	0.53	0.43	.188	0.58	.151
16	20	0.545	0.125	1.29	1.21	.062	1.24	.038
17	20	0.545	0.275	0.72	0.63	.012	0.77	.069
18	20	0.545	0.425	0.49	0.58	.183	0.55	.122
19	30	0.145	0.125	1.09	1.0	.082	1.14	.042
20	30	0.145	0.275	0.66	0.57	.136	0.71	.075

21	30	0.145	0.425	0.49	0.58	.183	0.56	.142
22	30	0.345	0.125	1.05	1.15	.095	1.12	.066
23	30	0.345	0.275	0.71	0.8	.126	0.77	.084
24	30	0.345	0.425	0.52	0.62	.192	0.63	.211
25	30	0.545	0.125	1.01	1.10	.089	1.07	.059
26	30	0.545	0.275	0.68	0.58	.147	0.73	.073
27	30	0.545	0.425	.49	0.42	.072	0.44	.102

Below fig. shows the graph between ideal gain,gain obtain by using fuzzy mmdani contoler and fuzzy sugeno controler

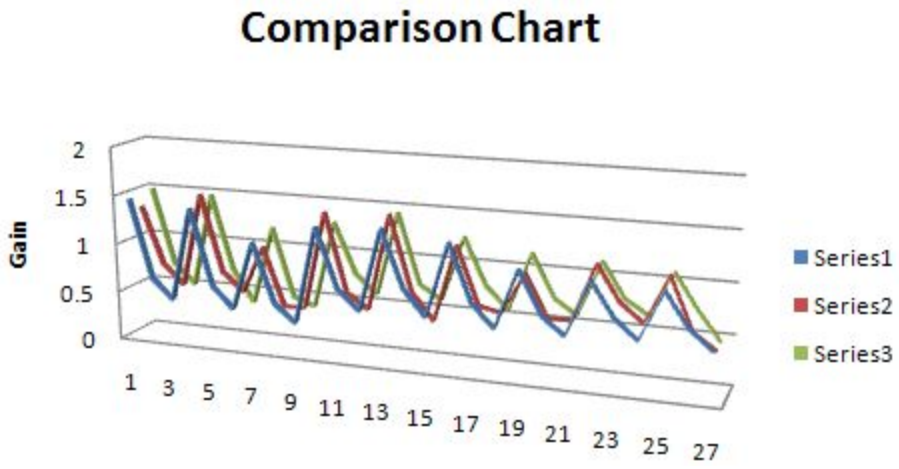

Figure 7.1: Comparison chart

In the above figure:

Series 1: Ideal Gain K_i

Series 2: Gain obtained using Fuzzy Mamdani controller

Series 3: Gain obtained using Fuzzy Sugeno controller

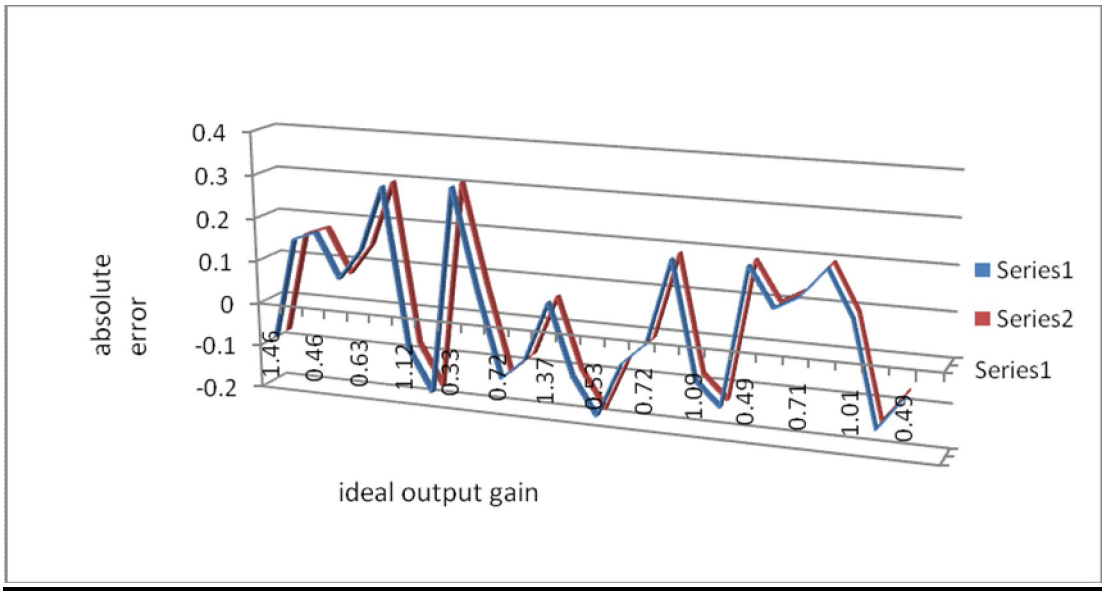

Figure 7.2: Comparison chart

In the above figure:

Series 1: Absolute error e_1

Series 2: Absolute error e_2

CONCLUSION AND FUTURE SCOPE

Fuzzy provides a robust inference mechanism with no learning and adaptability and artificial neural network provides learning and adaptability. Artificial neural networks and fuzzy systems have been successfully applied to the LFC problem with rather promising results. The salient feature of these techniques is that they provide a model-free description of control systems and do not require model identification. In this thesis, an adaptive fuzzy gain scheduling scheme for conventional PI and optimal controllers has been simulated and tested for off-nominal operating conditions. From the simulation and the result obtained in this thesis it has been shown that the proposed adaptive fuzzy logic controller offers better performance than fixed gain controllers.

An adaptive fuzzy gain scheduling scheme for conventional PI and optimal load frequency controllers has been proposed, The controllers have been simulated on a two area interconnected system. Comparison between conventional controllers and the proposed adaptive fuzzy controllers, using fuzzy gain scheduling, reveals the effectiveness of fuzzy gain scheduling used for off nominal operating conditions.

In the future we intend to apply the particular method to the other type of power-controller design. Fuzzy logic and genetic algorithms can be combined to get the optimal gain scheduling. The combined use of fuzzy logic and genetic algorithms enables a scheduling methodology to develop with broad applicability to industry.

REFERENCES

[1] Momoh, J.A. Ma, X.W. Tomsovic, K., "Overview and literature survey of fuzzy set theory in power systems" IEEE Transaction On Power Systems, Aug 1995 Volume: 10, Issue: 3 pp. 1676-1690

[2] D. K. Ranaweera, N. F. Hubele and G. G. Karady, "Fuzzy logic for short term load forecasting" International Journal of Electrical Power & Energy Systems Volume 18, Issue 4, May 1996, pp. 215-222

[3] Liu, K. Subbarayan, S. Shoults, R.R. Manry, M.T. Kwan, C. Lewis, F.I. Naccarino, J. , " Comparison of very short-term load forecasting techniques" IEEE Transactions on Power Systems , May 1996 Volume: 11, Issue: 2 pp. 877-882

[4] Christie, R.D.; Bose, A.; Power Systems, IEEE Transactions on Volume 11, Issue 3, Aug. 1996 Page(s):1191 - 1200

[5] Djukanovic, M.B Vlaisavljevic, D.;.; Sobajic, D.J.; Babic, B.S., "Fuzzy linear programming based optimal power system rescheduling including preventive redispatch" IEEE Transactions on Power Systems, pp. 525-531, Volume: 14, Issue: 2, May 1999

[6] Hiyama, T. Tomsovic, K. , " Current status of fuzzy system applications in power systems" IEEE International Conference on Systems, Man, and Cybernetics, 1999 Volume: 6, pp. 527-532 vol.6

[7] Talaq, J.; Al-Basri, F.; Power Systems, IEEE Transactions on Volume 14, Issue 1, Feb. 1999 Page(s):145 – 150

[8] Francisco Jurado, Manual Castro, Jose Corpio & Rivilla "Experience with Neural Network & Fuzzy Logic in an Electrical Engg. Control course", October 10-13,2001

[9] Y.S. Brar, Jaspreet S. Dhillon, D.P. Kothari "Multiobjective load dispatch by fuzzy logic based searching weightage pattern" Electric Power Systems Research (2002) pp. 149-160

[10] Amalia Sergakia , Kostas Kalaitzakis, " A fuzzy knowledge based method for maintenance planning in a power system" Journal on Reliability Engineering & System Safety Volume 77, Issue 1, July 2002, pages 19-30

[11] Didier Duboi, Helene Fargier and Philippe Fortemps, "Fuzzy scheduling: Modelling flexible constraints vs. coping with incomplete knowledge" European Journal of Operational Research Volume 147, Issue 2, 1 June 2003, pages 231-252

[12] Rerkpreedapong, D.; Hasanovic, A.; Feliachi, A.; Power Systems, IEEE Transactions on Volume 18, Issue 2, May 2003 Page(s):855 – 861

[13] Bansal, R.C., "Bibliography on the fuzzy set theory applications in power systems (1994-2001)" IEEE Transactions on Power Systems, Nov. 2003 Volume: 18, Issue: 4 pages 1291- 1299

[14] "M.Masiala, M.GHribi & A.Kaddouri," An adaptive Fuzzy Controllers Gain Scheduling for power systems Load Frequency control, 2004 IEEE international Conference on industrial Technology"

[15] Juang, C.-F.; Lu, C.-F.; Generation, Transmission and Distribution, IEE Proceedings- Volume 153, Issue 2, 16 March 2006 Page(s):196 – 204

[16] Sedghisigarchi, K.; Feliachi, A.; Energy Conversion, IEEE Transaction on Volume 1, Issue 1, March 2006 Page(s):250 - 256

[17] Mathur, H.D.; Ghosh, S.;Efficient Knowledge Based Gain System, Power India Conference, 2006 IEEE 10-12 April 2006 Page(s):245-250

[18] A.B. Beevi and R. Iyer, "A New Approach For The Solution Of Economic Load Dispatch Using Fuzzy Logic Controller" Asian Power and Energy Systems - 2007 pp. 560-567

[19] Chennakesava R. Alavia "Fuzzy logic & Neural Network" New Age International Publisher -2007 pp. 5-11

[20] S.RC Jang, T.Sun, E.Mizutani, "Neuro Fuzzy & soft computing", Pear Son Education-2007 pp. 6-9

[21] Sabahi, K.; Nekoui, M.A.; Teshnehlab, M.; Aliyari, M.; Mansouri, M.; Control & Automation, 2007. MED '07. Mediterranean Conference on 27-29 June 2007 Page(s):1 – 5

[22] Lianfang Kong; Lei Xiao; Control and Automation, 2007. ICCA 2007. IEEE International Conference on May 30 2007-June 1 2007 Page(s):2514 – 2518