APPLICATION OF NEURAL NETWROK ANALYSIS TO SOLUTION OF A STEP RESPONSE OF A PROPORTIONAL INTEGERAL DERIVATIVE (PID) CONTROLLER FOR A STIRRED TANK REACTOR.

BY

DAVID IBITAYO LANLEGE
M.TECH/SSSE/2004/1332

A THESIS SUBMITTED TO THE DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE, IN PARTIAL FUFILMENT OF THE REQUIREMENTS FOR THE AWARD OF DEGREE OF MASTERS OF TECHNOLOGY (M.TECH) IN MATHEMATICS OF FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA, NIGERIA.

NOVEMBER, 2007

CERTIFICATION

This thesis titled: APPLICATION OF NEURAL NETWORK ANALYSIS TO SOLUTION OF A STEP RESPONSE OF A PROPORTIONAL INTEGRAL DERIVATIVE (PID) CONTROLLER FOR A STIRRED TANK REACTOR by DAVID IBITAYO LANLEGE (M.TECH/MSSE/2004/1332) meets the regulations governing the award of the degree of Master of Technology (M.Tech) of the Federal University of Technology, Minna and is approved for its contribution to scientific knowledge and literary presentation.

.....
Prof. K. R. Adeboye
Supervisor

15/5/2008
.....
Date

.....
Dr. N. I. Akinwande
Head of Department

15-5-08
.....
Date

.....
Prof. M. Galadima
Dean, School of Science
and Science Education

16/5/08
.....
Date

.....
Prof. S.L. Lamai
Dean, Post Graduate School

5/8/08
.....
Date

DEDICATION

This project work is dedicated to the Almighty God who strengthens me through thick and thin of this program

ACKNOWLEDGEMENT

Special; thanks to my supervisor and mentor, Prof. K.R. Adeboye whose unswerving dedication, contagious enthusiasm, constant encouragements and motivation have served as propelling forces that have helped me to continue to climb the academics ladder to this very point. I wish to express my sincere gratitude for his scholarly guidance and invaluable assistance and contributions through out the course of this work, even when I was on transfer to Enugu from Minna.

I also like to thank Dr. N.I Akinwande for his fatherly advice, also for his contribution towards building me up to this level in my academic pursued.

My profound gratitude also goes to Dr. V.O Waziri for his unrelenting assistance towards the accomplishment of this project work. I thank him for his contributive criticism, thoughtful and valuable reviews and suggestions which help in improving the formulation of the control problems considered in the research work

My thanks go to the entire academic staff of the Mathematics/Computer Science Department of the Federal University of Technology Minna who have in one way or the others contributed to the success of this program. I wish to particularly thank Prof. K.R. Adeboye, Dr N.I Akinwande, Dr. Y.N Ayesinmi, Dr U.Y Abuabakar, Dr V.O Waziri, Mr. Danladi Hakimi, Mr. Abraham Ochoche, Mr. V. Akinola, Mr. JIya Mohammed and Dr. L.N. Ezeako.

I thank all my colleagues such as Mr. Lakan, Mr. Dare Jayeola, Mr. Korede, Mr. Oborh, Mrs. Nurah, Pastor Cole and Mr. Ogunbiyi for their valuable contributions in solving different problems of tutorials and working together during this program.

Worthy of mentioning also are Pastor and Mrs. Adeyeye the State overseer of Deeper Life Bible Church, Niger State, for their constant prayers, encouragement and love.

Finally, I must thank my parents Mr. and Mrs. Lanlege for there support and my friends TelImu Abubakar, Pastor Tunji Abolusoro and Rufia Ogi for their encouragement

TABLE OF CONTENTS

ABSTRACT

In this thesis, we propose a proportional integral derivative (PID) Neural Network Algorithm, which is used to model and solve continuous stirred tank mixer (CSTM) problem. This hybrid algorithm, which is new is robust and converges fast without being trapped into a local minimal as is the case with the popular neural network we establish the characteristics equation governing the dynamics of the continuous stirred tank mixer/ reactor. A controller was formulated and was tested and found to be consistent. The proportional integral derivative (PID) network was used to simulate typical continuous stirred tank mixer reactor (CSTMR) problems, of which predictive accuracy was found to be 96%. Also examine the computational richness of the human brain, which is exhibited in the highly complex interconnection of neurons with the brain. Simplified neurobiological studies of concepts learning by the human brain were modeled mathematically.

CHAPTER ONE

INTRODUCTION

1.1 GENERAL BACKGROUND TO NEURAL NETWORK

Neural network simulations appear to be a recent development. However, this field was established before the advent of computers, and has survived at least one major setback and several eras.

Many important advances in technology have been boosted by the use of inexpensive computer emulations. Following an initial period of enthusiasm, the field survived a period of frustration and disrepute. During this period when funding and professional support were minimal, important advances were made by relatively few researchers. These pioneers were able to develop convincing technology which surpassed the limitations identified by Minsky and Papert (Ref.). Minsky and Papert, published a book (in 1969) in which they summed up a general feeling of frustration (against neural networks) among researchers. Currently, the neural network field enjoys a resurgence of interest and a corresponding increase in finding.

The history of neural networks that was described above can be divided into several periods.

FIRST ATTEMPTS:- There were some initial simulations using formal logic. McCulloch and Pitts (1943) developed models of neural networks based on their understanding of neurology. These models made several assumptions about how neurons work. Their thresholds were based on simple neurons which were considered to be binary devices with fixed thresholds. The results of their model were simple logic functions such as "a" or "b" and "a and b". Another attempt was by using computer simulations. Two groups (Farley and dark, 1954; Rochester Holland, Haibit and Duda, (1956). The first group (IBM researchers) maintained closed contact with neuroscientist at MC Gill University. So, whenever their models did not work, they consulted the neuroscientists. This interaction established a multidisciplinary trend which continues to the present day.

PROMISING AND EMERGING TECHNOLOGY

Not only was neuroscience influential in the development of neural networks, but psychologists and engineers also contributed to the progress of neural network simulations. Rosenblatt (1958) stirred considerable interest and activity in the field when he designed and developed the perceptron. The perceptron had three layers with the middle layer known as the association layer. This system could learn to connect or associate a given input to a random output unit.

Another system was the ADALINE (Adaptive Linear Element) which was developed in 1960 by Widrow and Hoff (of Stanford University). The ADALINE was an analogue electronic device made from simple components. The method used for learning was different to that of the perceptron, it employed the least-mean-squares (LMS) learning rule.

PERIOD OF FRUSTRATION AND DISREPUTE

In 1969 Minsky and Papert wrote a book in which they generalized the limitations of single layer perceptrons to multilayer systems. In the book they said "our intuitive judgment that the extension (to multilayer systems) is sterile". The significant result of their book was to eliminate funding for research with neural network simulations. The conclusions supported the disenchantment of researchers in the field. As a result, considerable prejudice against this field was activated.

INNOVATION

Although public interest and available funding were minimal, several researchers continue working to develop neuromorphically based computational methods for problems such as pattern recognition.

During this period several paradigms were generated which modern work continues to enhance Grossberg's (Steve Grossberg and Gail Carpenter in 1988) influence founded a school of thought which explores resonating algorithms. They developed the ART (Adaptive Resonance theory) networks based on biologically plausible models. Anderson and Kohonen developed associative techniques independent of each other. Klopf (A. Henry Klopf) in 1972, developed a basis for learning in artificial neurons based on a biological principle for neuronal learning called heterostasis.

Werbos (Paul Werbos (1974) develop and used the back – propagation learning method, however several years passed before this approach was popularized. Back propagation nets are probably the most well known and widely applied of the neural networks today. In essence, the back – propagation net is a perceptron with multiple layers, a different threshold function in the artificial neuron, and a more robust and capable learning rule.

Amari (A shun – ichi 1967) was involved with theoretical developments: he published a paper which established a mathematical theory for a learning basis (error-correction method) dealing with adaptive pattern classification. While Fukushima (F-Kunihike) developed a stepwise trained multilayered neural network for interpretation of handwritten characters. The original network was published in 1975 and was called the cognitron.

RE-EMERGENCE

Progress during the late 1970s and early 1980s was important to the re-emergence on interest in the neural network field. Several factors influenced this movement. For example comprehensive books and conferences provided a forum for people in diverse fields with specialized technical language, and the response to conferences and publication was quite positive.

Academic programs appeared and courses were introduced at most major universities in the US and Europe. Attention is now focused on funding levels throughout Europe, Japan and the US and as this funding becomes available, several new commercial with applications in industry and financial institutions are emerging.

TODAY:- Significant progress has been made in the field of neural networks – enough to attract a great deal of attention and fund for further research. Advancement beyond current commercial applications appears to be possible, and research is advancing the field on many fronts. Neurally based chips are emerging and applications to complex problems developing. Clearly, today is a period of transition for neural network technology.

1.2 BACKGROUND TO THE STUDY

What is a Neural Network?

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the biological nervous systems, such as the brain, process

information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (Neurones) working in unison to solve specific problems.

Artificial Neural Networks, like people learn by example, (ANN) is configured for a specific application such as pattern recognition or data classification through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques.

A trained neural network can be thought of as an "Expert" in the category of information it has been given to analyze. This expert can be used to provide projections given new situation of interest. The importance of Neural networks are:

1. Adaptive Learning:- An ability to learn how to do tasks based on the data given for training or initial experience.
2. Self organization:- An Artificial Neural Network can create its own organization or representation of the information. It receives during learning time.
3. Real time operation:- Artificial Neural Network computation may be carried out in parallel and special hardware devices are being designed and manufactured which take advantage of this capacity.
4. Fault tolerance via redundant information coding partial destruction of a network leads to the corresponding degraduation of performance. However, some network capabilities may be retained even with major network damage.

The application of neural network gives this description of neural networks and how they work, what real world applications are they suited for?

Neural networks have broad applicability to real world business problem. In fact, they have already been successfully applied in many industries. Since neural networks are best for identifying patterns or trends in data, they are well suited for prediction or forecasting needs including:

1. Sales forecasting.
2. Industrial processing control.

3. Customer research.
4. Data validation.
5. Risk management.
6. Target marketing.

But to give some more specific examples, artificial neural network are also used in the following specific paradigms:

1. Recognition of speaker in communication
2. Diagnosis of hepatitis
3. Recovery of telecommunication from faulty software
4. Interpretations of nuitinceaning Chinese word
5. Under sea mining detective
6. Texture analysis
7. Three dimensional object recognition hand-written word recognition and facial recognition.

Also, an artificial neural network is a massively distributed processor or simply computing system made up of a number of simple highly inter-connected signal or information processing units (called artificial neuron) for storing experimental knowledge and making it available for use. (Haykins, (1994). It resembles the human brain in that.

(i) It can acquire and store knowledge through a learning process.

(ii) It can make available the knowledge stored when required.

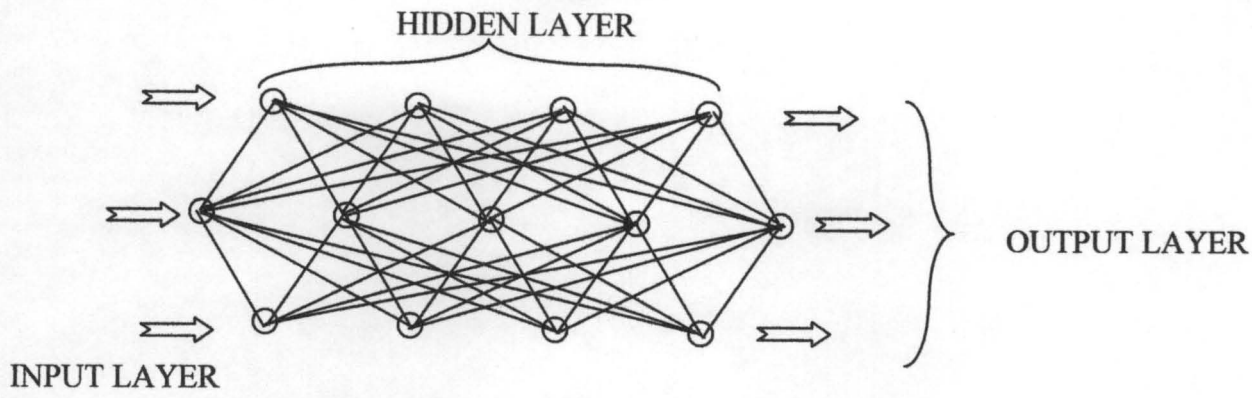


Fig. 1.1(a)

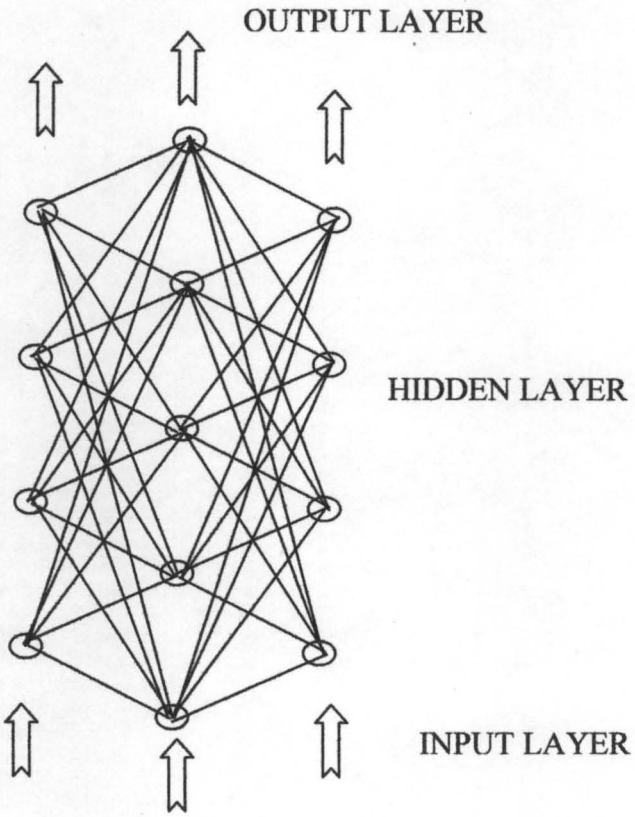


Fig. 1.1 (b)

FIG. 1.1 SCHEMATIC ARCHITECTURE OF ARTIFICIAL NEURAL NETWORK

Specific paradigms:

1. Recognition of speaker in communication
2. Diagnosis of hepatitis
3. Recovery of telecommunication from faulty software

4. Interpretations of nuitinceaning Chiness word
5. Undersea mine detective
6. Texture analysis
7. Three dimensional object recognition hand-written word recognition and facial recognition.

1.3 THE PID CONTROLLER

PID controller meaning Proportional-Integral-Derivative controller (PID controller) is a common feedback loop component in industries control system.

The PID loop tries to automate what an intelligent operator with a gauge and a control knob would do. The operator would read a gauge showing the input measurement of a process, and use the knob to adjust the output of the process ("the action") until the process's input measurement stabilizes at desired valve on the gauge. In order control literature this adjustment process is called a "rest" action.

The position of the needle on the gauge is a "measurement", "process value" or "process variable". The desired valve on the gauge is called a "Set Point". The difference between the gauge's needle and the Set Point is error. A control loop consists of three parts

1. Measurement by a sensor connected to the process
2. Decision in a controller element
3. Action through an output device (Actuator) such as a control valve.

The PID loop adds positive correction, removing error from the process's controllable variable (it input). Differing terms are used in the process control industry. The "Process Variable" is also called the Process Input or "Controller Output".

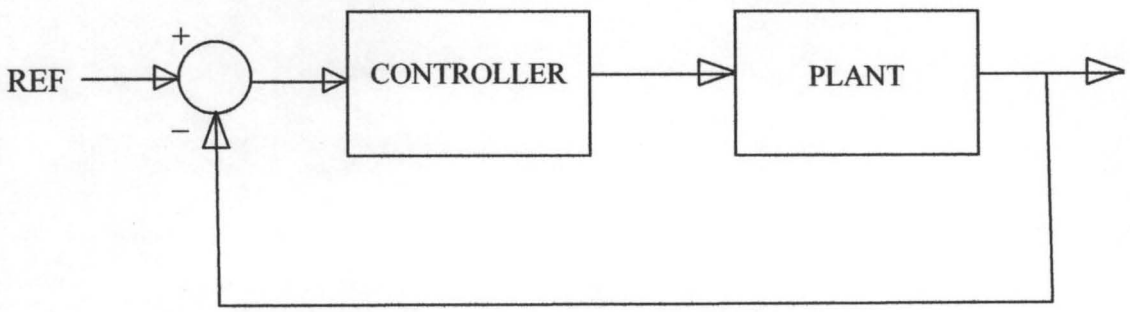


FIG. 1.2 OUTPUT FEEDBACK CONTROL SYSTEM

The process's output is called the "measurement" or "input" controller. This, up-a-bit down-a-bit movement of the Process's Input Variable is how the PID loop automatically finds the correct level of input for the process.

Removing the error "turn the control knob" adjusting the process's input to keep the process's measured output at the setpoint.

PID is named after its three correcting calculations, which all add to and adjust the controlled quality, these additions are actually subtractions of error because the proportional are usually negative.

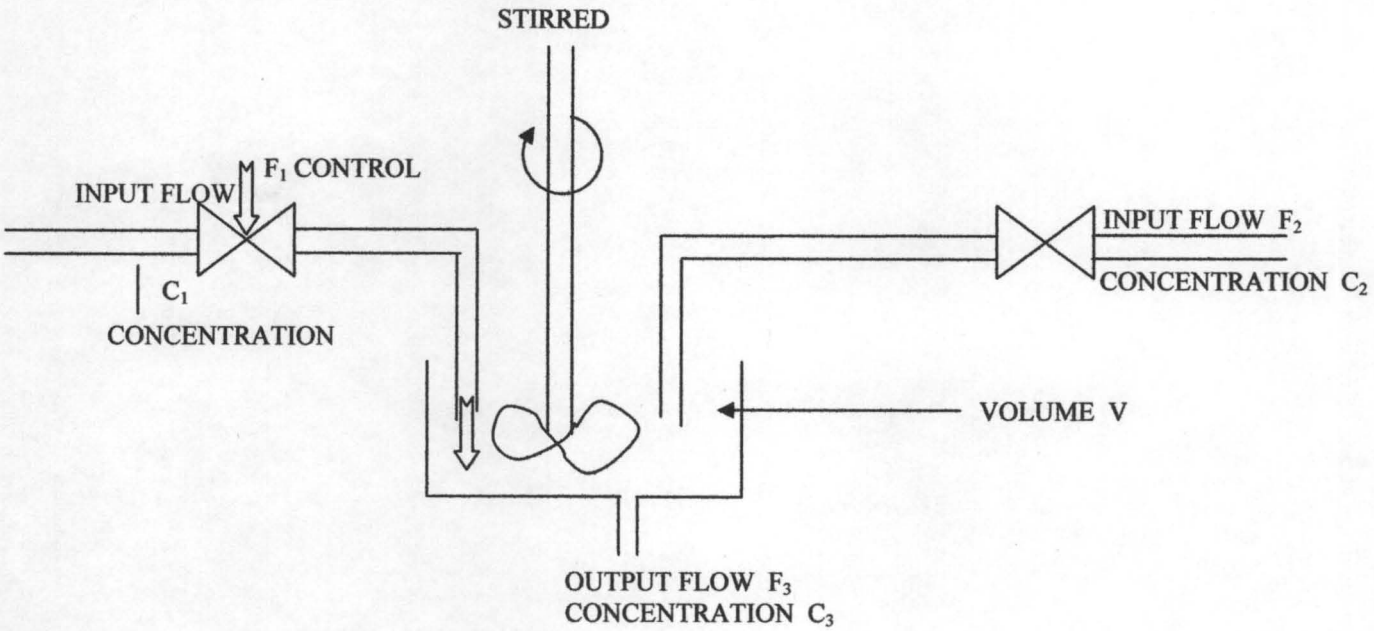


FIG. 1.3 STIRRED TANK REACTOR

Variation in mixture volume in the tank. The input flow rates F_1 and F_2 are in Fig. 1.3 controlled by input valves while the output flow is assumed to be through a constant size orifice. This output flow rate is assumed to vary as the square root of the static pressure head at the orifice with constant cross-sectional area of the tank this means.

$$F_3 = K\sqrt{V}$$

Where V is the volume of the material in the tank and K is an experimentally determined constant following K and S

$$K = 0.002m^{3/2} \text{ sec.}$$

is used here. The problem here is to control input flows F_1 and F_2 so that the output product concentration C_3 is as near as possible to the desired nominal valve C_3. At the

same time the volume of the mixture in the tank (V) is to be maintained at or near a desired nominal value so as to insure both a sufficient hold time in the tank to achieve the desired degree of mixing and also a sufficient near constant output flow rate F_3.

1.4 DYNAMICS OF THE TANK

With notation as defined above, the dynamics of the tank may be described by the mass balance equation as

$$\frac{d}{dt}(C_3V) = C_1F_1 + C_2F_2 - C_3F_3$$

$$\frac{dV}{dt} = F_1 + F_2 - F_3$$

This equation simply means that the rate of change of product in the tank varies with time as the difference between input product rate and output product rate. Volume V in the tank change as the difference between flow in and flow out with a few obvious manipulation (6.96) may be put into the form.

$$dC_3 = \frac{(C_1 - C_3)F_1 + (C_2 - C_3)F_2}{V}$$

$$\frac{dV}{dt} = F_1 + F_2 - F_3$$

Which is the x = f(x) form desired here.

1.5 NEURAL COMPUTING TECHNIQUES:

Neural networks take a different approach to problem solving from that of conventional computers. Conventional computers use an algorithmic approach i.e. the computer follows a set of instructions in order to solve a problem. Unless the specific steps that the computer needs to follow are known the computer cannot solve problem. That restricts the problem solving capacity of conventional computers that we already understand and known how to solve. But computer would be so much more useful it they could do things that we don't exactly know how to do.

Neural Network processes information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements (neurons) working in parallel to solve a specific problem. Neural Network learns by example. They cannot be programmed to perform a specific task. The examples must be selected carefully otherwise useful time is wasted or even worse the

network might be functioning incorrectly. The disadvantage is that because the network finds out how to solve the problem by itself, its operation can be unpredictable.

In contrast to conventional computing which requires an explicit analysis of the problem to be solved to enable the programmer write down a step by step sets of instruction to be followed by a computer, Neural computing does not require an explicit description of how the problem is to be solved. The neural computer is able to adapt itself during training period, based on examples of similar problems often with a desired solution to each problem. After a sufficient training the neural computer is able to relate the problem data to the solutions input to outputs, and it is then able to offer a variable solution to a brand new problem.

1.6 THE MAJOR ELEMENTS OF NEURAL NETWORK

The neuron is the basic unit of brain, and is a stand alone analogue logical processing unit.

The basic function of a biological neuron is to add up its inputs and produce an output. If the sum of the input is greater than sum value, known as the threshold value; then the neuron will be activated and "FIRED" if not, the neuron will remain in its inactive, quiet state.

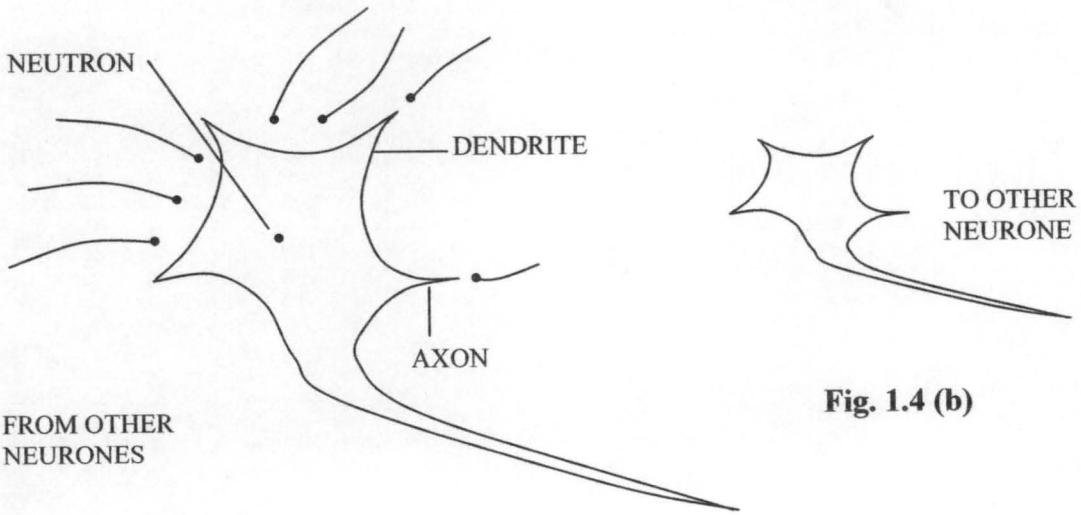


Fig. 1.4 (b)

Fig. 1.4 (a)

FIG. 1.4 THE COMPONENT OF A NEURON

The inputs of the neuron arrive along the dendrites which are connected, to the outputs from the other neurons by specialized function called synpses. These functions alter the effectiveness with which the signal is transmitted, some synapses are good function and pass a large signal across whilst other are very poor and allow very little through.

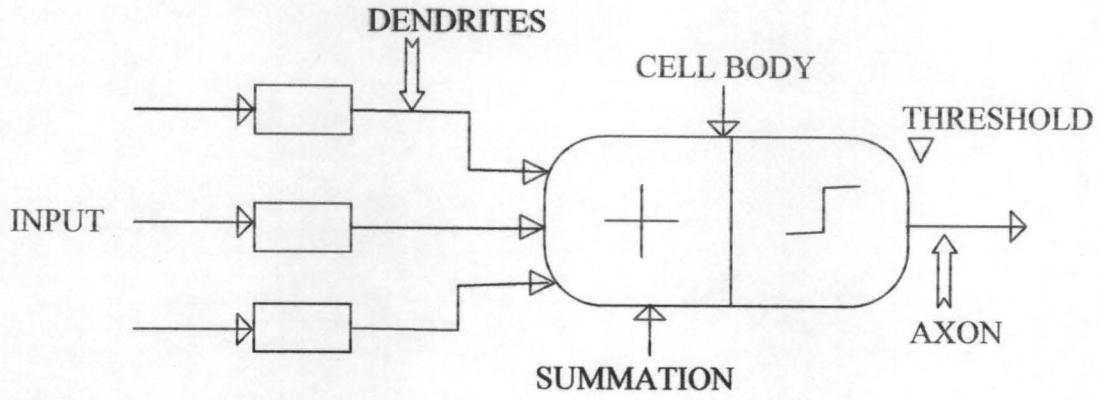


FIG. 1.5 A SIMPLE NEURON

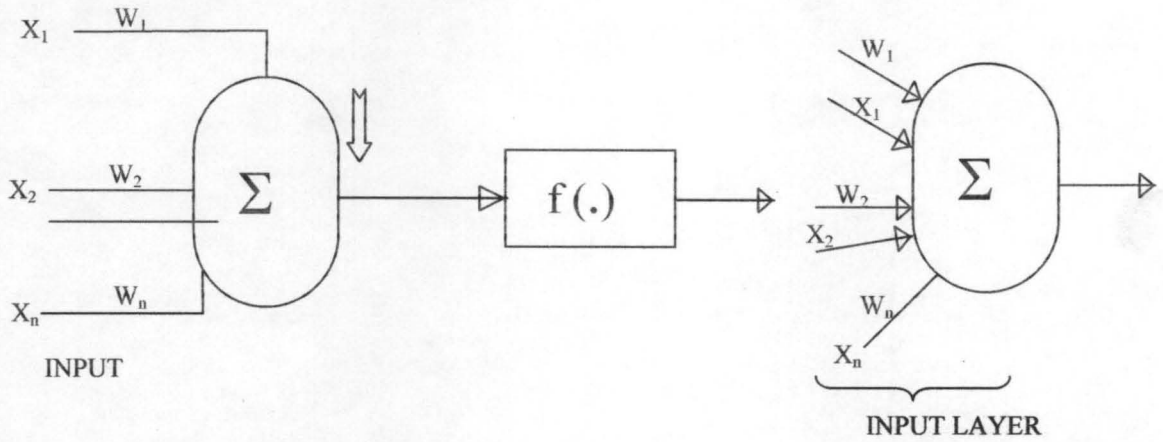


Fig. 1.6(a) **Fig. 1.6(b)**

FIG. 1.6 A SINGLE MODEL NEURON

The firing rule is an important concept in neural networks and accounts for their high flexibility. A firing rule determines how one calculates whether a neuron should fire for any pattern. It relates to all the input patterns, not only the ones on which the node was trained. Just like the biological neural network, the neuron is the major elementary processing units in artificial neural network.

In most common networks, neurons are arranged in layers with the input layer. The data fed to the network at input layer. The data then passes through the network to the output layer to provide the solution or answer.

At each neuron, every input has an associated weight which is the strength of each input connected to that neuron. The neuron simply adds together all the inputs and calculates an output to be passed on.

1.7 BENEFITS OF NEURAL COMPUTING

(i) ABILITY TO TACKLE NEW KINDS OF PROBLEMS

Neural network are effective at solving problems whose solutions are difficult, if not impossible to define, and since it has the ability to learn from experience (previous examples) when presented with a new but similar problem, it can provide solution.

(ii) FAULT TOLERANT

Since data and processing are distributed rather than centralized, neural network can be very tolerant of faults. This contrasts with conventional systems where the failure of one component usually means the failure of the entire system.

(iii) ROBUSTNESS

Neural networks tend to be more robust than their conventional counterparts. They have ability to cope well with incomplete FUZZY data.

(iv) FAST PROCESSING SPEED

Neural network are very fast because they consist of larger number of massively inter-connected processing units, all operating in parallel on the same problem. This contrast to the serial, one step at a time processing.

(v) FLEXIBILITY AND EASE OF MAINTENANCE

Neural computers are very flexible in that they are able to adapt their behaviours to new and changing environments. This contrasts to the serial conventional computing which is strictly algorithmic and requires writing a new program for any modification.

They are also easier to maintain. To accommodate changes or modifications some networks have the ability to learn from experience in order to improve their own performance.

(vi) **PARALLEL PROCESSING**

Parallel processing is a processing technique, which involves multiple operations being carried out simultaneously. Parallelism reduces computational time. For this reason, it is used for many computationally intensive applications such as predicting economic trends or generating special visual effects feature films. The high speed with which the brain and Artificial Neural Networks (ANNs) are able to process information is astounding. Consider the amount of computational needed to process a single visual image. If one restricts the image resolution to 1,000 x 1,000 receptors, a small number compared to the retina over one million (three millions for colour images) must be examined and several million computational performed in order that object in the image are identified. Even at the nanosecond speed of modern computers, this task can require several seconds in conventional computers. In contrast biological visual system computerizes such tasks in milliseconds.

(vii) **By Patterning Themselves** after the architecture of the brain, they provide a plausible model of intelligent mechanism.

(viii) They provide a tool for modeling and exploring brain function.

1.8 NEURAL NETWORK ARCHITECTURES

Artificial Neural Networks (ANN) use a variety of architectural of which the multilayer feed forwards perceptron and the recurrent neural networks are notable. Others include the radial basic function network [(Broomhead and Lowe 1988) – (Moody and Darken 1989), the adaptive and learned vector quantization network (Kangas et-al, 1990)] use for data compression, the Kohonen self-organizing maps (Kohonen, 1988, 1998 the counter-propagation neural network (Hecht-Alnielson, 1987), the adaptive resonance theory ART1 and ART2) proposed by (Carpenter and Grossbery 1987, 1988), Probabilistic neural network (Cain 1990, Specht, 1990), the

self-organizing feature maps (SOFM) Fukushima 1989 and the cellular neural networks (CNN) (Chua and Yang 1988), Chua et at (1993).

1.9 SCOPE OF THE STUDY, AIMS AND OBJECTIVE OF THE RESEARCH METHODOLOGY

AIMS

(1) To use Neural Network to obtain the output of a PID controller stirring tank reactor.

(2) To simulate a controller for a Chemical Processing problem the continuous Stirred Tank Mixer (CSTM) and continuous stirred tank reactor.

THE OBJECTIVES

(i) Formulate a control tutorial for matLab to solve problem.

(ii) Formulate a cost function for PID control process in a mixer by keeping the effluent and volume of the mixture in the tank at a desired level.

CHAPTER TWO

LITERATURE REVIEW

2.1 NEURAL NETWORK RESEARCH.

Artificial Neural Network (ANN) has been shown to be effective as computational processor for various tasks including data compression, classification; combinational optimization problem solving modeling and forecasting, adaptive control, multisensor data fusion, pattern recognition e.g. speech and visual Image recognition, noise filtering etc. ANN uses a variety of architectures, the notable ones are listed in section 1.2 in Chapter One. It is believed by many researchers in the field that neural network models offer the most promising unified approach to building truly intelligent computer system; and that the use of distributed, parallel computations as performed in ANN is the best way to overcome the combinational explosion associated with symbolic serial computations when using Von Neuman Computer Achitecture.

Hopfeild (1982): Presented a paper at the national academy of science describing how an analysis of stable point could be performed for symmetrical recurrent crossbar networks. The analysis was based on the use of a Lyapunuvs energy function for the nonlinear equation. He showed that the energy function dissipated (decreased) and converged to a minimum and remained there.

The successful implementation of the application of artificial Neutral networks (ANN) have been reported in areas such as control[(Balakrishan and Weil, 1996)], telecommunications [(cooper,1994)] BioMedical [(Alvager et al 1994) Hasnain (2001), Muhammad (2001)],remote sensing [(Goita, et al.(1994)] pattern recognition [(Smetanin, Y. G (1995)] RF/ Microwave design[(Zhang and creech,(1999)] Microstrip circuit design [(Horng et al (1993)] Microwave circuit analysis and optimization [(Zaabab et al 1994)] Application of ANNs to biologicals systems (Stem cells) [(Szu and hwang (2003) Szu (2003), Grop-Hardt and Laux (2003)].

Neural Networks based on adaptive resonance theory are equipped with unique computational abilities that are needed to function authomonuosly in a changing environment [(Carpenter and Grossberg, (1988)] [(Catpenter et al. (1992)]

[(Aldrich C, Moolman D.N and van Deventer,(1995) at the Department of Chemical Engineering at the University of Stellenbosch successfully implemented a self-organizing and adaptive neural network system in the monitoring and control of the behaviour of an Industrial/Platinum flotation plant (Hydrometallurgical process). Other network formalism; namely radial basis function (RBF) and adaptive resonance theory – 2 (ART 2) network have also been employed for fault detection diagnosis and process monitoring task [[Leonard and Kramer, (1991)], [Whiteley and Davis, (1994)]. [Zhang and Julian, (1994)] used a locally recurrent Neural Network to model the pH dynamics in a continuous stirred tank reactor (CSTR) in a problem taken from [McAvoy et al, (1972)].

Stephen Grossbery (1982), founder and director of the centre for adaptive systems and professor of mathematics, psychology and biomedical engineering at Boston University also carried out several researches on the Psychological and biological Information Procession and in the use of Artificial neural Network (ANN) to model human perceptron and recognition. His earlier work focus on co-operative competitive learning systems leading to the creation of constructs such as instar, outstar and avalanche used in learning and recall of spatial- temporal patterns. Later, Grossbery and his colleagues focused on the Mathematical dynamical properties of ANN. This work led an important theorem on the global convergence of dynamic networks [(Cohen and Grossbery 1983)]. Gorssbery is perhaps best known for the highly successful adaptive resonance theory networks (ART Networks).

The Japanese researcher Kunihiko Fukushima (1988) founder, of the cognitron and Neocognitron networks [(Fukushima and Myaki 1982) Fukushima (1988)], has the most recent networks called the Neocognition which is a hierarchical feed forward network that learns through either; supervised or unsupervised methods. The network is modeled after biological Visual neutral systems.

Fukushima K. and Myaki (1982) published result showing that Neocognitron is capable of recognizing hand written character, independent of scale, position and some deformation in the characters [(Fukushima and Wake 1991)].

One of the earliest example of ANN used in solving optimization problem is the Dynamic Recurrent Network (DRNN). The behaviour of the DRNN has been described by Pneda, (1988), (1989) and studied by several other researchers, including Almeida (1987), Williams and Peng (1990), Zisper(1989), Peaelmutter (1998) among others. ANN has been used for a number of problems that require finding an optimal

or near optimal solutions. Such problems typically require the satisfaction of some constraints. Some examples of optimization problems include, scheduling of manufacturing operation; finding the shortest of all distance of paths through a large number of cities that must be traversed sequentially with a single visit, minimizing some cost functions under a set of constraints, N-Queen Problem, graph colouring and max (graph) cut etc. These grow at exponential rate with the problem parameter size. Optimal solutions to those problems require prohibitive computational costs. Therefore, near optimal solutions are acceptable compromises.

Tank and Hopfield (1987) implemented the dynamic recurrent Neural Network in solving the traveling salesman problem. Since then, many other solutions have been proposed, including those due to Aarts and Korst (1989) and the SOFM network angeniol et al (1988). Following the results of the Application of neural network for industries (ANNIE) project in Europe documented in the project ANNIE Hand book, Croall and Manson (1991) revealed that one of the applications chosen for investigation and successfully solved is the crew scheduling problem.

The objective of the problem is to optimally satisfy a given set of tasks using available resources subject to certain constraints (priorities, deadlines, cost). The goal is to minimize a cost function, which depends on service time and cost of service. In 1986, Carnegie – Mellon University converted a commercial van into a laboratory vehicle (Navlab. 1) to act as a test bed for the autonomously driven (driveless) vehicle experiments [Thorpe et al (1991), kanade et al., (1994)]. One of the control systems, the ALVNN (i.e. Automatic Land vehicle in a Neural Network) is neural net based. ALVNN is not the only successful ANN control system for autonomous vehicle driving.

The Advanced Research Project Agency (ARPA) had also built Automatic Land Vehicles (ALV) as well as European and Japanese organizations. Staib and Staib (1993) of the Neural Application Corporation develop an intelligent arc furnace ANN controller for positioning electrode in the furnace for steel making, thus minimizing the high cost involved and improving on the operating efficiency. Toshiba also developed a microwave oven toaster that is controlled by both a neural network and fuzzy logic controller. The oven is capable of estimating the number of items being cooked and the oven temperature. The network then decides the optimum cooking time using fuzzy reasoning. The cooking time depends on the number of

items in the cooker and their initial temperature. This is sensed and estimated by the neural network, which regulates the flow of cooking gas.

Recurrent Neural Network (RNNS) has been used in a number of interesting Applications including associative Memories, Spatiotemporal pattern classification, control, optimization and forecasting. It has been shown to perform computational task equivalent to finite state automata (Elman 1991) as well as more general turning machine (Williams and Zipser 1989)

One of the most important developments of recent neural network research is the discovery of a learning algorithm to adjust the weights in multiplayer feed forward networks (also called multiplayer perceptron). The algorithm is known as back propagation. Since the weights are adjusted from the output layer backwards, layer-by-layer to reduce the output errors. The method was discovered at different times by Werbos (1974), Parker (1985) and Rumelhart, Hinton and Williams of the parallel distributed processing (PDP) Group (Rumelhart et al 1986).

The feed forward networks are commonly utilized for pattern recognition task. The feed forward network trained, using the back propagation (BP) algorithm (Rumelhart et al, 1986, Rumelhart and McClenland (1986) is found to be very effective for process faults detection. Reddy et al (1997) is found to be very effective for process fault detection Reddy et al (1997) implemented a modified back propagation neural network methodology for identifying and interpreting variation in process pattern. In the application of neural network to control and optimization problems, much success has been made and the discipline is still an active area of research.

2.2 PROCESSING CONTROL SYSTEM

Process control involves the regulation of variables in a process. A process is any combination of materials and equipment that produces a desirable result through changes in energy, physical properties, or chemical properties, Bateson (1993). Example of a process includes a petroleum refinery, a fertilizer plant, a food processing plant, an electric power plant and a home heating system. The most common controlled variables in a process are temperature, pressure, flow rate and level. Others include density, composition, PH and viscosity.

The process control system usually incorporates a device or combination of devices that automatically controls a mechanism, a source of power or energy or other

variables. The system automatically compares the controlled output of a mechanism to the controlling input. The difference between the settings or position of the output and input is called the error signal which regulates the output to a desired value.

The process of sending the error signal back for comparison with the input is called feedback and the whole process of the input, output, error signal and feedback is called a closed loop, process control systems may be either open-loop or closed loop, but the closed loop system are more common.

Process control systems are usually encountered among other fields in chemical engineering, electrical engineering and mechanical engineering. This research work focuses on chemical process control. Theoretical papers on chemical process control started to appear about 1930. Grebe et al, (1933) discussed some difficult PH control problems and showed the advantage of using controllers with derivative action. Ivanoff, (1934) introduced the concept of potential quantitative evaluation of control system. [Calendar et al, (1936) showed the effect of time delay on the stability and speed of response of control systems. The field has continued to attract researchers and many attractive research results continue to appear in publications. [Kestenbaum et al, (1976) published an article on "Design concepts for process control"], [Castellano et al, (1978)] published results of "Digital control of a Distillation System", [Brosilow and Tong, (1978) discussed "The Structure and Dynamics of Inferential Control Systems", [Stephanopoulos (1982)] researched and published results on the "optimization of closed-loop Responses". [Aldrich C., Moolman D.N. and Van Deventer, (1995) of the Department of Chemical engineering at the University of Stellenbosch successfully implemented a self-organizing and adaptive neural network system in the monitoring and control of the behavior of an industrial/platinum flotation plant (Hydrometallurgical process).

2.2.1 FEEDBACK CONTROL

Essential to all automatic control mechanism is the feedback principle, which enables a designer to endow a machine, reactor or system with the capacity for self-correction. A feedback loop is a mechanical, pneumatic, or electronic device that senses or measures a physical, quantity such as position, temperature, size or speed e.t.c., compares it with a pre-established standard, and takes whatever preprogrammed action that is necessary to maintain the measured quantity within the limits of the acceptance standard. In a feedback control loop, the controlled variable is compared

to the set point, with difference, deviation or error acted upon by the controller to move in such a way as to minimize the error. This action is specifically a negative move in such feedback, in that an increase in deviation, moves so as to reduce the deviation. (Positive feedback would cause the deviation to expand rather than diminish and therefore does not regulate.) The action of the controller is selectable to allow the use on process gains of both signs.

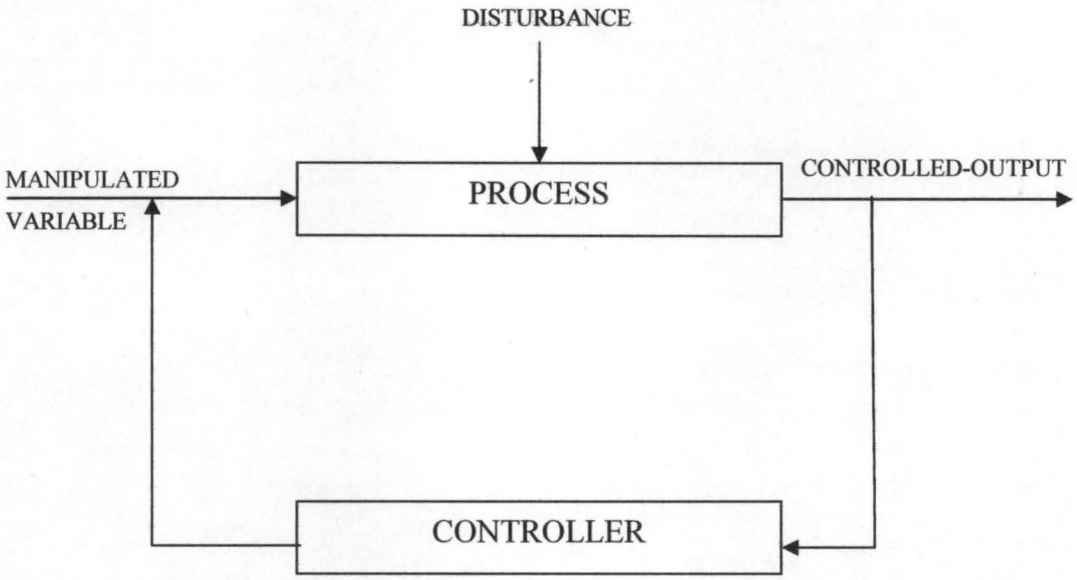


FIG. 2.1 STRUCTURE OF A FEEDBACK CONTROL SCHEME CHEMICAL PROCESS CONTROL STEPHANNOPOULOS

Basically feedback control loop consists of:

i.) Sensor: to detect the process variable.

ii.) Transmitter: the transmitter is the interface between the process and its control system. It incorporates a converter or transducer that converts the sensor signal (millivolts, mechanical movement, pressure differential e.t.c.) into an equivalent air or electrical control signal.

iii.) Controller: That compares this process signal with a set point and produces an appropriate control signal.

iv.) Control value.

There are different types of feedback controllers namely

(i) Proportional controller.

(ii) Proportional-integral controller.

(iii) Proportional-derivative controller.

(iv) Proportional-integral-derivative controller (PID)

2.2.2 FEED FORWARD CONTROL

Most feedback systems act "Post facto" (after the fact) that is after the effect of the disturbances has been felt by the process. Unlike the feedback systems, a feed forward system uses measurements of disturbance variables to position the manipulated variable in such a way as to minimize any resulting deviation.

The disturbance variable could be either measured loads or the set point. The feed forward gain must be set precisely to offset the deviation of the controlled variables from the set point.

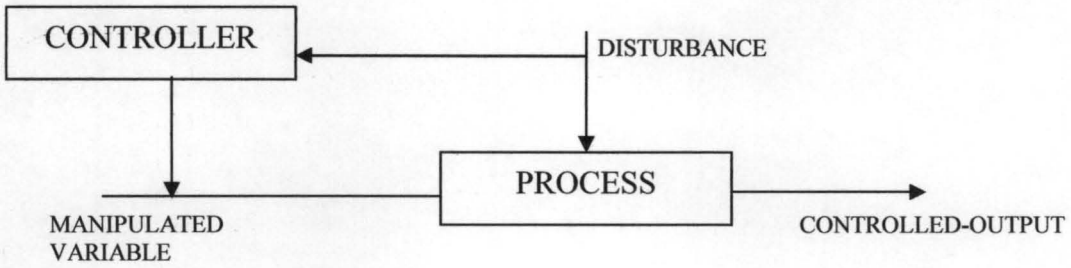


FIG. 2.2 STRUCTURE OF A FEED FORWARD CONTROLL SCHEME (CHEMICAL-PROCESS CONTROL, STEPHANNOPOULOS 1982)

Feed forward control is usually combined with feed back control to eliminate any offset resulting from inaccurate measurements and calculations and unmeasured load components.

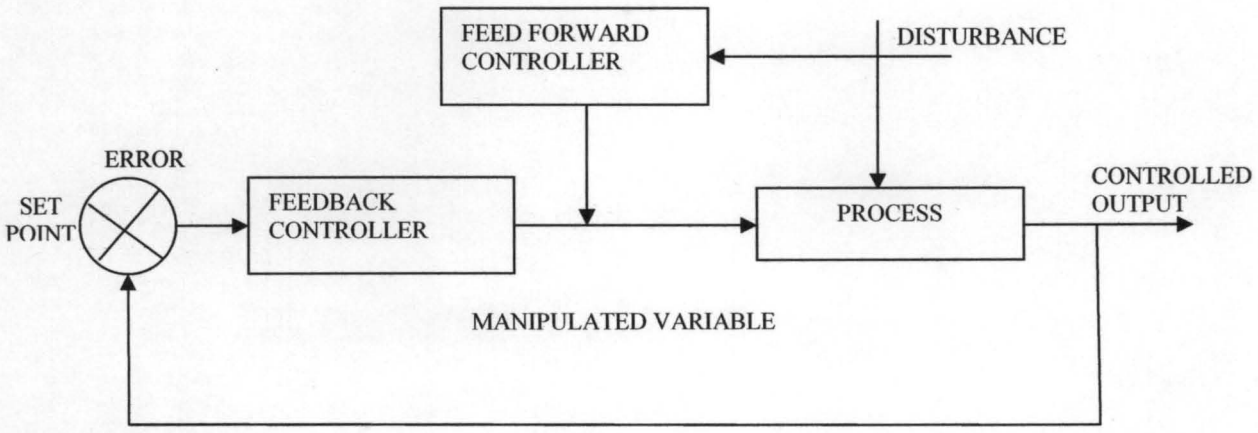


FIG: 2.3 FEEDBACK LOOP

2.2.3 COMPUTER CONTROL

Computer has been used to replace analog Proportional Integral-Derivative (PID) controllers, either by setting set point of lower level controllers in supervisory control or by driving value directly indirectly digital control.

In distributed control systems with the digital processor shared among many control loops, the separate processors are used to monitor different activities. Other types of controller include self-optimizing controllers and Neuro-Fuzzy based controller (Shink, 1988).

2.2.4 CHEMICAL PROCESS CONTROL

Most chemical process control occurs in a chemical plant. A chemical plant is an arrangement of processing units (reactors, heat exchangers, pumps, distillation columns, absorbers, evaporators, tank e.t.c.) integrated with one another in a systematic and rational manner for performance of a set task. The plant's overall objective is to convert certain raw material (input, feedstock) into desired products using available sources of energy in the most economical way.

Chemical plants must satisfy several requirements imposed by its designers and the general technical, economic and social conditions in the presence of ever-changing external influence (disturbances).

Among such requirements are the following.

(i) production specifications
(ii) operational constraints.
(iii) Safety
(iv) Environmental regulations
(v) Economics

These requirements listed above dictate the need for continuous monitoring of a chemical plant and external intervention (control) to guarantee the satisfaction of the operational objectives. In the light of the foregoing, chemical control system must satisfy the following three general classes of need.

(1) Suppressing the influence of external disturbances.
(2) Ensuring the stability of a chemical process.
(3) Optimizing the performance of a chemical process.

Chemical reactors are often the most difficult units to control in a chemical plant, particularly if the reactions are rapid and exothermic. For example an increase in temperature of 1^0C may increase the reaction rate by 10% enough to cause significant changes in conversion and perhaps in yield. And the increase in rate with increasing temperature tends to make the reactor unstable.

TYPES OF REACTOR

There are several types of reactor. Below is the list of the most common reactors used in chemical process plants.

(1) Batch reactor
(2) Continuous reactor
(3) Semi – continuous reactor.
(4) Tank reactor
(5) Tubular reactors
(6) Tower reactor
(7) Fluidized – bed – reactor.

2.2.5 DESIGN ELEMENT OF A CONTROL SYSTEM

(A) VARIABLES

In attempting to design a control system that will satisfy the control needs for a chemical process one must be able to identify and classify the variables associated with the chemical process.

The variables (flow rates, temperature, pressure, concentrations e.t.c) associated with a chemical process are divided into two:

(1) Input variables
(2) Output variables.

The input variables can be further classified into two categories:

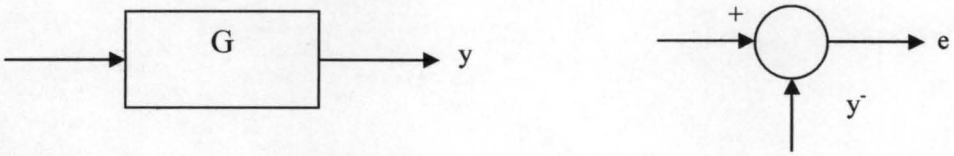
(1) Manipulated (or adjustable) variables.
(2) Disturbances.

The output variables are also classified into two:

(1) Measured output variables
(2) Unmeasured output variables.

(B) BLOCK DIAGRAMS

For control problems, it is helpful to use a block diagram to show the functional relationship between input and output.



(a) $y = Gx$

$y = f(x,t)$

(a) Dynamic relationship

(b) $e = x^{-y}$

(b) Comparison of signals

FIG. 2.4: BLOCK DIAGRAM COMPONENT

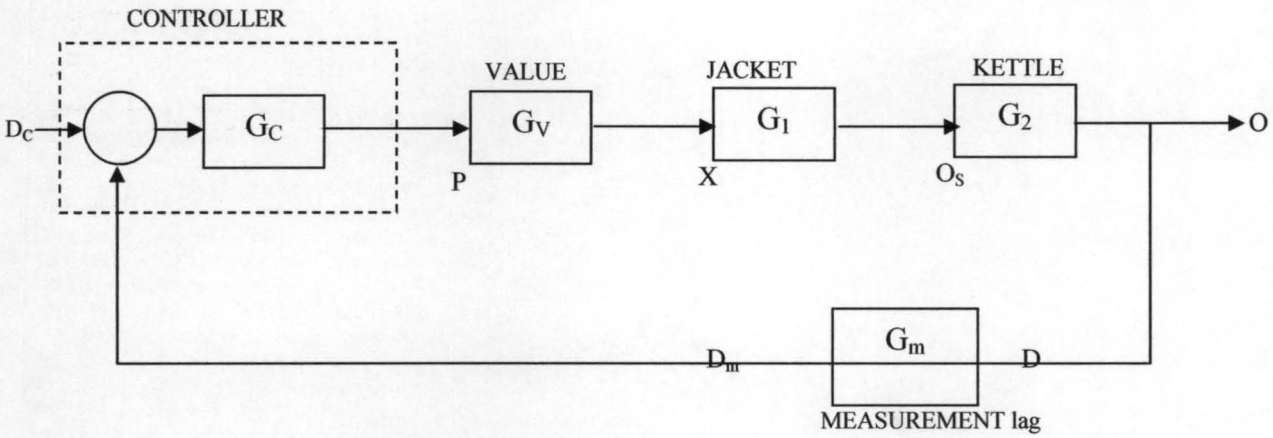


FIG. 2.5: BLOCK DIAGRAM FOR A TEMPERATURE CONTROL SYSTEM (PROCESS CONTROL HARRIOT)

(C) STATE VARIABLE AND STATE EQUATIONS FOR THE CHEMICAL PROCESS

In order to characterize a processing system (tank, batch reactor, distillation column e.t.c.) and its behavior, the following are needed.

(i) A set of fundamental dependent quantities whose values will describe the natural state

(ii) A set of equations, in the variables above which will describe how the natural state of the given system changes with time.

For most of the processing systems of interest to a chemical engineer there are only three such fundamental quantities: mass, energy, and momentum: quite often, though, the fundamental dependent variables cannot be measured directly and conveniently. In such cases other variables are selected, which can be measured conveniently, and when grouped appropriately they determine the value of the fundamental variables. Thus, mass, energy and momentum can be characterized by variables such as density, concentration, temperature, pressure and flow rate.

These characterizing variables are called state variables and their values define the state of a processing system. The equations that relate the state variable (dependent variables) to the various independent variables are derived from application of the conservation principle on the fundamental quantities and are called state equations.

The principle of conservation of a quantity S states that

ACCUMULATION OF S WITHIN A SYSTEM / TIME PERIOD = FLOW OF S IN THE SYSTEM / TIME PERIOD - FLOW OF S OUT OF THE / TIME PERIOD

+ AMOUNT OF S GENERATED WITHIN SYSTEM / TIME PERIOD - AMOUNT OF S CONSUMED WITHIN A SYSTEM / TIME PERIOD.

The quantity S can be any of the following fundamental quantities:

Total mass.

Mass of individual component

Total energy

The balance equations for these quantities are given as.

Total mass balance.

$$\frac{d(pv)}{dt} = \sum_{r:inlet} \rho_i F_i - \sum_{j:outlet} \rho_j F_j \qquad 2.1$$

Mass balance on component A.

$$\frac{d(n_A)}{dt} = \frac{d(C_A V)}{dt} = \sum_{r:inlet} C_{A_i} F_i - \sum_{j:outlet} C_{Aj} F_j \pm rV \qquad 2.2$$

Where the variables in the above equations are

ρ = density of the material in the system

ρ_i = density of the material in the i^{th} inlet stream

ρ_j = density of the material in the j^{th} outlet stream

V = Total volume of the system.

F_i = Volumetric flow rate of the i^{th} inlet stream

F_j = Volumetric flow rate of the j^{th} output stream

C_A = Molar concentration (Moles/volume) of A in the system.

C_{Ai} = Molar concentration of A in the i^{th} inlet

C_{Aj} = Molar concentration of A in the j^{th} output

r = reaction rate per unit volume for component A in the system.

2.3 THE MATHEMATICAL THEORY OF BIOLOGICAL NEURAL NETWORK

According to Pellionisz (1990), brain theory or the mathematical theory of biological neural network is only in its infancy. There is a strong belief that the mathematics underlying brain function is neither algebra nor even calculus, but Geometry, particularly fractal geometry in the neuron structure.

In this section, it is shown that the mathematics associated with the functioning of biological neural networks is not fractal geometry only. In order to understand the mathematical theories or concepts surrounding biological neural network, it requires a good knowledge or understanding of the basic mathematical theories/concepts underlying the functionality, dynamics, capabilities and general behavior of Neural Network (System). Our finding and other researches reveal that some of the mathematical concepts/theories that are closely related to neural network theory and operations include.

(1) Vector and linear algebra.

(2) Matrices

(3) Pseudo – inverse.

(4) Principle Component Analysis (PCA)

(5) Statistics and Probability.

(6) Calculus.

(7) Differential Equations.

(8) Fuzzy set theory and fuzzy logic.

(9) Information theory.

(10) Lyapunov exponents.

(11) Fractal Dimension.

(12) Non – Linear System theory.

2.4 MATHEMATICAL MODEL OF A NEURAL NETWORK

The basic function of a neuron is to concept inputs add them in some fashion and produce an output. A neuron can accept many inputs x_i. If there are n inputs, associated with each input neuron is a weight vector W_i and a bais O. A neuron yields its output by performing the weighted sum, adding a bias and then passes the result through a non – linear function known as the activation function or linear combiner.

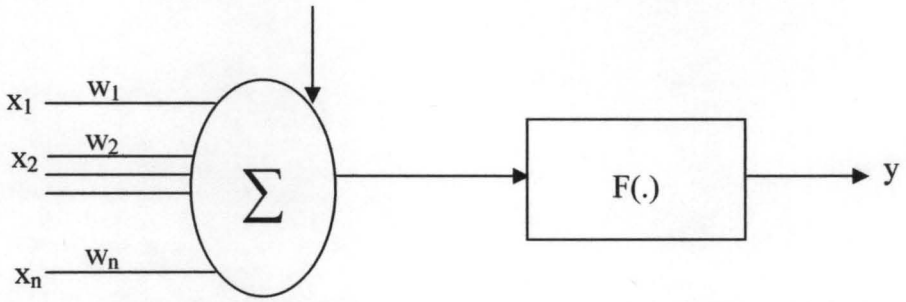


FIG. 2.6 INPUTS/OUTPUT TO A SINGLE MODEL NEURON A NETWORK

Here we define the input to the neuron as a vector x of n – tupelo. i.e.

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \cdot \\ \cdot \\ X_n \end{pmatrix} \text{ or } X^T = \begin{pmatrix} X_1, X_2, \ldots X_n \end{pmatrix}$$

The model neuron calculates the weighted sum of its inputs as follows:

It takes the first input, multiplies it by the weight on the input line, and does the same for the next input and so on, adding them all up at the end. This sum is then compared in the neuron with the threshold value.

This is written as

Total input $= W_1X_1 + W_2X_2 + \ldots + W_nX_n.$

$$= \sum_{i=1}^{n} W_1X_1 \qquad 2.3$$

In a multi-layer feed forward Network, the neutrons are arranged in consecutive layers (fig2.7) so that inputs to neutron in a particular layer are composed solely to the outputs of the neutrons in the immediately preceding layers

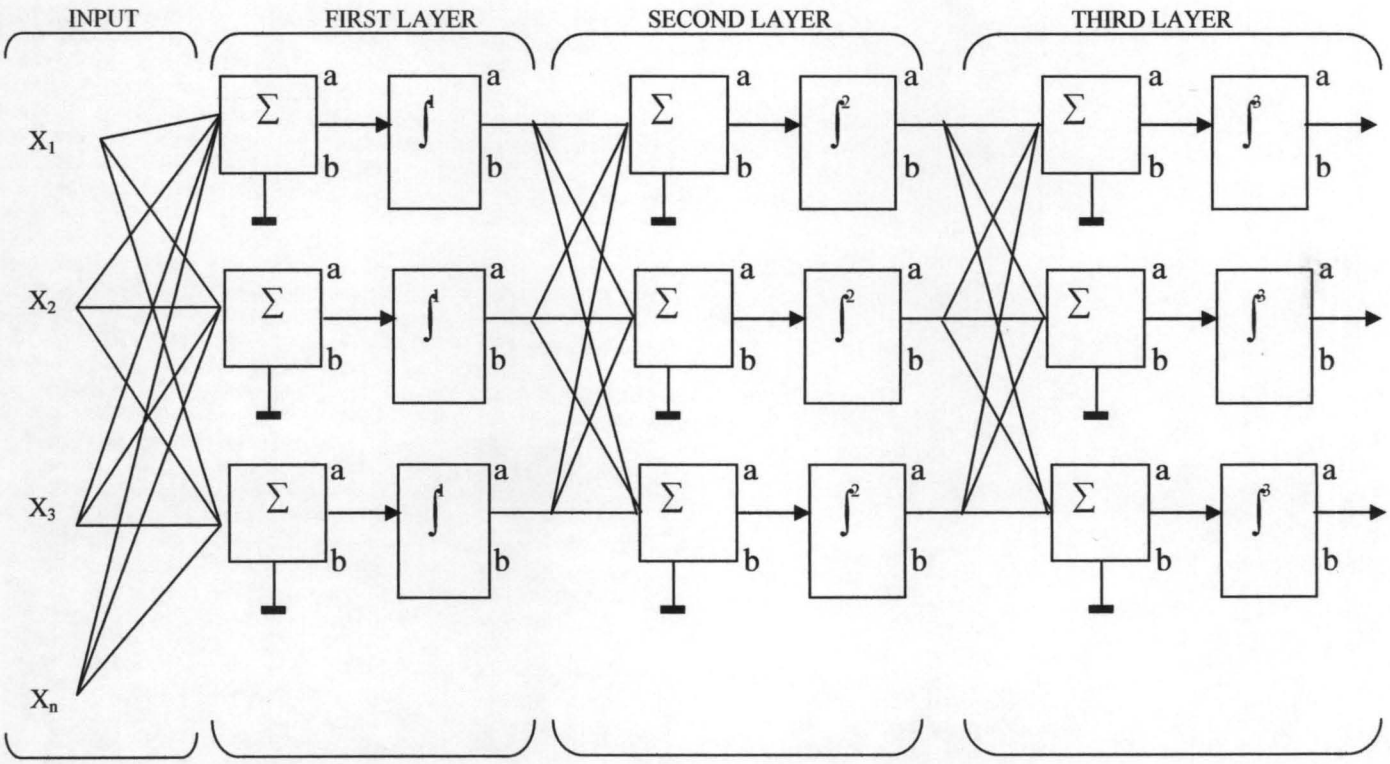


FIG. 2.7 A MULTIPLAYER FEEDWARD PERCEPTRON MATLAB NEUTRAL NETWORK.

Now, we present the forgoing using mathematical notations, for the purpose of designating a particular neutron, we use subscript notations to identify the layer and the position within the layer.

Let the vectors $xp_1 = [xp11, xp12 \ldots\ldots xp_1n]^T$

Where T denotes transpose, be the inputs vectors to any neutron in the 1^{th} layer corresponding to the P^{th} pattern presented to the net. Associated with the n^{th} layer is a weight vector.

W 1n =[W 1n1, W $1n_2$ .....w1n N] ^{T} and bias Oin; is the number of weights in the layer.

The neutron forms the sum:

$$Y_{pin} = +X_{pi}^T W_{in} + Q_{in}. \qquad 2.4$$

Augmenting Xpi by 1 and W_{in} by O_{in} allows us to write the sum in a simplified form as. $Y_{pin} = X_{pi}^{T} W_{in}$ 2.5

Where X_{pi} by 1 and w_{in} are now N +1 vectors.

The outputs of the neutron is

$$Y_{pin} = F [Y_{pin}]. \quad 2.6$$

The output to any neutron in the next layer is then

$$X_{P1+1} = [X_P^{1+1}, X_P^{I+1,2} \ldots\ldots XP^{1+1;}] \quad 2.7$$

$$= [ZP_{pli}, Z_{pl2,} \ldots\ldots Z_{pli}]^T$$

Where j is the number of neutrons in the layer.

The neutron in the $(L+1)^{st}$ layer generate the outputs ,which are inputs to neutrons in the next layer, and this continues to the neutron in the output layer whose output is the output of the net

A widely used activation function is the sigmoid function, given by

$$Fs(Y) = \frac{1-\ell^{-sy}}{1+\ell^{-sy}} \quad \begin{cases} 1 \\ \\ -1 \end{cases} \quad 2.8$$

Where the parameter S specifies the steepness of the curve. The sigmoid function is a smooth switch function having the property of

$$F(y) \rightarrow \begin{cases} 1 & \text{as } y \Rightarrow \infty \\ \\ -1 & \text{as } y \Rightarrow \infty \end{cases} \quad 2.9$$

Other possible activation functions are the arc tangent function given by

$$F(y) = \frac{2}{\pi} \text{arc} - \tan(y) \quad 2.10$$

And the hyperbolic – tangent function given by

$$F(y) = \frac{\left(\ell^{y} - \ell^{-y}\right)}{(\ell^{y} + \ell^{-y})} \quad \begin{cases} 1 \\ \\ -1 \end{cases} \quad 2.11$$

All these logistic functions are bounded, continuous monotonic and continuously differentiable.

2.4.1 CONCEPT LEARNING IN ARTIFICIAL NEURAL NETWORKS

DEFINITION 2.1

A concept is the mental representation of a category of objects in the world. Concept is grouped into classes of equivalence. The mechanism by which intelligent systems group physically distinct objects into classes of equivalence are probably among the most fundamental aspects of cognition. Without these mechanisms, every instant of each type of object, events or situation would appear new every time it is encountered.

Since we are dealing with a dynamical environment which compensate for variations in input stimuli the Neural Network learning process should posses an adaptive learning property. There are several iterative learning procedures that have been developed for a variety of Artificial Neural Network (ANN) architectures and much interest and research s still been focused on their learning efficiencies. Jacobs (1988), Giles and Maxwell (1987), Werbos (1988), Hinton (1989).

Learning in ANN is accomplished in one of the following ways:

(1) By the establishment of connections between nodes.

(2) Adjustment of the weight values on the links connecting nodes.

(3) Adjustment of the threshold values of node activation function or combination of the three operations. If a bias input is included with each of the nodes in a network and the number of initial nodes and interconnections is sufficient for the application, it is possible to learn through weight adjustments alone, since the bias weight can serve as the threshold value.

Learning method for Artificial Neural Network (ANN) can be classified as one of three basic types:

(1) Supervised learning

(2) Reinforcement learning

(3) Unsupervised learning

2.4.2 PERCEPTRONS

The term, perceptron was first used by Rosenblatt to refer to a network that has adjustable thresholds on individual unit from which they could learn by systematically adjusting their weight (Rosenblatt 1962).

The term, perceptron, is used to refer to the class of two layer feed forward networks:

(a) Those whose first layer units have fixed functions with fixed connections weight from inputs and

(b) Those whose connections weights linking this first layer to the second layer of output are learnable, together with the thresholds of the units in that output layer.

We use the term (multi – layer learning network or multi-layer perceptron) to denote feed-forward networks that have more than one layer of learnable parameters.

DEFINITION 2.2

Let $R = \{X_1, X_2 \ldots X_n\}$ be a set of n real variables X_i , $i = 1,2, \ldots, n$ defined over an n-dimensional finite subspace in R^n. The components of R are sometimes arranged as a two-dimensional array referred to as a Retina. A predicate or decision function defined on R will be 1 and 0 or 1 or –

DEFINITION 2.3

Let Φ be a family of M(partial) function on the elements of R. The ψ is a linear threshold function (LTF) with respect to Φ if there exists set of coefficients $\{W_i\}_{i=1}^{m+1}$ such that

$$\Psi(x) = T_b\left(\sum_{i=1}^{m} W_i \phi_i(x) + W_{m+1}\right) = T_b\left(\sum_{i=1}^{m} W_i y_i + W_{m+1}\right) \qquad 2.12$$

$$\text{For } X = \{X_i \ldots X_k\} \subseteq R \text{ with } 1 \le i \le k \le n,\ y_i = \phi_i(x) \in \Phi \qquad 2.13$$

$$T_b(x) = \begin{cases} 1 \text{ if } x > 0 \\ \alpha \text{ if } x \le 0 \end{cases} \qquad 2.14$$

where α may be 0, - 1 or a small positive number.

DEFINITION 2.4

A perceptron is a feed forward network that has only one layer of fixed processing units and trainable threshold logic unit (TLU) and is capable of computing $\psi \in L(\Phi)$ for a given Φ by adjusting the weights and the threshold of its trainable TLU. The fixed processing unit computing may be realized by any fixed device (Bose and Liang 1996).

2.4.3 LINEAR SEPARABILITY OF TRAINING PATTERNS

Let the input to a network be denoted by the n-dimensional vector $X = (x_i + x_2 \ldots x_n)^T \in R^n$. An input vector is called a pattern, an exemplar, or a sample, and n-dimensional space R^n is called the pattern space.

In this section, it is assumed that every input X contains all the variables in R.

Therefore, an input X can be denoted as an n-dimensional vector $X = (X_1, X_2, \ldots X_n)^T \in R^n$ an input vector is called a pattern, exemplar, or a sample, and the n-dimensional space R^n is called a pattern space. Suppose that there are M neurons in the first layer.

The output from the J^{th} neuron in this layer is denoted by

$$Y_j = \emptyset_j(x) \text{ for } j = 1, 2, \ldots M. \qquad 2.15$$

The vector $(y_1, y_2, \ldots y_m)^T = \Phi(x) = (\emptyset_1(x)\ \emptyset_2(x) \ldots \emptyset_m(x))^T$ which is the input vector to the training TLU, is called a first – layer image pattern.

It represents the image of the original pattern x in the space R^M with $\emptyset_1(x)$ $\emptyset_2(x) \ldots \emptyset_m(x)$ as the coordinate axes.

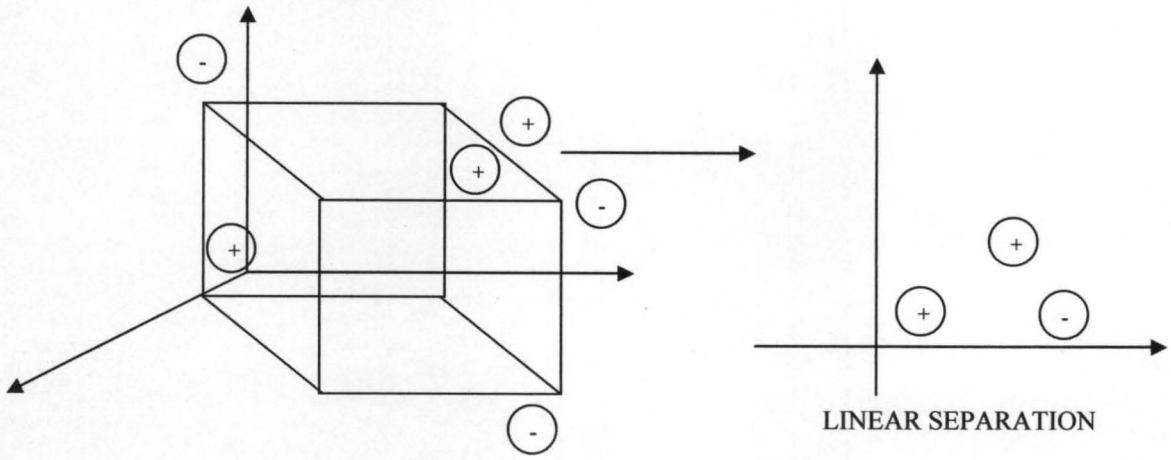
The space to which the image patterns belong is called the image pattern space or the first – layer image space. The mapping $\Phi(x)$: $R^n \rightarrow R^M$ from the original pattern space (referred to as the feature space) to the image pattern space, as illustrated below is nonlinear.

A surface that separates the patterns into different classes is called descision surface. Patterns belonging to one class all lie on one side of the decision surface, and patterns belonging to the other class lie on the other side of the side of the decision surface. In the case of a perceptron, the decision surface is the hyper plane defined in the image pattern space by

$$\sum_{i=1}^{m} w_i y_i + w_m y_i = 0$$

Perceptron categorizes a set of image patterns into two classes according to whether

$$\sum_{i=1}^{m} w_i y_i + w_{m+1} > 0 \text{ or } \sum_{i=1}^{m} w_i y_i + w_{m+1} < 0$$ and the hyperplane in A nonlinear map pattern space n = 3.



NOT LINEARLY SEPARABLE

FIG. 2.8 LINEAR SEPARABILITY

Case is called a linear dichotomy. A linear threshold function then defines linear separability of the image pattern $\{Ø_1(x)\ Ø_2(x) \ldots Ø_m(x)^T\}\ X \subseteq R$ in the image pattern space.

2.5 PERCEPTRON LEARNING ALGORITHMS

Let the training exemplar set be $\{X(i)\}_{i-1}^{N}$ where i is the index associated with the i^{th} training pattern. For simplicity, suppose that these exemplars are members of either of two known classes, C_+ and C_-. Let the value of the real-valued variable direction in which E(w) will decrease at the fastest possible rate, and therefore the weight update equation is

$$W(K + 1) = w(k) - c\ \nabla E \qquad 2.16$$

where c is a suitable constant. If we define an error function at an iteration as

$$\ell_K (w(k), y(k) = {}^1/_2 (|w(k).y(k)| - w(k).y(k)\ ; \qquad 2.17$$

then

$$\frac{\partial Ek(w(k) \quad y(K)}{\partial w(K)} = ½ (y(k)\ Tb(w\ (k).y(k)----y(k) \qquad 2.18$$

Where

$$Tb\,(w(k).\ y(K)) = \begin{cases} 1 \text{ if } w(k).\ y(k) > 0 \\ \\ -1 \text{ if } w(k).\ y(k) < 0 \end{cases} \qquad 2.19$$

Substitution of the equation (2.18 into 2.16 yields the fixed increment rule.

$$W(k+1) = W(k) + \frac{c}{2}\,(y(k) - y(k)\,Tb\,(w(k).\ y(k))) \qquad 2.20$$

The error function of equation 2. 17 is minimum when w (k). y (k) > 0. Note that this procedure differs from the stand and gradient descent algorithm in that the error function is different for each training pattern.

The foregoing error function can be considered as an estimate of the true function

$$E(w) = \epsilon_K\,(w, y(k)). \qquad 2.21$$

That should be minimized, where Y is the set of all training pattern. It can be shown that all solutions must lie in a single open convex polyhedral cone with its vertex at the origin. That is, the error function E (x) in the weight space has a unit unique valley with flat bottom. Therefore, convergence is guaranteed with gradient decent when the step size is appropriately chosen.

2.5.1. THE PERCEPTRON CONVERGENCE THEOREM

THEOREM 2.1

If the training set is linearly separable, i.e. if there exist a W such that w.y > 0 for all adjusted augmented training pattern, then perceptron learning using the fixed-increment rule will find a solution. W* infinite time such that W*. y> 0 for training patterns. In other words, there exist

K_o such that $W(k_o) = W(K0+1) = W(k_o + 2) =$ ------- and $W^* = W(k_o)$ 2.22

PROOF

Let Y be the set of all training patterns. Although Y is a finite set, the training patterns are presented repeatedly in learning. For national convenience, we feel free to donate y (1) by yi and w (k) by wk. Re-label the training patterns in successive steps of the algorithm as y_1, y_2----y_k-----

Let the corresponding weight vectors be $w_1, w_2, \ldots w_k \ldots$ Assume that the pattern and weight vectors at those steps where there is no change to the previous weight vectors, i.e $W_{k+1} = W_k$ are removed from the sequences

Since in each step where correction is needed, $W_j \cdot Y_j \leq 0$ and the weight vector is up dated using the fixed-increment rule with C = 1, we have

$W_{k+1} = W_1 = + Y_1 + Y_2 + \ldots\ldots ty_k$

Because the patterns are linearly separable, there exists a solution region W such that any WEW, w.y > 0 for all training patterns. Picked a W* EW (the w * here is chosen arbitrarily since W* is any of the winning neutrons within the solution region w), and

$$\text{let } \min_{y \in Y} y \bullet w^* = \alpha \text{ and } \beta = W_i \cdot W^* \qquad 2.23$$

Where $\alpha > 0$. Taking the inner product with w* of both sides of equation 2.20, we have

$$W_{k+1} \cdot W^* = W_1 \cdot W^* + y_1 \cdot W^* + y_2 \cdot W^* + \ldots\ldots ty_k W^* \qquad 2.24$$

From equation 2.23 noting that all the $y_i \cdot W^*$ terms in equation 2.22 are positive, we have

$$W_{k+1} W^* \geq K\alpha + \beta \qquad 2.25$$

$$\text{Since } (W_{k+1} \cdot W^*)^2 \leq |W_{K+1}|^2 |W^*|^2, \qquad 2.26$$

$$\text{Therefore, } |W_{k+1}|^2 \geq \left(\frac{(W_{K+1} \bullet W^*)^2}{|W^*|^2} \geq \frac{(K\alpha + \beta)^2}{|W^*|^2}\right. \qquad 2.27$$

Thus, the squared magnitude of the weight vector grows at least quadratically with number of correction steps. Next, we obtain an upper bound for the squared magnitude of the weight vector growth from another line of reasoning. Since

$W_{j+1} = W_j y_j$ for all j, we have

$$\left|W_{j+1}\right|^2 = \left|W_j\right|^2 + 2W_j \bullet y_j + \left|y_j\right|^2 \qquad 2.28$$

Since Wj . yj ≤ 0 (why ?)

$$\left|W_{j+1}\right|^2 - \left|W_j\right|^2 \leq \left|y_j\right|^2 \text{ for all j.} \qquad 2.29$$

The preceding inequality is summed over j = 1, 2 ..... k and simplified to yield.

$$\left|W_{J+1}\right|^2 \leq \sum_{j-1}^{K} \left|y_j\right|^2 + \left|w_j\right|^2 \leq kA + \left|w_i\right|^2 \qquad 2.30$$

Where A= max yϵY $\left|y\right|^2$

This inequality shows that the squared length of the weight vector cannot grow faster than linearly wit the number of correction steps

For sufficiently large k equation (2.26) and (2.30). becomes contradictory

Therefore, k cannot be larger than k_m which is the solution to the equation.

$$K_m A + \left|w_1\right|^2 = \frac{(km\alpha + \beta)^2}{\left|w^*\right|^2} \qquad 2.31$$

Therefore, the number of correction step must be less than km if the patterns are linearly separable. The training patterns are repeated until a solution vector is found the proof of this theorem is due to Rosenblath

2.6 HOW NEURAL NETWORK LEARNS (TRAINING IN THE NET)

Training in a neural net can be viewed as mapping a set of input vectors to another set of desired output vectors (supervised training). To train the net to perform the desired mapping, we apply an input vector to the net and compare the actual output of the net with the desired output (the output vector corresponding to the

applied input). The difference between the actual output and the desired output (i.e. the error) is used to update weights and biases associated with every neutron in the net, until this difference average over every input and output pair is below a specified tolerance.

A net performs the desired mapping when each neuron in the net yields a correct response. Training the net, hence implies training each neutron in the net. Now, assume that the desired output values of the neuron are known.

The output values of the linear combiner dp, p =1, 2.....m (where m is the number of patterns in the input are known. Then, the error of a node corresponding to pattern P is

$$E_p = \frac{1}{2}(dp - Yp)^2 \qquad 2.32$$

Where, $Yp = W^T X_p$ is the actual output of the linear combiner. s

The total mean squared error of the node is

$$E_T = \sum_{p=1}^{m} E_P \qquad 2.33$$

To find the optimum weight vector that minimize E_T, we take the gradient of E_T with respect to W and set it to zero, as follows:

$$\nabla E = \sum_{p=1}^{m} (d_P - W^T X_P) X_P = 0 \qquad 2.34$$

Hence, we have $$\left(\sum_{p=1}^{m} (X_P X_P^T) \right) W = \sum_{p=1}^{m} (d_P X_P) \qquad 2.35$$

Defining

$$R = \sum_{p=1}^{m} X_P X_P^T \qquad 2.36$$

and

$$P = \sum_{p=1}^{m} d_P X_P \qquad 2.37$$

We can write equation 2.34 in matrix form as $RW = P$ 2.38

The matrix R can be interpreted as the correlation matrix of the input set and the vector p as the cross correlation between the training patterns and the corresponding desired responses.

Equation 2.38 is referred to as the deterministic normal equation in the context of adaptive filtering, and the optimum weight vector is the solution to (2.38) equation (2.38) can be solved iteratively by a number of descent methods, such as the steepest descent method, which yields the popular Delta – leaning rule, by conjugate gradient method, and several other quasi-Newton methods.

2.6.1 BACK PROPAGATION OF ERROR

Recall (2.33)

Let $E_{TP} = \sum E_{TLN}$ 2.39

Where E_{TP} is the total mean square error of the net association with the pth training pattern. L is the number of layer in the network and the index n is over the neurons in the output layer..

From 2.39 and 2.32. We know that the error

$\delta_{PL\,n} = d_{pLn} - Y_{pLn}$ 2.40

Associated with the P^{th} training pattern at the n^{th} output node can be expressed as

$$\delta_{pln} = \frac{-\partial E_{TP}}{\partial Y_{pln}} \qquad 2.41$$

Based on this observation, we define the error associated with any node; hidden or otherwise as the derivative of E_{TP} w.r.t the linear combiner output at the node:

$$\delta_{pln} = \frac{-\partial E_{TP}}{\partial Z_{Pln}} \qquad 2.42$$

The task now is to calculate d pln at every node. For the output nodes the desired result is given by (2.40) for the hidden layers (i.e L =1,2….L -1), we have that:

$$\delta_{pln} = \frac{\partial E_{TP}}{\partial Y_{pln}} = \frac{-\partial E_{TP}}{\partial Z_{pln}} \cdot \frac{\partial Z_{TP}}{\partial Y_{pln}} \qquad 2.43$$

By the chain rule, the first factor in 2.43 can be expressed as a linear combination of the derivations of ETP w.r.t the variables associated with the node in the next L+1 layer as follows:

$$\frac{\partial E_{TP}}{\partial Z_{pln}} = \sum \frac{\partial E_{TP}}{\partial Y_{pl+1r}} \cdot \frac{\partial Y_{pl+1r}}{\partial Z_{pln}} \qquad 2.44$$

Where r is over the nodes in the layer L+1 using 2.42 in 2.44 w arrive at the recursion

$$\delta_{pln} = F^{1}(Y_{pln})\bar{Z}\delta_{pl} + 1.2W_{l+1.r.n} \qquad 2.45$$

Where

$$F^{1}(Y_{pln}) = \frac{\partial Z_{pln}}{\partial Y_{pln}} \qquad 2.46$$

By definition

To summarize, the output errors calculated from the 2.40 are back propagated to the hidden layers by recursively using 2.45.

Once the error is available in a particular node, it can be used to update the weights, as will be shown in what follows

Let the weight vector at iteration t be Win (t). The minimizing E (Ep) or taking the gradient with respect to the weight vector and adjusting Win (t) in the direction of steepest descent yields

$$W_{ln}(t+1) = W_{ln}(t) + \mu(-\nabla Ep) \qquad 2.47$$

Where n is the step size.

Since

$$Y_{pln} = W_{ln}^{T} \; Xpl \qquad 2.48$$

We can write.

$$\nabla_{EP} = \frac{-\delta E_{TP}}{\partial Y_{pln}} . \nabla_{Ypln} \qquad 2.49$$

Substituting 2.42 in 2.49 and observing from 2. 47 that $\nabla Yp\,ln = xpl$. we have

$$\nabla_{EP} = -\delta p\,ln\,Xpl \qquad 2.50$$

And hence

$$W_{in}(t+1) = W_{in}(t) + \mu\,\delta p\,ln\,Xpl \qquad 2.51$$

Equation 2.51 is variously called the Delta rule or method of steepest descent. When 2.51 is used in conjunction with 2.42 and 2.45 the resulting iterative scheme, is referred to as the back propagation. (Rumelhart et al 1987), (Parker, 1985) and Hacht-Nielson; 1987).

2.7 THE CONJUGATE GRADIENT ALGORITHM

The back propagation algorithm was the first and until recent, the only algorithm to train feed forward Multi layer perceptrons. We here present the CGM Variant.

Until recently, when the extended conjugation gradient Method (ECGM) was formulated, the conjugate gradient method (CGM) has been one of the most effective Method among the iterative method for solving linear system of equations (of the form in equation 2.38) as a minimization method. The CGM provides faster convergence for quadratic functional than gradient descent methods while avoiding computation of the inverse of the Hessian matrix.

FORMULATION

Since R is a positive definite, real symmetric n x n matrix, then the quadratic functional

$$F(W)=\frac{1}{2}W^{T}RW-W^{T}P+C \qquad 2.52$$

Has a unique minimum point W* which is a solution of the system of equations Qx = b

Since $\nabla f(x)=Qx-b$ 2.53

Then we can write

$Qx^{*}-b$

The minimization iteration updating method for (2.5.2) is given as

$$W_{K+1}=W_{K}+\alpha_{K}(RW_{K}-P) \qquad 2.54$$

Where

$$\alpha_{K}=\frac{(\nabla f(x_{K}))^{T}d_{K}}{d_{k}^{T}Qd_{K}} \qquad 2.55$$

The value of the step SIZE α_K that minimizes $f(X_{K+1})$ can be defined by setting

$$df\frac{(x_{K}-\alpha_{K})^{T}(Qx_{K}-b)}{d\alpha_{K}} \qquad 2.56$$

Yielding

$$\alpha_{K}=\frac{(Qx_{K}-b)^{T}(Qx_{K}-L)}{(Qx_{K}-b)^{T}Q(Qx_{K}-b)} \qquad 2.57$$

With the step size α_K chosen as in equation (2.55), we have the following important result on the convergence rate of the gradient descent method.

THEOREM 2.2

For any $X_0 \; E \; R^n$, the gradient descent method with α_K chosen as in equation (2.55) converges to the unique minimum point X^* of a quadratic function f. Moreover, the following inequality holds at every iteration K:

$$E(X_{K+1}) \leq \left(\frac{r-1}{r+1}\right)^2 E(X_K) \qquad 2.58$$

Where $E(x) = \frac{1}{2}(x - x^*)^T Q(x - x^*)$ and r is the condition number of the matrix Q, defined as the ratio of the largest and the smallest eigen values of Q

Hence we state conjugate Gradient Algorithm Being

Set $co = -\nabla E_0$

For each P.

$$\mu_P = -\frac{(\nabla E_P)^T Cp}{CP^T R^c P} \qquad 2.59$$

$$\alpha_P = -\frac{(\nabla E_P + 1)^T RcP}{C^T PR^C P} \qquad 2.60$$

$$W(L+1) = W(t) + \mu_p \, Cp \qquad 2.61$$

$$Cp+1 = -\nabla Ep + 1 + \alpha P^C P \qquad 2.62$$

If estimating R is not feasible, we may set up μp and αp constant: $\mu p = \mu$ and $\alpha p = \alpha$. In this case, after substituting (2.61) in (2.62) the rule by which the weight vector is updated becomes

$$W(t+1) = W(t) + \mu \, ((-\nabla Ep) + \mu \alpha^C p - 1 \qquad 2.63$$

But since,

$$Cp-1 = (1/\mu) \, (w(t) - W(t-1)\} \qquad 2.64$$

From (2.62),(2.64) is written as

$$W(t-1) = w(t) + \mu(-\nabla Ep) + \alpha W(t) - w(t-1) \qquad 2.65$$

2.7.1 THE MINIMUM OF A FUNCTIONAL

PROPOSITION 2.1

For a given quadratic functional of the form

$$F(x) = F_o + \frac{1}{2}[a, x] + \frac{1}{2}[X_1, Ax]$$ if the operation A is positive definite

then a minimum X*exists and it is unique and given by $X^* = -A^{-1}a$

PROOF:

$$F(x)=F_o+\langle a, x\rangle + \frac{1}{2}\langle x, Ax\rangle \qquad 2.66$$

Where x, a, AEH a Hilbert space.

Since A is symmetric

$$\rightarrow \langle x, Ay\rangle = \langle AX, Y\rangle \qquad 2.67$$

As in the classical extreme problems of the calculus ,we consider where the gradient is null to find the extremum.

$$\therefore F^1(x) = 0 + \langle a,\bullet\rangle + \frac{1}{2}\langle \bullet, Ax\rangle + \frac{1}{2}\langle x, A\bullet\rangle = \langle a + Ax,\bullet\rangle \qquad 2.68$$

Where the dot on RHS indicates the position of the argument of the functional. The gradient of the functional F is hence

$$g(X) = a + Ax \qquad 2.69$$

We consider X* such that

$$AX^* = -a \qquad 2.70$$

We assume for the moment that such an X* exist, and obviously for each an X* the gradient is null.

We evaluate,

$$F(X^*) = F_o\langle a, X^*\rangle + \frac{1}{2}\langle x^*, AX^*\rangle_t \qquad 2.71$$

$$= F_O\, 1/2\, \langle a, x*\rangle$$

Using (2.70) now we consider a perturbation about x* of the form $(x* + z)\epsilon\, \varepsilon$, for this element

$$F(X* \div Z) = F_O + \langle a, x* \div z\rangle + \tfrac{1}{2}\, \langle x* \div z, A\, \langle x x* \div z\rangle$$

$$= f_o\, \langle a, x*\rangle \div \langle a, z\rangle \div {}^1/_2 AX\ \ A x*, \rangle\, 1/2$$

$$\langle x* AZ\rangle + \tfrac{1}{2}\, \langle Z, AX*\rangle + 1/2\, \langle Z, AZ\rangle \qquad 2.72$$

$F_O + ½ \langle Z, AX* \rangle + ½ Z, AZ\rangle$

From (2.66) and fact that (Z, AX*) = (AZ,X*) since A is symmetric form (2.71) and (2.70)

$$F(\times * + z) = F(\times *) + ½ \langle Z, A \rangle \qquad 2.73$$

From (2.73) we draw the following conclusion about F.

1. If the linear operator A is positive definite i.e. (Z, AZ) $\rangle \circ \forall Z$

Then $\times *$ is a minimum of F

ii. If A is positive semi definite, 1.e. (Z, AZ) $\geq o \forall Z$

Then $\times *$ is a minimum from F but it is not necessarily unique.

iii. If A is indefinite.? Then $\times *$ does not exist.

2.7.2 DESCRIPTION OF THE CONJUGATION GRADIENT DESCENT

Recall (2.66)

$$\text{Min: } F(x) = F_0 + (a_1 x) + \tfrac{1}{2}(x_1 Ax) \qquad 2.74$$

With X and a in Hilbert space H (i.e a_1XEH) and A a positive definite symmetric linear operator.

From 2.52 we know that a unique minimum X* exists.

With the CGM, first element of the descent sequence X_o is unique, while the remaining members of the sequence are found as follows.

$$P_o = -g_o = -(a + AX_o) \qquad 2.75\text{ a}$$

$$X_{j+1} = X_j + \alpha_j P_j \qquad 2.75\text{ b}$$

$$\alpha_1 = \frac{\langle g_1 g_1 \rangle}{\langle P_j A P_j \rangle} \qquad 2.75\text{ c}$$

$$g_{j+1} = g_1 + \alpha_j A P_j \qquad 2.75\text{ d}$$

$$P_{i+1} = -g_i + 1 + \beta_i P_i \qquad 2.75\text{ e}$$

$$\beta_i = \frac{\langle g_{i+1}, g_{i+1} \rangle}{\langle g_1, g_i \rangle} \qquad 2.75\text{ f}$$

Where g_i, α_i and P_i denote respectively the gradient of F(x), the step length of the descent sequence, X_i and the descent direction at the i^{th} step.

It is obvious from (2.75c) that the sequence converges, that is $\alpha_i = 0$, if the gradient g_i is null.

We give an expression for the descent direction at the I [th] step P_i in terms of the gradients

Recall

$$P_o = -g_o$$

Thus.

$$P_i = -g_i + \beta_o P_o \qquad 2.76$$

$$= -g_i - \frac{\langle g_1 g_1 \rangle}{\langle g_o g_o \rangle} g_o \qquad 2.77$$

$$P_2 = -g_2 + \beta_1 P_1 \qquad 2.78$$

$$= -g_2 + \frac{\langle g_2 g_2 \rangle}{\langle g_1 g_1 \rangle}\left[-g_o - \frac{\langle g_1 g_1 \rangle}{\langle g_o g_o \rangle} g_o\right] \qquad 2.79$$

$$= -g_2 + \frac{\langle g_2 g_2 \rangle}{\langle g_1 g_1 \rangle} g_1 - \frac{\langle g_2 g_2 \rangle}{\langle g_o g_o \rangle} g_o \qquad 2.80$$

$$P_3 = -g_3 - \frac{\langle g_3 g_3 \rangle}{\langle g_2 g_2 \rangle} g_2 - \frac{\langle g_3 g_3 \rangle}{\langle g_1 g_1 \rangle} g_1 - \qquad 2.81$$

$$\frac{\langle g_3 g_3 \rangle}{\langle g_o g_o \rangle} g_o.$$

From which we can see the recursion relationship

$$P_K = -\langle g_K g_K \rangle \sum_{i=o}^{K} \frac{g_1}{\langle g_1 g_1 \rangle} \qquad 2.82$$

CHAPTER THREE
METHODOLOGY

3.1 INTRODUCTION TO PID CONTROLLER

A proportional- integral-derivative controller (PID controller) is a common feed back loop component in industrial control systems. The controller compares a measured value from a process with a reference set point value. The difference (or "error" signal) is then processed to calculate a new value for a manipulated process input that brings the process measured value back to its desired set point. Unlike simpler control algorithms, the PID controller can adjust process outputs based on the history and rate of change of the error signal which gives move accurate and stable control.

(It can be shown mathematically that PID loop will produce accurate, stable control in cases where a simple proportional control would either have a steady state error or would cause the process to oscillate). Unlike more complicated control algorithms based on optimal control theory PID controllers do not require advance mathematics to design and can be easily adjusted (or "tuned") to the desired application.

3.2 PROPERTIES OF A PID CONTROLLER

The PID loop adds positive correction, removing error from the process's controllable variable (its input). Differing terms are used in the process's control industry. The "process variable" is also called the "process's input" or "controller's output". The process's output is also called the "measurement" or "controller's input". This up a bit, down a bit movement of the process's input variable is how the PID loop automatically finds the correct level of input for the process. Removing the error "turns the control knob" adjusting the process's input to keep the process's measured output at the set point. The error is found by subtracting the measured quantity from the set point.

"PID" is named after its three correcting calculations, which all add to and adjust the controlled quantity. These additions are actually "subtractions" of error, because the proportions are usually negative.

3.2.1 PROPORTIONAL

To handle the present, the error is multiplied by a (negative) constant p (for "proportional") and added to (subtracting error from) the controlled quantity. P is only valid in the band over which a controller's output is proportional to the error of the system.

For example, for a heater, a controller with a proportional band of 10°c and a set point of 20°c would have an output of 100% at 10°c, 50% at 15°c and 10% at 19°c. Note that when the error is zero, a proportional controller's output is zero.

3.2.2 INTEGRAL

To handle the past, the error is integrated (added up) over a period of time, and then multiplied by a (negative) constant I (making an average) and added to (subtracting error from) the controlled quantity. Integral (I) average the measures error to find the process output's average error from the set point. A simple proportional system oscillates moving back and forth around the set point because there is nothing to remove the error when it over shoots? By adding a negative proportion of (i.e subtracting part of) the average error from the process input the average difference between the process output and the set point is always being reduced. Eventually, a well-tuned PID loop's process output will settle down at the set point.

3.2.3 DERIVATIVE

To handle the future the first derivative (the stop of the error) over time is calculated, and multiplied by another (negative) constant D, and also added to (subtracting error from) the controlled quantity. The derivative terms controls the response to a change in the system.

The larger the derivative term, the more rapidly controller responds to changes in the process's output. Its D term is the reason a PID loop is also called a "Predictive controller". The D term is a good thing to reduce when trying to dampen a controller's response to short term changes. Practical controller's for slow process can even do without D. Move technically, a PID loop can be characterized as a filer applied to a complex frequency-domain system. This is useful in order to calculate whether it will

actually reach a stable valve. If the values are chosen incorrectly the controlled process input can oscillate and the process output may never stay at the set point.

PID controller is called PI, PD, or P controller in absence of respective control actions. It may be noted that EWMA (Exponential Weighted Moving Average) controller is equivalent to PI .The generic transfer functions for a PID controller of the interacting form is given by the transfer function;

$$H(S) = P\frac{DS^2 + S + I}{S + C}, \qquad 3.1$$

with C being a constant which depends on the band width of the controlled system.

Traditionally we can write.

Output = P contribution +I contribution + D contribution - 3.2

Where P contribution. I contribution and D contribution are the feedback contributions from the PID controller, defined below

$$\text{P contribution} = K_p\, e^{(t)} \qquad 3.3a$$

$$\text{I contribution} = K_I \int_{-\infty}^{t} e^{(t)}\, dt \qquad 3.3b$$

$$\text{D contribution} = K_d \frac{de^{(t)}}{dt} \qquad 3.3c$$

Where K_p, K_I, K_d are constant that are used to tune the PID control loop.

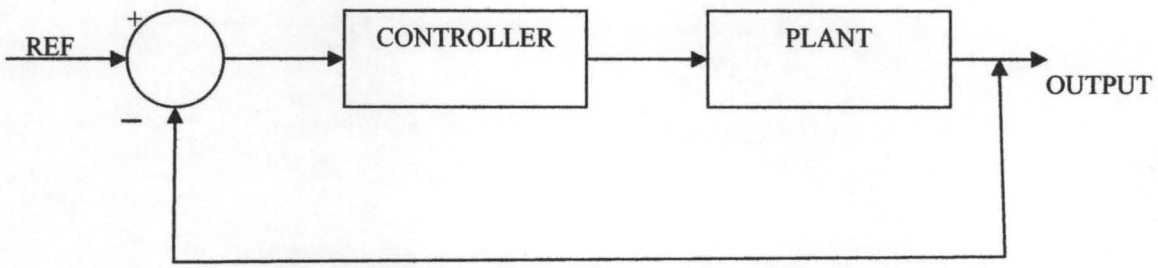


FIG. 3.1 FEEDBACK ONLY THE OUTPUT SIGNAL.

3.2.4 K_p: PORPORTIONAL GAIN - Larger Kp typically means faster response since the larger the error the larger the feedback to compensate. K_I : INTEGRAL GAIN – Larger K_I implies steady state errors are eliminated quicker. The trade-off is larger overshoot ANY negative error integrated during transient response must be integrated a way by positive error before we reach steady state.

3.2.5 K_d: DERIVATIVE GAIN - Larger K_d reduces overshoot, but slows down transient response. Normally it is implemented with K_p the I contrib. and D contrib. terms as well in the following form.

$$\text{Output} = K_p(e^{(t)}+K_I \int_{-\infty}^{t} e^{(t)}dt + K_d \frac{de^{(t)}}{dt} \quad 3.4$$

Most standard turning method, such as Ziegler. Nichols and others, are based on this form, as it reduces interaction. In this form, the K_i and K_d gains relate only to the dynamics of the process, and the K_p ((proportional gain) relates to the gain of the process. Often, one deals with discrete time interval. instead of the continuity.

The PID controller may also be dealt with recursively.

$$\text{Output}_{n+1} = \text{OUTPUT}_n + K_{pen} + K_{d(en-en-1)} \quad 3.5$$

Here, the first term is integral the second proportional and the third derivative. Note that in this form, K_i must be identically I, otherwise the controller will not even come close to converging to the set point. This isn't quite the same integral as in the continuous form, but its analogous. In piratical, most PID controller employs 3 slightly different constant which correspond to these proportional, integral, and derivative gain

3.3 PROPORTIONAL BAND: - (Often abbreviated Pb) This is the band where proportional gain acts upon. To get larger Kp we decrease Pb as follows:

$$K_p = \frac{1}{Pb} \quad 3.6$$

3.3.1 INTEGRAL TIME: - (Often abbreviated I_t) This is the time we finding the average error over. Because it is time, we conclude the following with a dimensional analysis

$$Ki = \frac{1}{It} \quad 3.7$$

3.3.2 DERIVATIVE TIME: - Often abbreviated D_t this is the time we evaluate the derivative of the error over. Because D_t is time, we conclude the following with dimensional analysis.

$$K_d = D_t. \quad 3.8$$

This example will show that the characteristics of each of the proportional (p), integral (I), and the derivative (D) controls, and how to use them to obtain a desired response. In this example, we will consider the following unity feed back system.

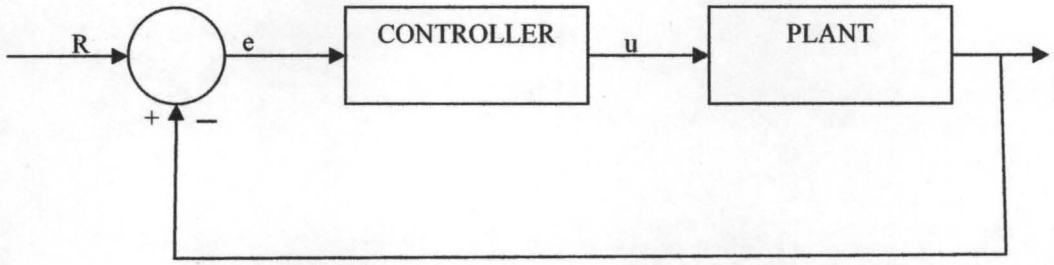


FIG 3.2: A SYSTEM TO BE CONTROLLED

Controller: provides the excitation for the plant, designed to control the over all system behavior. First, let us take a look at how the PID controller works in a closed-loop system using the show above. The variable (e) represents the tracking error the difference between the desired input value (R) and the actual output (Y). This error signal (e) will be sent to the PID controller, and the controller computes both the derivative and the integral of the error signal. The signal (U) just past the controller is now equal to the proportional gain (K_P) times the magnitude of the error plus the integral gain (K_I) times the integral of the error plus the derivative gain (K_d) times the derivative of the error. That is $U = K_p|e_p| + K_i|e_i| + K_d|e_d|$

The transfer function of the PID controller looks like the following.

$$K_p + \frac{KI}{S} + K_{DS} = \frac{K_D S + K_p S + K_I}{S} \qquad 3.9$$

where

K_p = proportional gain

K_I = integral gain

K_d = derivative gain

and

$$PI. = K_p e + K_I \int e dt \qquad 3.10$$

$$PU = K_p e + K_d \frac{de}{dt} \qquad 3.11$$

$$U = K_p e + K_I \int e dt + K_d \frac{de}{dt} \qquad 3.12$$

$$PID = K_p e + K_I \int e dt + K_p \frac{de}{dt}$$

From that controller the signal (u) will be sent to the plant, and the new output (y) will be obtained. This new output (y) will be sent back to the sensor again to find the new error signal (e). The controller takes this new error signal and computes its derivative and its integral again. This process goes on and on.

3.3.3 CHARACTERISTICS OF P, I, AND D CONTROLLERS

A proportional controller (K_p) will have the effect of reducing the rise time and will reduce, but never eliminate, the steady-state error. An integral control (K_I) will have effect of eliminating the steady-state error, but it may make the transient response worse. A derivative control (K_d) will have the effect of increasing the stability of the system, reducing the overshoot and improving the transient response. Effects of each of the controllers K_p, K_d and K_I on a closed-loop system are summarized in the table below.

CL RESPONSE	RISE TIME	OVERSHOOT	SETTING TIME	S-S – ERROR
Kp	Decrease	Increase	Small change	Decrease
KI	Decrease	Increase	Increase	Eliminate
Kd	Small change	Decrease	Decrease	Small change

FIG 3.3 CHARACTERISTIC OF PID

Note: that these correlation may not be exactly accurate, because K_p, K_I and K_d are dependent of each other. In fact changing one of these variables can change the effect of the other two for this reason, the table should only be used as a reference when you are determining the value for K_I, K_p, and K_d.

Suppose we have a simple mass, spring and damper problem.

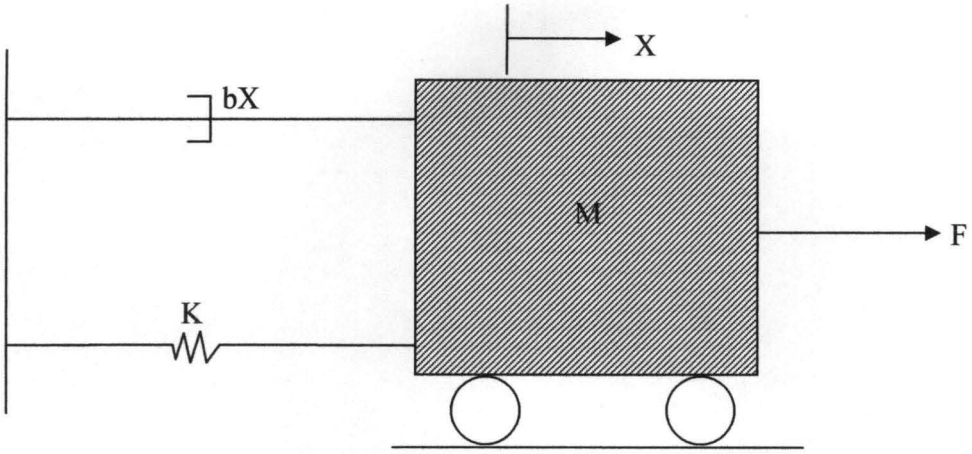


FIG. 3.4 SIMPLE MASS, SPRING AND DAMPERS

THE MODELLING EQUATION OF THIS SYSTEM is

$$Mx + bx + kx = F \text{ -------} \qquad 3.13$$

$$x(m + b + k) = F$$

$$x(T) = F$$

$$x = \frac{F}{T}$$

Taking the Laplace transform of the modeling equation (3.13), we obtain;

$$MS^2X(S) + bSX(S) + kX(S) = F(S) \qquad 3.14$$

The transfer function between the displacement X(S) and the input F(S) then becomes

$$\frac{X(S)}{F(S)} = \frac{1}{MS^2 + bS + K} \qquad 3.15$$

3.4 A STEP RESPONSE OF PID CONTROLLER

Lets first view the open-loop step response with a new m-file and add in the following code

```
num = 1
Den = [1,10,20]
STEP (Num, Den)
```

Running this m-file in the matLab command window will give us the plot shown below

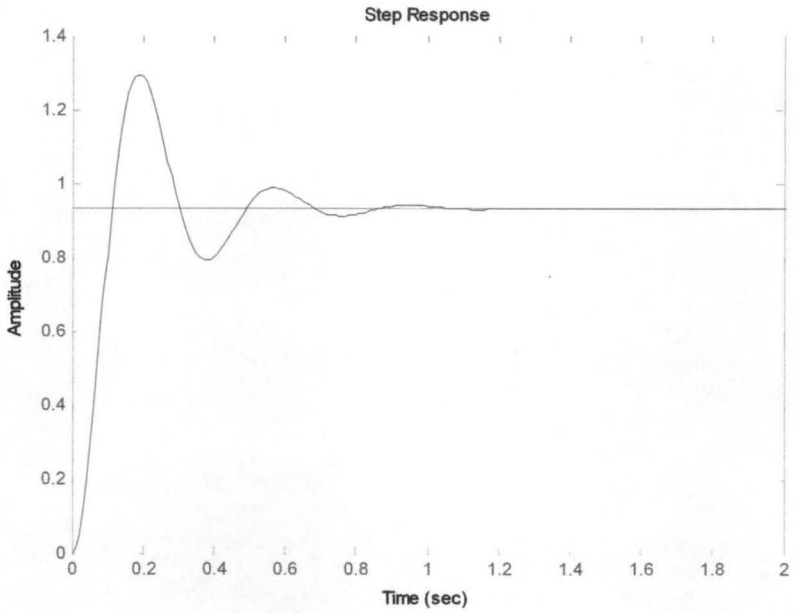


FIG. 3.5 OPEN LOOP STEP

The DC gain of the plant transfer function is 1/20 so 0.05 is the final value of the output to an unit step input. This corresponds to the steady-state error of 095, quite larger indeed. Furthermore, the rise time is about one second and the setting time is about 1.5sec. Let's design a controller that will reduce the setting time and eliminate the steady-state error.

EXAMPLE 3.1

PROPORTIONAL CONTROL

From the table shown above, we see that the proportional controller (Kp) reduces the rise time, increases the overshoot and reduces the steady-state error. The closed-loop transfer function of the above system with a proportional controller is

$$G_p(s) = \frac{1}{S^2+S+1} \qquad 3.16$$

$$U(s) = K_p E(s) \qquad 3.17$$

$$OLTF = \frac{K_p}{S^2+S+1} \qquad 3.18$$

$$CLTF = \frac{K_p}{S^2+S+1+K_p} \qquad 3.19$$

So from the table above we have

$$\frac{X(S)}{F(S)} = \frac{K_p}{S^2 + 10S + (20 + K_p)} \qquad 3.20$$

So, let the proportional gain (Kp) equals 300 and change the m-file to the following.

$K_p = 300$

Num = $[K_p]$

Den = $[1,10,20,+K_p]$

t = [0,0:01:2]

step (num, den t)

By running this m-file in the matLab command window will give us the following plot.

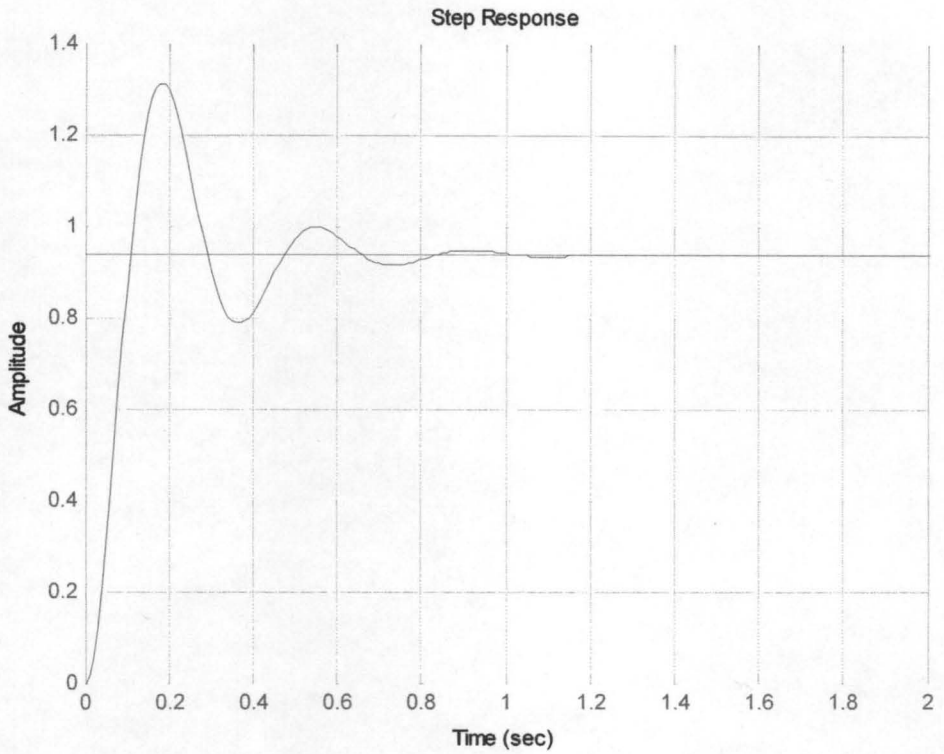


FIG. 3.6 CLOSED-LOOP STEP: Kp = 300

Note: The matLab function called loop can be used to obtaining a closed-loop transfer function directly from the open-loop transfer function (instead of obtaining closed-loop transfer function by hand). The following m-file uses the loop command that should give us the identical plot as the one shown above.

Num:= 1

Den = [1, 10, 20]

Kp = 300

[num cl, den cl] = cloop [Kp * num, den]

t = [0, 0.01, 2]

STEP (num cl, den cl, t)

The above plot shows that the proportional controller reduces both the rise time and the steady-state error, increases the overshoot, and reduces the setting time by a small amount.

EXAMPLE 3.2

PROPORTIONAL – DERIVATIVE CONTROL

$$U(s) = (K_p + K_{DS})\, E(s) \qquad 3.21$$

$$OLTF = \frac{K_{DS} + K_p}{S^2 + S + 1} \qquad 3.22$$

$$CLTF = \frac{K_{DS} + K_p}{S^2 + (1 + K_d)S + (1 + K_p)} \qquad 3.23$$

Now let's take a look at a PD control from the table shown below. We see that the derivative controller (Kd) reduces both the overshoot and the setting time. The close of loop transfer function of the given system with a PD controller is

$$\frac{X(S)}{F(S)} = \frac{K_{DS} + K_p}{S^2 + (10 + K_d)S + (20 + K_p)} \qquad 3.24$$

Let K_p be equal to 300 as before and let K_d, be 10; enter the following command into an m-file and run it in the matLab command window.

$K_p = 300$

$K_d = 10$

num = [K_d, K_p]

den = [1, 10+K_d, 20+K_p]

t = 0; 0.01; 2

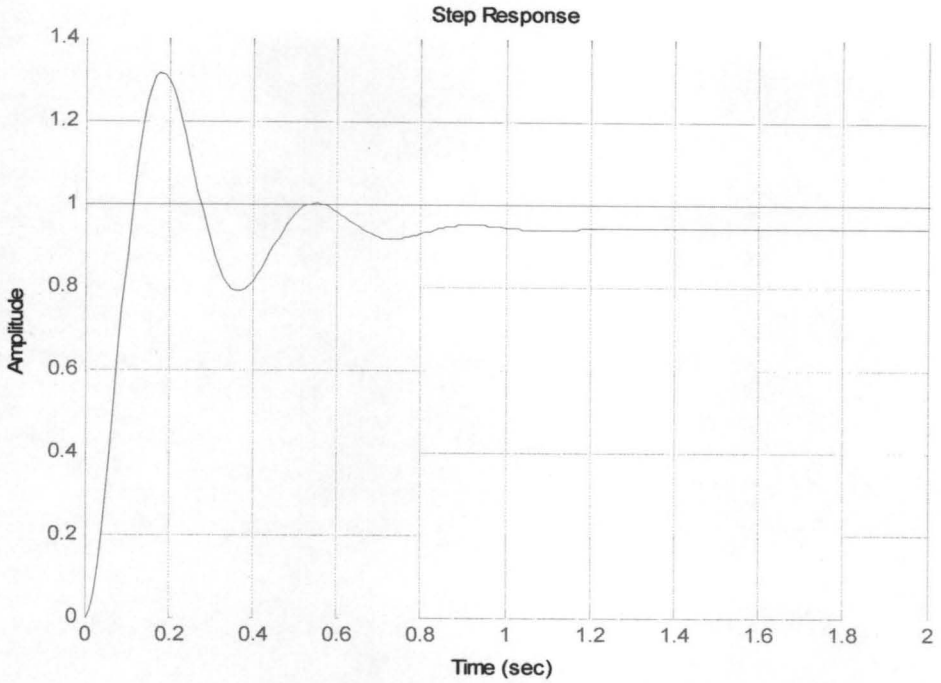


FIG. 3.7 CLOSED – LOOP STEP $K_p = 300$, $K_d = 10$

This plot shows that the derivative controller reduced both the overshoot and the settling time, and had small effect on the rise time and the steady state error.

EXAMPLE 3.3

PROPORTIONAL – INTEGRAL CONTROL

$$\frac{U(S)}{E(S)} = \left(K_p + \frac{K_I}{S}\right) \tag{3.25}$$

$$\frac{Y(S)}{E(S)} = \left(K_p + \frac{K_I}{S}\right) G_P(S) \tag{3.26}$$

$$= \left(\frac{Kp_S + K_I}{S}\right) G_P(S) \tag{3.27}$$

$$CLTF = \frac{K_P^S + K_I}{S^3 + S^2 + (1 + K_p)S + K_I} \tag{3.28}$$

Before going into PID control, let's take a look at a PI control from the table, we see that an integral controller (K_I) decreases the rise time, increases both the overshoot and the settling time and eliminates the steady-state error for the given system, the closed-loop transfer function with PI control is

$$\frac{X(S)}{F(S)} = \frac{K_P^S + K_I}{S^3 + 10S^2 + (20 + K_p)S + K_I} \qquad 3.29$$

Let's reduce the K_p to 30, and let K_I equal to 70 create an new m-file and enter the following commands.

$K_p = 300$

$K_I = 370$

num = [K_p . K_I]

den [1, 10, 20+K_p, K_I]

t . 0.0, 01, 2.

Step (num, den, t)

We run this m-file in the matLab command window and obtain the following plot.

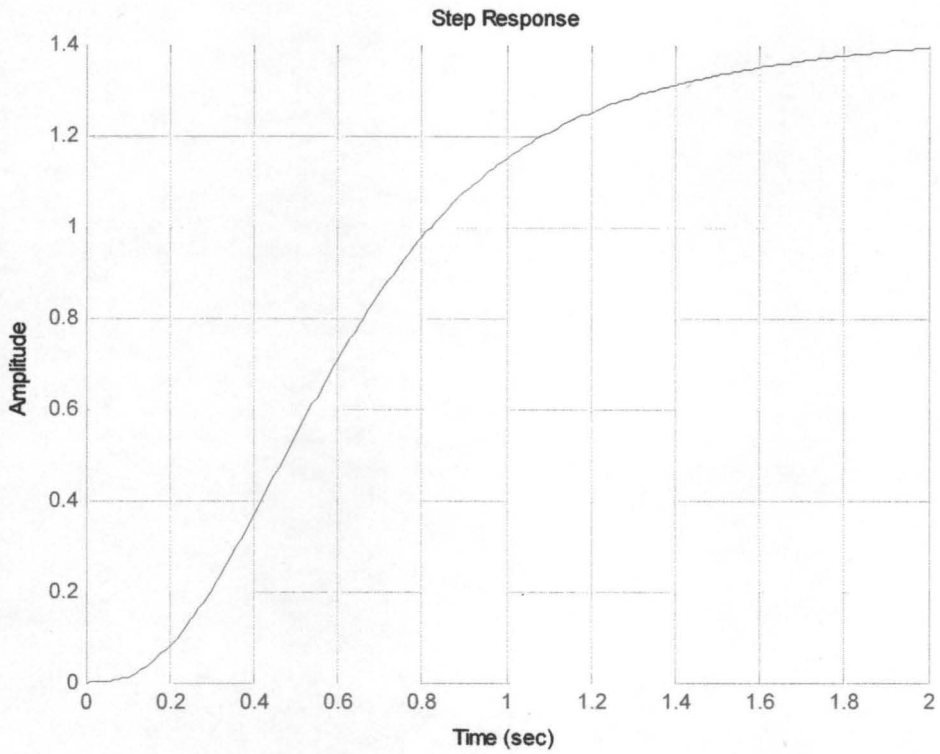


FIG. 3.8 CLOSED-LOOP STEP: $K_p = 30$ $K_I = 70$

We have reduced the proportional gain (Kp) because the integral controller also reduces the rise time and increase the overshoot as the proportional controller does (double effect). The above response shows that the integral controller eliminated the steady-state error.

EXAMPLE 3.4

PROPORTIONAL – INTEGRAL DERIVATIVE CONTROL

$$U(S) = (K_P + \frac{K_I}{S} + Kd^S)\, E(S) \qquad 3.30$$

$$\frac{K_d{}^{S^2} + K_P^S + K_I}{S} E(S) \qquad 3.31$$

$$OLTF = \frac{K_d{}^{S^2} + K_P^S + K_I}{S^3 + S^2 + S} \qquad 3.32$$

$$CLTF = \frac{K_d{}^{S^2} + K_P^S + K_I}{S^3 + (10 + K_d)S^2 + (20 + K_p)S + K_I} \qquad 3.33$$

Now let's take a look at a PID controller the closed-loop transfer function of the given system with a PID controller is

$$\frac{X(S)}{F(S)} = \frac{K_D S^2 + K_p{}^S + K_I}{S^3 + (10 + K_D)S^2 + (20 + K_p)S + K_I} \qquad 3.34$$

After several trial and error runs, the gain Kp = 350, KI = 300, and Kd = 50 provided the desired response. To confirm, enter the following command window will get the following step response.

$K_p = 350$

$K_I = 300$

$K_d = 50$

num [K_d, K_p, K_I]

den [1, 10+K_d, 20+K_p, K_I]

t = 0:0, 01:2

Step (num, den, t)

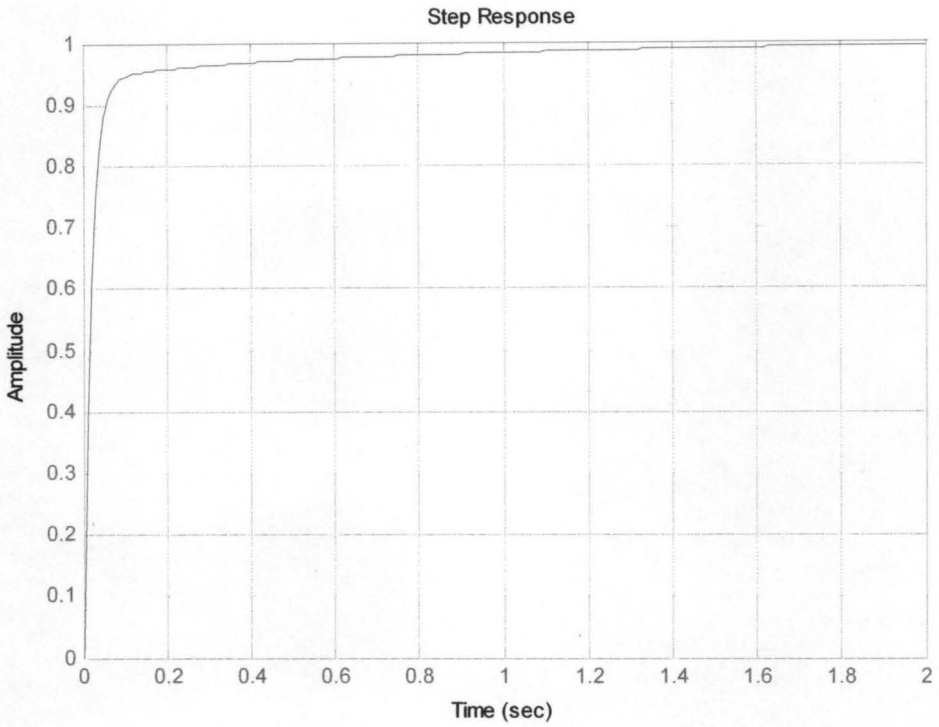


FIG. 3.9 CLOSED – LOOP STEP: $K_p = 350 K_I = 5500$

Now, we have obtained the system with no overshoot t, fast rise time, and no steady-state error.

3.5 THE DYNAMIC OF THE STIRRED TANK MIXER

The problem under consideration is a non – linear stochastic problem which to be precise is stirring tank mixer (CSTM). This has practical application in many chemical, pharmaceutical and petroleum industries as well as in environmental engineering and waste management.

The problem from (Hasdorff, 1976), is illustrated in fig. 3.9 below. There are two input flows (with flow rates F_1 and F_2, concentrations C_1 and C_2 respectively) going in at the top of the mixer. The two inputs are mixed in the tank to produce output with flow rate F_3 and concentration C_3 out at the bottom.

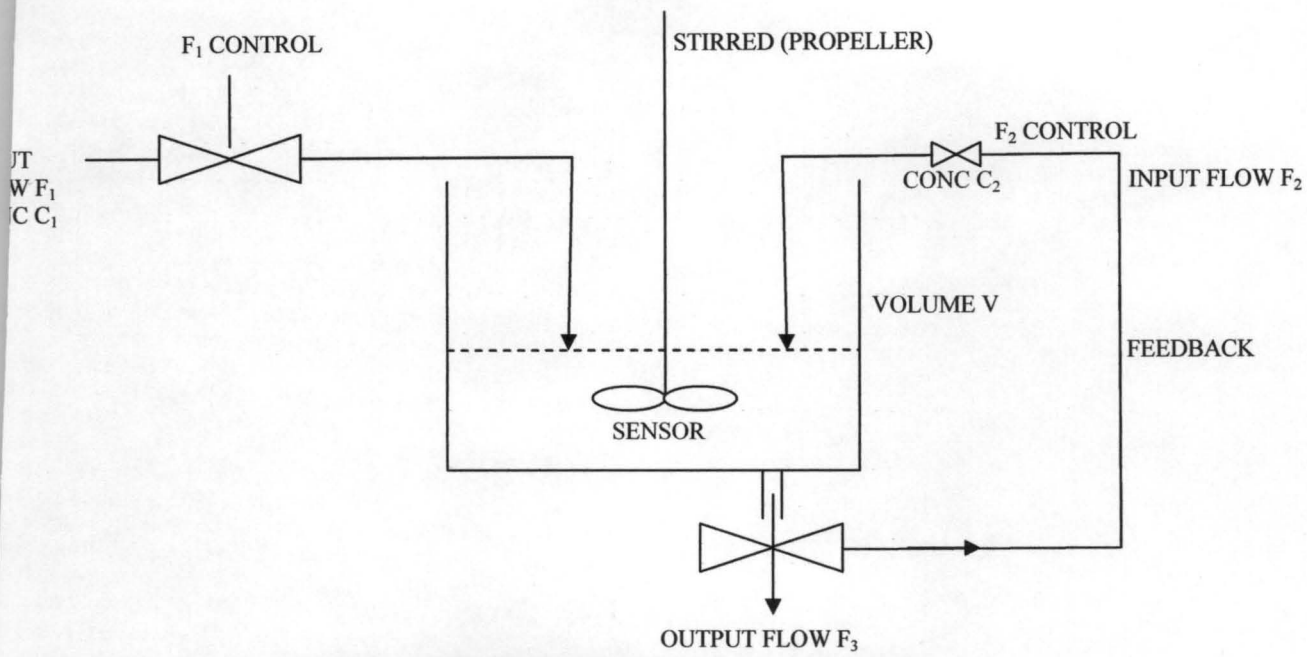


FIG. 3.10: A CONTINUOUS STIRRED TANK MIXER

We see that volume V in the tank changes as the difference between flow in and flow out with a few obvious manipulations.

$$\frac{dC_3}{dt} = \frac{(C_1 - C_3)F_1 + (C_2 - C_3)F_2}{V} \qquad 3.35$$

$$\frac{dV}{dt} = F_1 + F_2 - F_3 \qquad 3.36$$

which is the x = F(x) from desired here.

3.5.1 CHOOSING THE CONTROL VARIABLE AND DESIGN OF THE CONTROLLER

From the tank Fig 3.9 it can be seen that the input flow rate F_1 and F_2 are the controlled variables. Since they can be measured directly and conveniently, moreso that measurement of output concentration C_3 and flow rate F_3 is a related to volume V as in (equation 3.4) the controller is thus to measure C_3 and F_3 and produce control variables F_1 and F_2.

The simplest type of controller that can be though of to do this is a linear controller that measure perturbations of the control variable. Such a controller is shown in the figure below and the control law is given by.

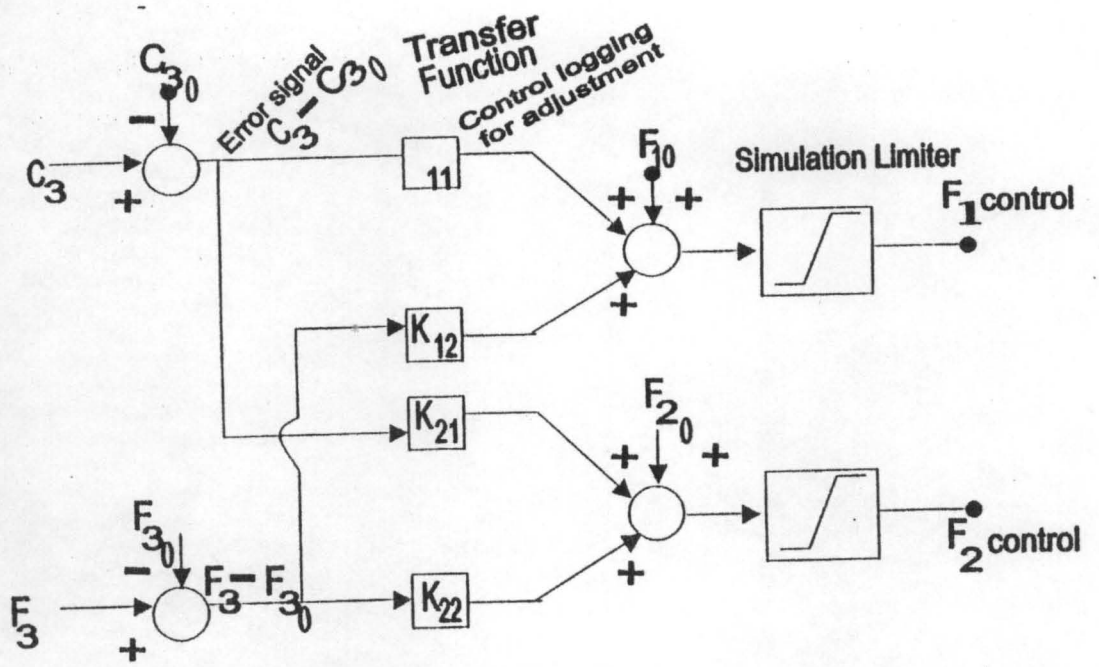


FIG. 3.11 A FEEDBACK CONTROLLER FOR THE CONTINUOUS STIRRED TANK MIXER7

$$F_1 = K_{11}C_3 + K_{12}F_3 - K_{11}C_{30} - K_{12}F_{30} + F_{10} \qquad 3.37a$$

$$F_2 = K_{21}C_3 + K_{22}F_3 - K_{21}C_{30} - K_{22}F_{30} + F_{20} \qquad 3.37b$$

Realistically, the input flow rates can only fall within finite ranges. Here we use the following bounds.

$$0 \le F_1 \le F_1 \max = 2F_{10}$$

$$0 \le F_2 \le F_2 \max = 2F_{20}$$

From the designed controller, the problem is reduced to that of determining the four parameters K_{11}, K_{12}, K_{21} and K_{22}. Now, we choose a cost criterion on the operation of the system and then minimize this cost criterion as

$$J = \int_{t_0}^{tf} [C_3(t) - C_3]^2 dt + R\int_{t_0}^{tf} [V(t) - V_0]^2 dt \qquad 3.38$$

This is simply an integrated square error criterion on the error nominal output concentration C_3 and volume V.

Hence, we can now write a proper formulation for the cost criterion and the system dynamics as (there in equation (3.35), (3.38) with two new variables)

$$\left.\begin{aligned} X_1 &= \frac{(C_1 - X_1)F_1 + (C_2 - X_1)F_2}{X_2} \\ X_2 &= -F_{30}\sqrt{X_2 + F_1 - F_2} \\ X_3 &= (X_1 - C_{30})^2 \\ X_4 &= (X_2 - V_0)^2 \end{aligned}\right\} \quad 3.39$$

Where the state variable assignments are

$$\left.\begin{aligned} X_1 &= C_3 \\ X_2 &= V \end{aligned}\right\} \quad 3.40$$

have been made X_3 and X_4 can be seen to give sum square error in concentration C_3 and volume V. The objective turning 3.38 is then expressed as

$$J = X_3\,(tf) + R\,.\,X_4\,(tf)$$

If $X_3(t_0) = X_4\,(SSt_0) = 0$ is of the desired form

$$J = \emptyset\,(x(tf)).$$

Then the control law using the state variable assignment of 3.40 and control law from 3.36 is

$$F_1 = K_{11}X_1 + K_{12}\,(F_{30}\sqrt{X_2}\,) - K_{11}C_{30} - K_{12}F_{30} + F_{10} \quad 3.41a$$

$$F_2 = K_{21}X_1 + K_{22}\,(F_{30}\sqrt{X_2}\,) - K_{21}C_{30} - K_{22}F_{30} + F_{20} \quad 3.41b$$

With saturation limits as

$$0 \le F_1 \le 2\,F_{10}$$

$$0 \le F_2 \le 2\,F_{20}$$

with the dynamics equation 3.39 and the control law 3.43a and with cost criterion as in 3.41 the problem is in the proper formulation.

3.5.2 SOLUTION BY GRADIENT METHOD

Our cost criterion equation 3.38 is a continuous, quadratic, constrained optimization problem subject to a dynamical set of constraints. The problem could be properly put as

$$\text{Minimise } J = \emptyset(x(t_f)) \quad 3.42$$

Subject to the dynamical equations:

$$\dot{X}_1 = (C_1 - X_1)F_1 + (C_2 - X_1)F_2$$

$$\dot{X}_2 = -F_{30}\sqrt{X_2 + F_1 - F_2}$$

$$\dot{X}_3 = (X_1 - \overline{C}_{3e})^2$$

$$\dot{X}_4 = (X_2 - Ve)^2$$

where X_3 and X_4 are two new state variables and C_{30} and V_o are here replaced with C_{3e} and Ve in other to solve this problem, we first convert the constrained problem into an unconstrained one by introducing penalty constraints and then writing we have

$$\text{Min } J = \int_{t_0}^{t}\left[C_3 - \overline{C}_{3e}\right]^2 dt + R\int_{t_0}^{tf}\left[V - \overline{Ve}\right]^2 dt \qquad 3.43$$

Subject to

$$\dot{X}_1 = (C_1 - X_1)[K_{11}X_1 + K_{12}(0.02\sqrt{x_2}) - K_{11}C_{30} - K_{12}F_{30} + F_{10}]$$

$$+ (C_2 - x_1)[K_{21}x_1 + K_{22}(0.02\sqrt{x_2}) - K_{21}C_{30} - K_{22}F_{30} + F_{20}]/x_2 \qquad 3.44$$

$$\dot{X}_2 = -0.02\sqrt{x_2}\,[K_{11}X_1 + K_{12}(0.02\sqrt{x_2}) - K_{11}C_{30} - K_{12}F_{30} + F_{10}]$$

$$+ [K_{21}x_1 + K_{22}(0.02\sqrt{x_2}) - K_{21}C_{30} - K_{22}F_{30} + F_{20}] \qquad 3.45$$

$$\dot{X}_3 = (x_1 - C_{30})^2 \qquad 3.46$$

$$\dot{X}_4 = (x_2 - V_0)^2 \qquad 3.47$$

We introduced four penalties constant λ_1, λ_2, λ_3 and λ_4 and then write the Hamiltonian form as

$$H = (x_1 - 1.25)^2 + R(x_2 - 1)^2 + \lambda_1[(C_1 - x_1)(K_{11}x_1 + K_{12}(0.02\sqrt{x_2}) - K_{11}C_{30} - K_{12}F_{30}$$

$$+ F_{10}] + (C_2 - x_1)[K_{12}x_1 + K_{22}(0.02\sqrt{x_2}) - K_{21}C_{30} - K_{22}F_{30} + F_{20}]/x_2 + \lambda_2[-$$

$$0.02\sqrt{x_2} + [K_{11}x_1 + K_{12}(0.02\sqrt{x_2}) - K_{11}C_{30} - K_{12}F_{30} + F_{10}] + [K_{12}x_1 + K_{22}(0.02\sqrt{x_2}$$

$$- K_{21}C_{30} - K_{22}F_{30} + F_{20}] + \lambda_3[(x_1 - C_{30})^2] + \lambda_4[(x_2 - V_0)^2] \qquad 3.48$$

The necessary condition for optimality of H is

$$\frac{\partial H}{\partial x_1} = -\dot{\lambda}_1 = 0 \qquad 3.49$$

$$\frac{\partial H}{\partial x_2} = -\dot{\lambda}_2 = 0 \qquad 3.50$$

$$\frac{\partial H}{\partial x_3} = -\dot{\lambda}_3 = 0 \qquad 3.51$$

$$\frac{\partial H}{\partial x_4} = -\dot{\lambda}_4 = 0 \qquad 3.52$$

$$\frac{\partial H}{\partial x_1} = 2(x_1-1.25)+\lambda_1\{C_1K_{11}-2k_{11}X_1-K_{12}(0.02\sqrt{x_1})+K_{11}C_{30}+K_{12}F_{30}-F_{10}+[C_2K_{12}-$$

$$2K_{12}X_1-K_{22}(0.02\sqrt{x_2})+K_{21}C_{30}+K_{22}F_{30}-F_{20}]\}/X_2+\lambda_2(K_{11}+K_{21})+\lambda_3[2(X_1-C_{30})] \qquad 3.53$$

$$\frac{\partial H}{\partial X_2} = 2R(X_2-1)+\lambda_1\{\tfrac{1}{2}[(0.02\sqrt{x_1})+K_{11}C_{30}+K_{12}F_{30}-F_{10}+[C_2K_{12}-2K_{12}X_1-$$

$$K_{22}(0.02\sqrt{x_2})+K_{12}C_{30}+K_{22}F_{30}-F]\}/X_2+\lambda_2(K_{11}+K_{21})+\lambda_3[2(X_1-C_{30})]. \qquad 3.54$$

$$\frac{\partial H}{\partial X_3} = 0 \qquad 3.55$$

$$\frac{\partial H}{\partial X_4} = 0 \qquad 3.56$$

In solving the above set of equations, it is necessary for us to obtain quantifiable numerical throughput values for the respective concentrations and flow rates by chemical laboratory experiment. These would enable us obtain the states and co-states of the above stated simultaneous set of equations. These are solvable using first order partial differential equations principle. With the analytical solutions so obtained, we can do simulations to have visual concepts of the nature of the respective states and co-state functions.

CHAPTER FOUR

THE APPLICATION OF NEURAL NETWORK PREDICTIVE CONTROLLER ON THE STIRRED TANK REACTOR

4.1 SYSTEM IDENTIFICATION

The first stage of model predictive control is to make a neural network to represent the forward dynamics of the plant. The prediction error between the plant output and the neural network output is used as the neural network training signal. The process is represented by the following figure 4.1

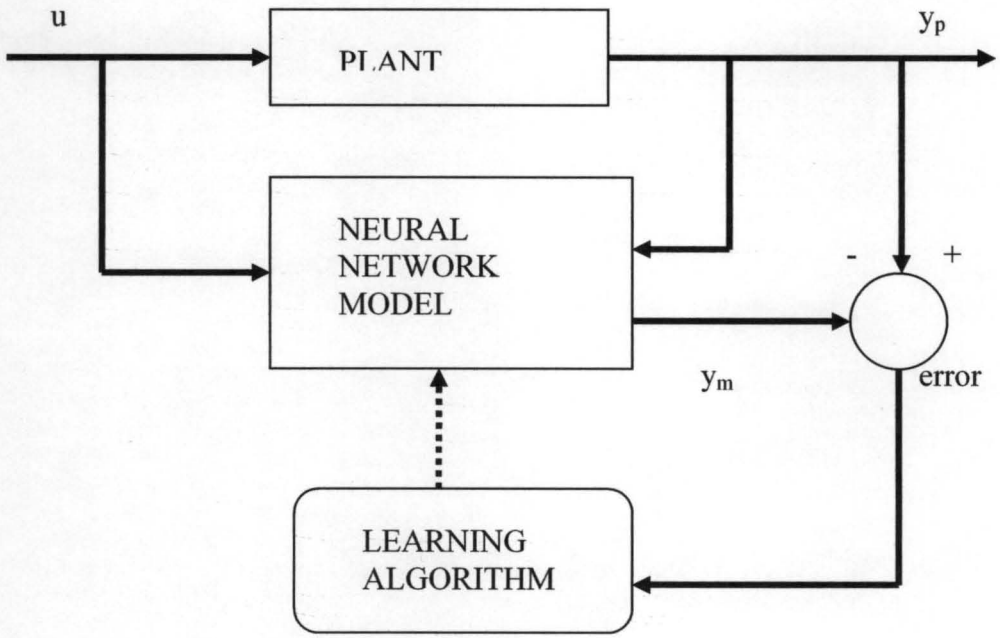


FIGURE 4.1: NEURAL NETWORK PLANT MODEL

The neural network plant model uses previous inputs and previous plant outputs to predict future values of the plant output. The structure of the neural network plant model is given in the following figure 4.2

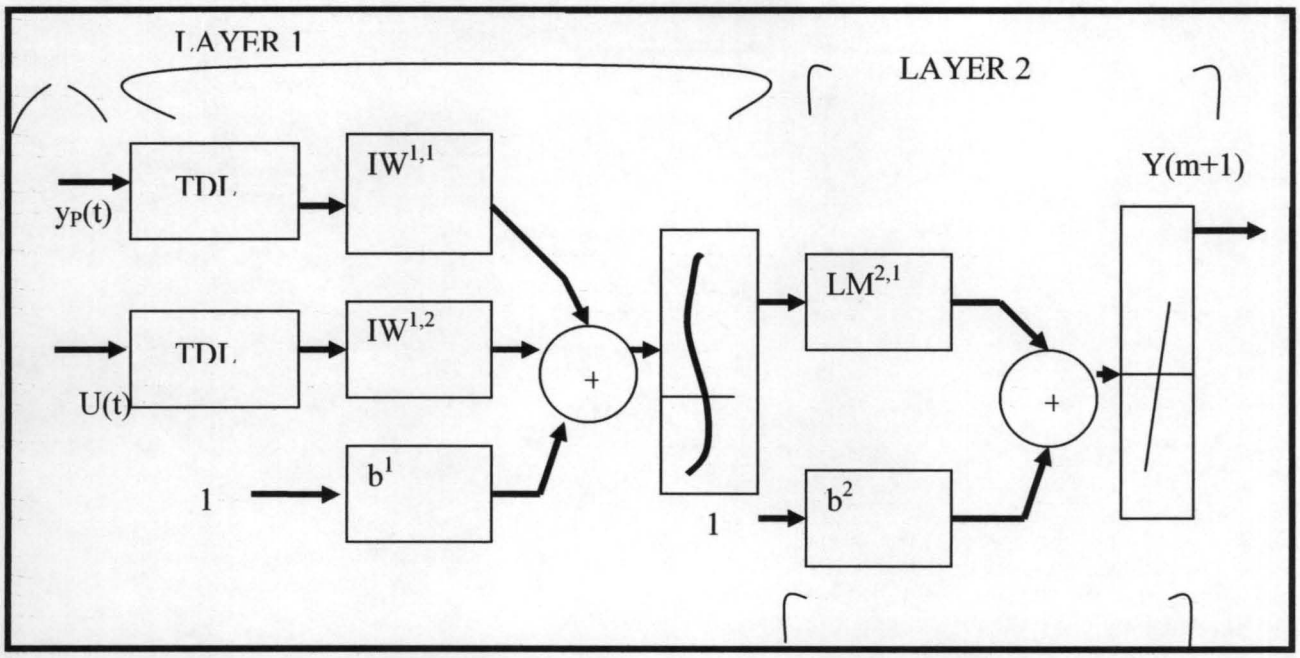


FIGURE 4.2: NEURAL NETWORK

This network can be trained offline in batch mode, using data collected from the operation of the plant. Any of the training algorithms discussed in Back propagation, can be used for network training. This process is discussed in more detail later in this chapter.

4.2 PREDICTIVE CONTROL

The model predictive control method is based on the preceding horizon technique. The neural network model predicts the plant response over a specified time horizon. The predictions are used by a numerical optimization program to determine the control signal that minimizes the following performance criterion over the specified horizon

$$Minimize\ J = \sum_{N_1}^{N_2}(y_r(t+j) - y_m(t+j))^2 + \rho\sum_{j=1}^{N_u}(u'(t+j-1) - u'(t+j-2) \qquad 4.1$$

where N_2, N_1 and N_u define the horizons over which the tracking error and the control increments are evaluated. The u' variable is the tentative control signal, y_r is the desired response and y_m is the network model response. The ρ value determines the contribution that the sum of the squares of the control increments has on the performance index.

The following block diagram illustrates the model predictive control process. The controller consists of the neural network plant model and the optimization block. The optimization block determines the values of u' that minimize J, and then the optimal u is input to the plant. The controller block has been implemented in Simulink, as described in the following section.

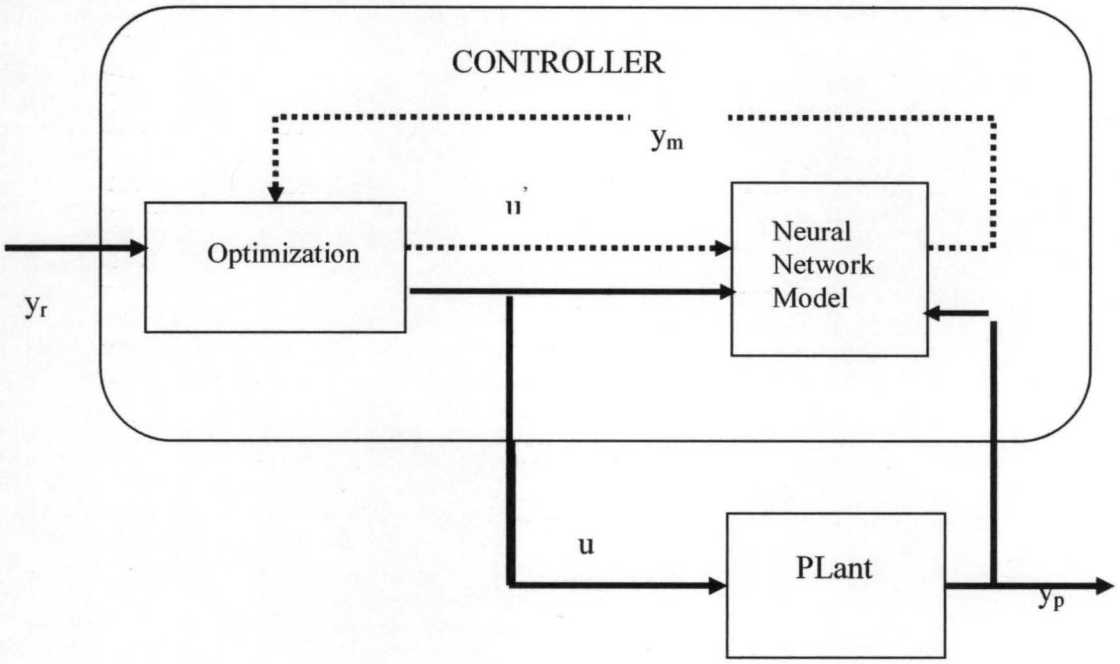


FIGURE 4.3: NEURAL NETWORK PREDICTIVE SYSTEM CONTROLLER BLOCK

4.3 USING THE NN PREDICTIVE CONTROLLER BLOCK

This section demonstrates how the NN Predictive Controller block is used. The first step is to copy the NN Predictive Controller block from the Neural Network Toolbox blockset to your model window. This demo uses a catalytic Continuous Stirred Tank Reactor (CSTR). A diagram of the process is shown in the following figure4.4.

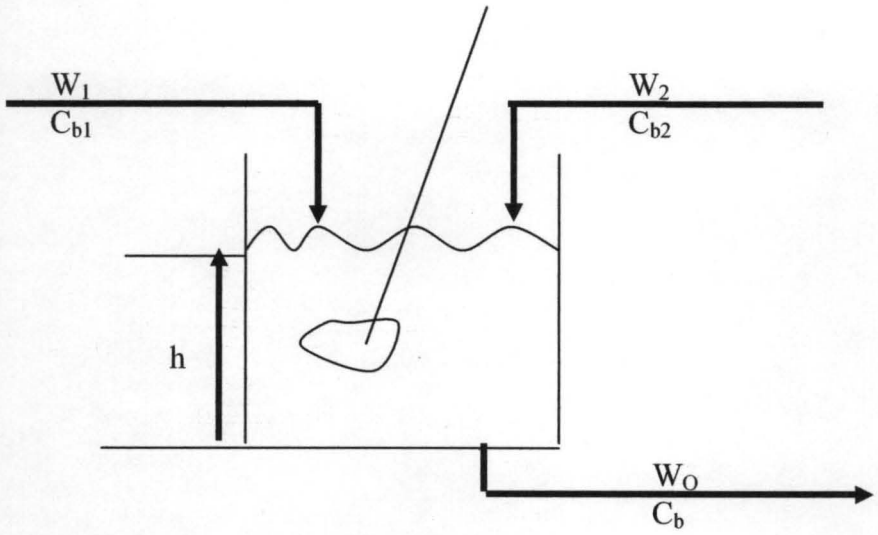


FIG 4.4: A STIRRING TANK REACTOR

The dynamic model of the system is:

$$\frac{dh(t)}{dt} = w_1(t) + w_2(t) - 0.2\sqrt{h(t)} \qquad 4.2$$

$$\frac{dC_b(t)}{dt} = (C_{b1}(t) - C_{b2}(t)\frac{w_1(t)}{h(t)} + (C_{b2} - C_b(t))\frac{w_2(t)}{h(t)} - \frac{k_1 C_b(t)}{(1+k_2 C_b(t))^2} \qquad 4.3$$

where $h(t)$ is the liquid level, $C_b(t)$ is the product concentration at the output of the process, $w_1(t)$ is the flow rate of the concentrated feed C_{b1} , and $w_2(t)$ is the flow rate of the diluted feed C_{b2} . The input concentrations are set to $C_{b1} = 24.9$ and $C_{b2} = 0.1$. The constants associated with the rate of consumption are $k_1 = 1$ and $k_2 = 1$.

The objective of the controller is to maintain the product concentration by adjusting the flow $w_2(t)$. To simplify the demonstration, we set $w_1(t) = 0.1$. The level of the tank $h(t)$ is not controlled for this experiment. To run this demo, follow these steps

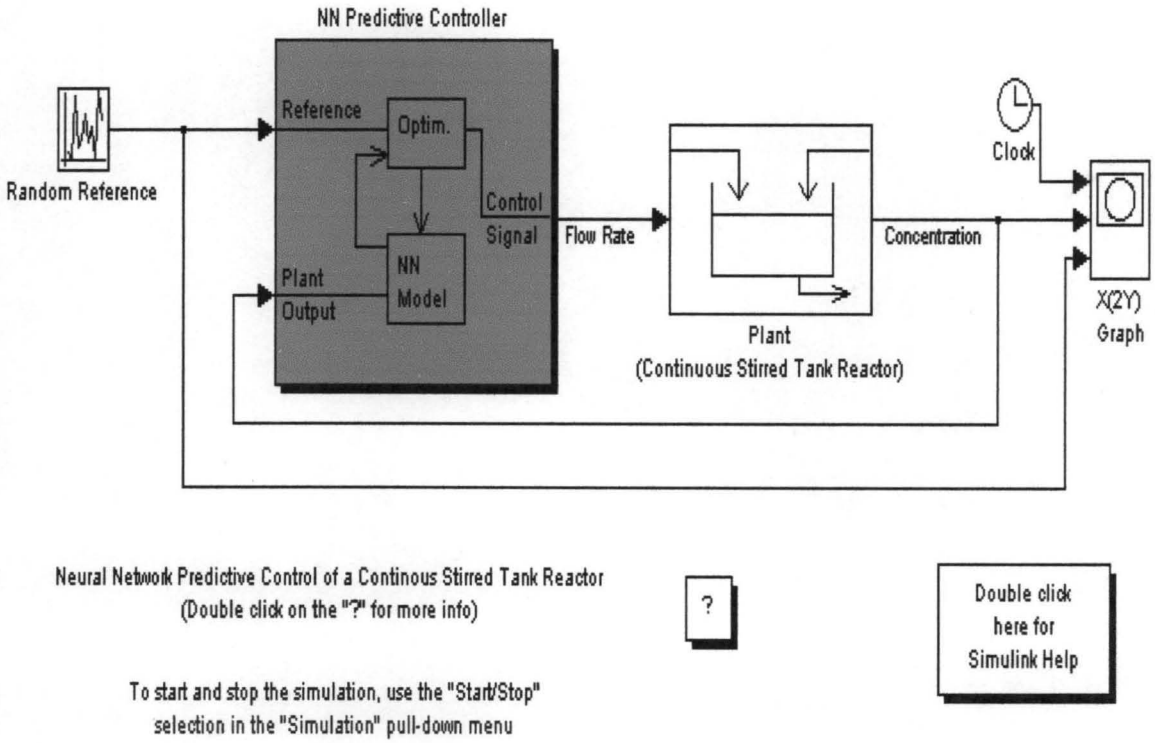
Example 4.1 (Running the Predictive Controller)

To run this demo, follow these steps.

(1) Start MATLAB.

(2) Run the demo model by typing **predcstr** in the MATLAB® command window. This command starts Simulink and creates the following model

window. The NN Predictive Controller block has already been placed in the model.



The continuous stirred reactor tank has the following embedded dynamic transfer diagram

FIGURE 4.5: PREDICTIVE CONTROLLER BLOCK

3 Double-click the NN Predictive Controller block. This brings up the following window for designing the model predictive controller. This window enables us to change the controller horizons N_2 and N_u. (N_1 is fixed at 1.) The weighting parameter ρ, described earlier, is also defined in this window. The parameter α is used to control the optimization. It determines how much reduction in performance is required for a successful optimization step. One can select which linear minimization routine is used by the optimization algorithm, and decide how many iterations of the optimization algorithm are performed at each sample time. The linear minimization routines are slight modifications of those discussed in Backpropagation

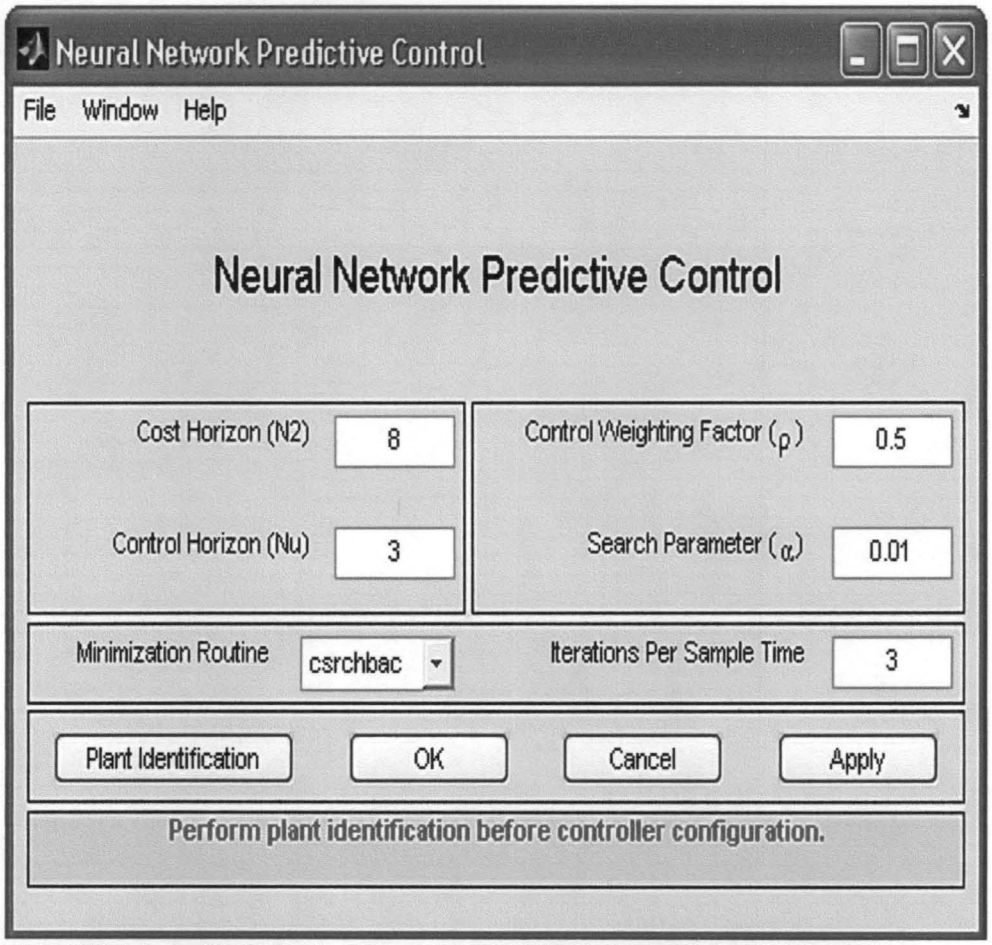


FIGURE 4.6: NEURAL NETWORK PREDICTIVE CONTROL

4 Select Plant Identification. This opens the following window. The neural network plant model must be developed before the controller is used. The plant model predicts future plant outputs. The optimization algorithm uses these predictions to determine the control inputs that optimize future performance. The plant model neural network has one hidden layer, as shown earlier. The size of that layer, the number of delayed inputs and delayed outputs, and the training function are selected in this window. One can select any of the training functions described in Backpropagation, to train the neural network plant model.

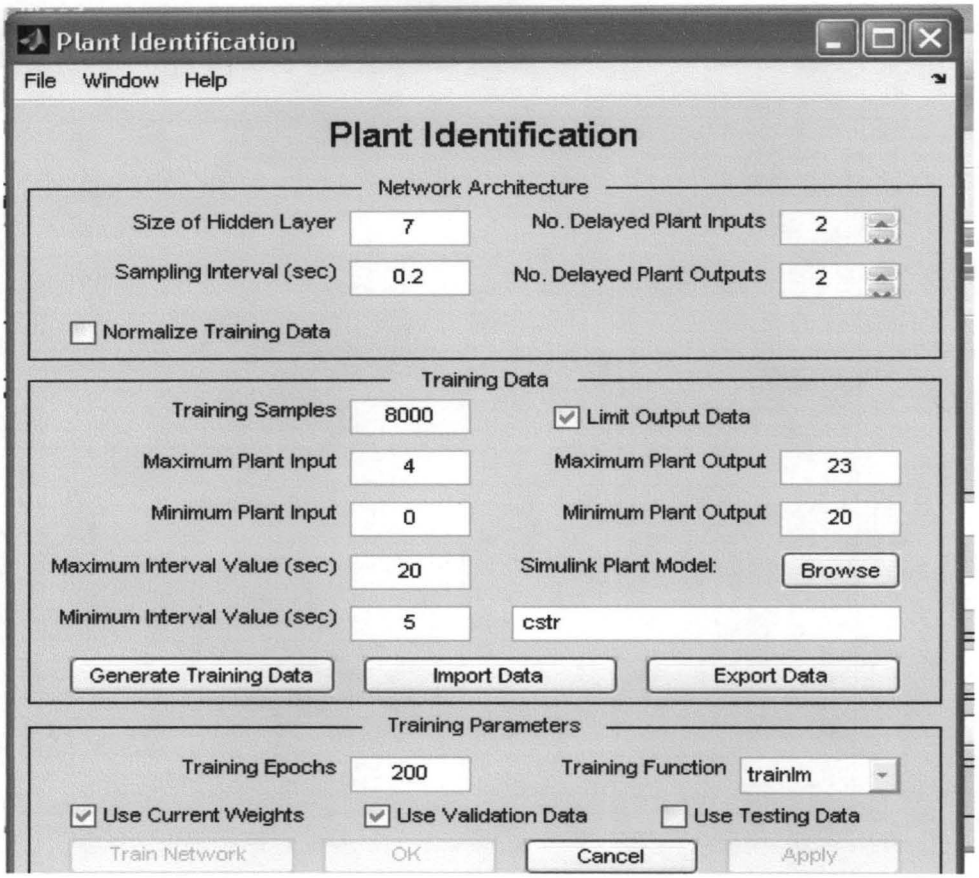


FIGURE 4.7: PLANT IDENTIFICATION

5 Select the Generate Training Data button. The program generates training data by applying a series of random step inputs to the Simulink plant model. The potential training data is then displayed in a figure similar to the following.

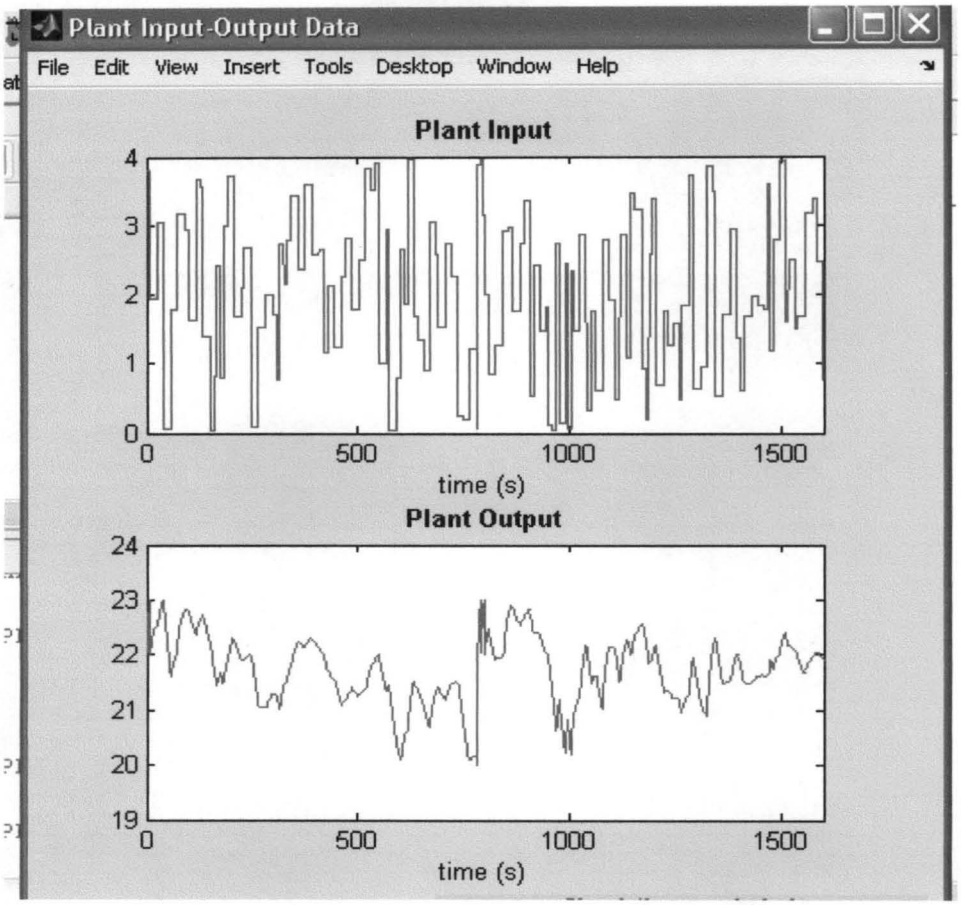


FIGURE 4.8: PLANT INPUT AND OUTPUT DATA

6 Select Accept Data, and then select Train Network from the Plant Identification window. Plant model training begins. The training proceeds according to the selected training algorithm (trainlm in this case). This is a straightforward application of batch training, as described in Backpropagation. After the training is complete, the response of the resulting plant model is displayed, as in the following figure. (There are also separate plots for validation and testing data, if they exist.) One can then continue training with the same data set by selecting Train Network again, erase Generated Data and generate a new data set, or accept the current plant model and begin simulating the closed loop system. For this demonstration, begin the simulation, as shown in the following steps.

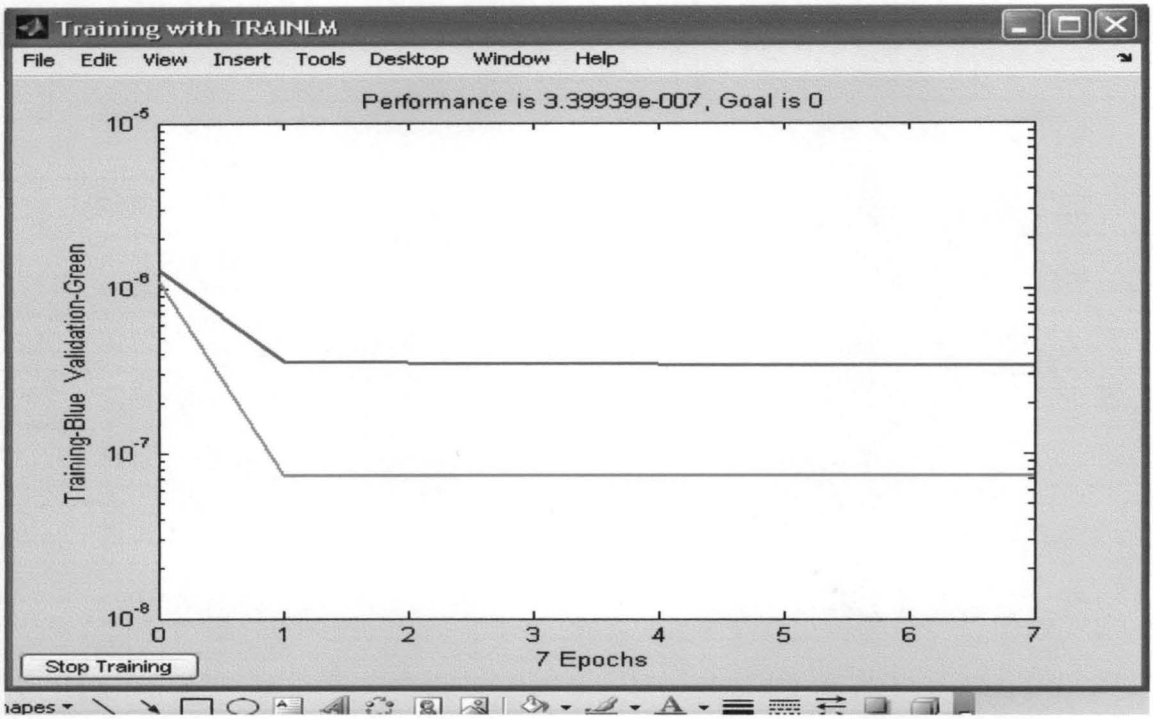


FIGURE 4.9: TRAINNING WITH TAINLM

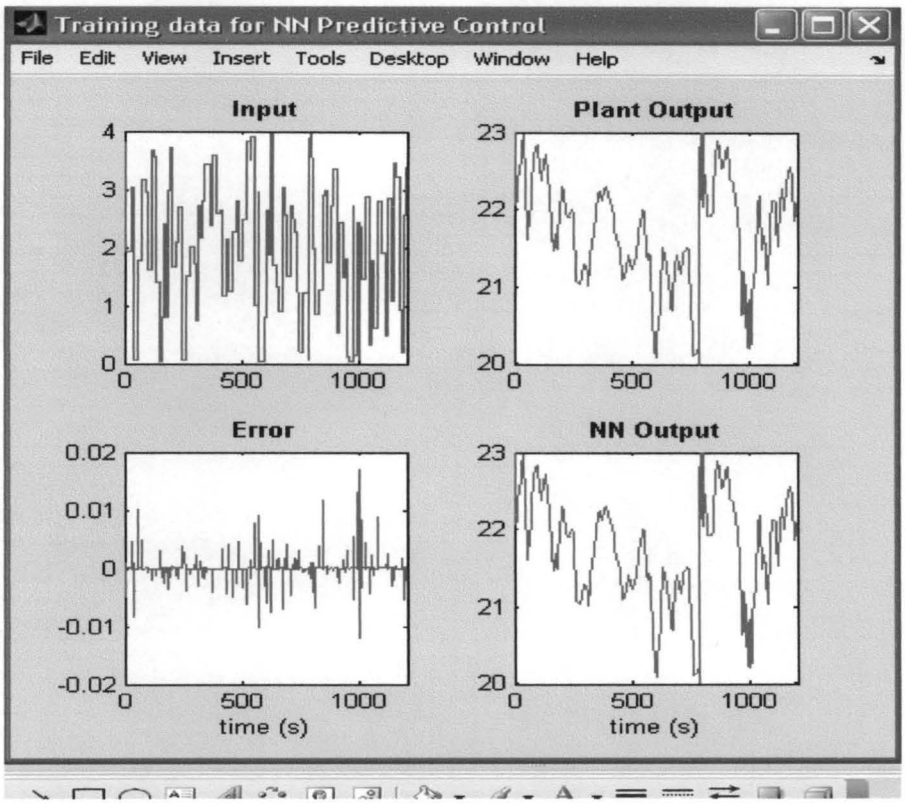


FIGURE 4.10: TRAINNING DATA FOR NN PREDICTIVE CONTROL

7 Select OK in the Plant Identification window. This loads the trained neural network plant model into the NN Predictive Controller block.

8 Select OK in the Neural Network Predictive Control window. This loads the controller parameters into the NN Predictive Controller block.

9 Return to the Simulink model and start the simulation by choosing the Start command from the Simulation menu. As the simulation runs, the plant output and the reference signal are displayed, as in the following figure.

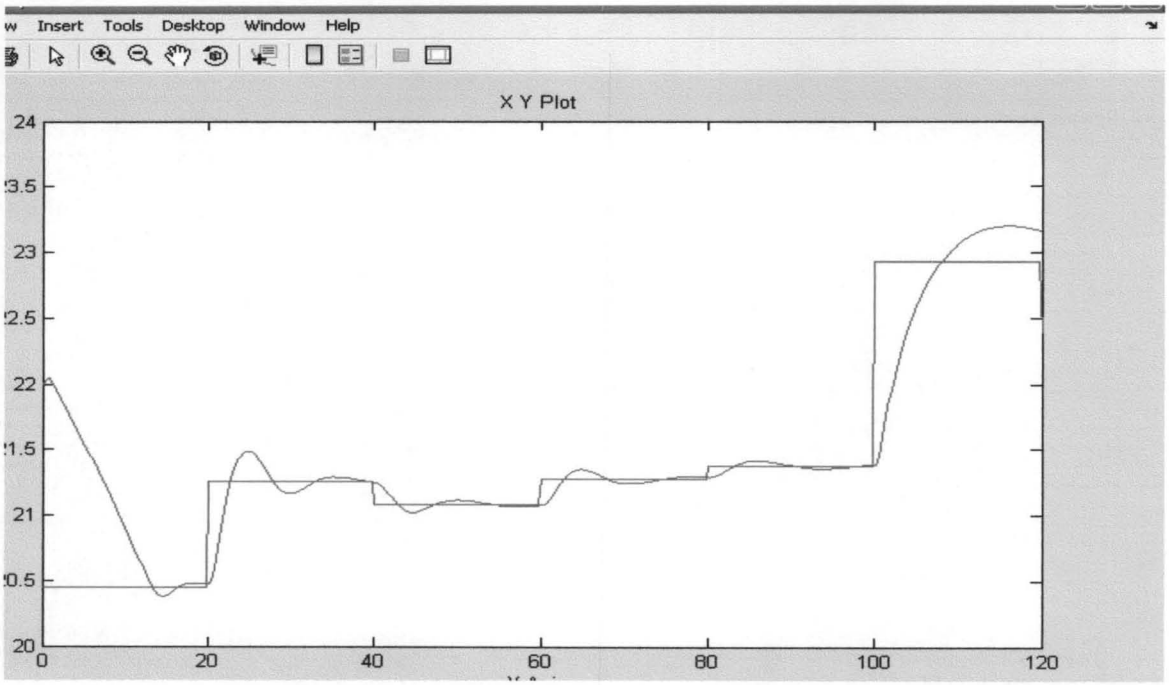


FIGURE 4.10: THE PLANT OUTPUT AND REFERENCE SIGNAL DISPLAYED

Example 4.2 (NN Predictive Controller)

In a sequential procedure as in example 4.1, we carry on the process of simulating for the Plant output and reference signal in this order. This second example is obtained by changing the inputs data as shown in the figures below:

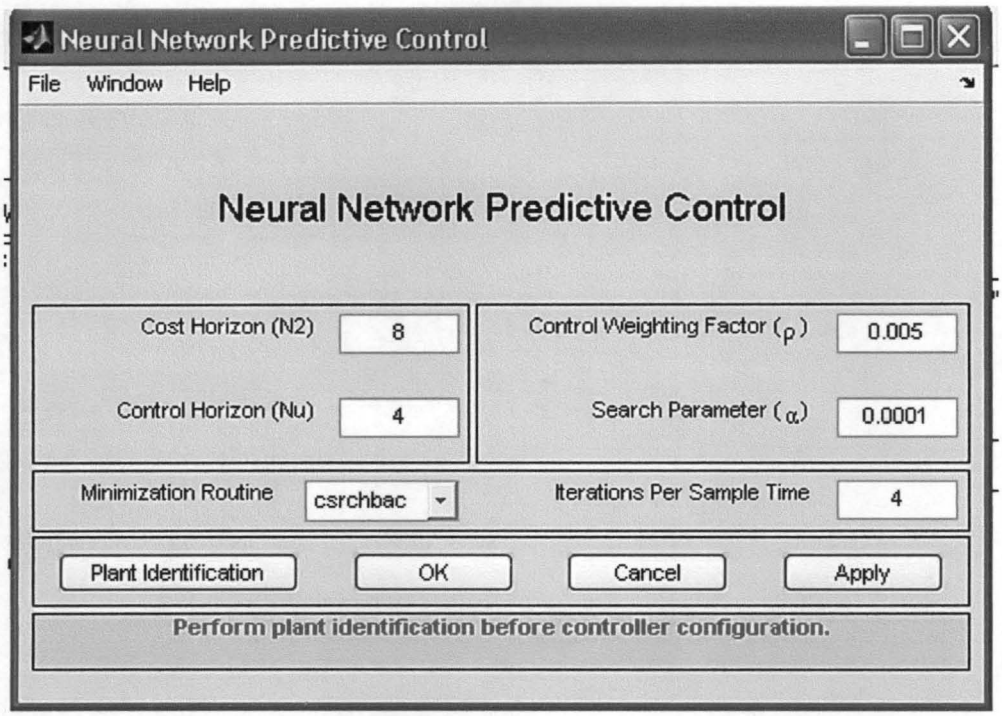


FIGURE 4.11: NEURAL NETWORK PREDICTIVE CONTROL

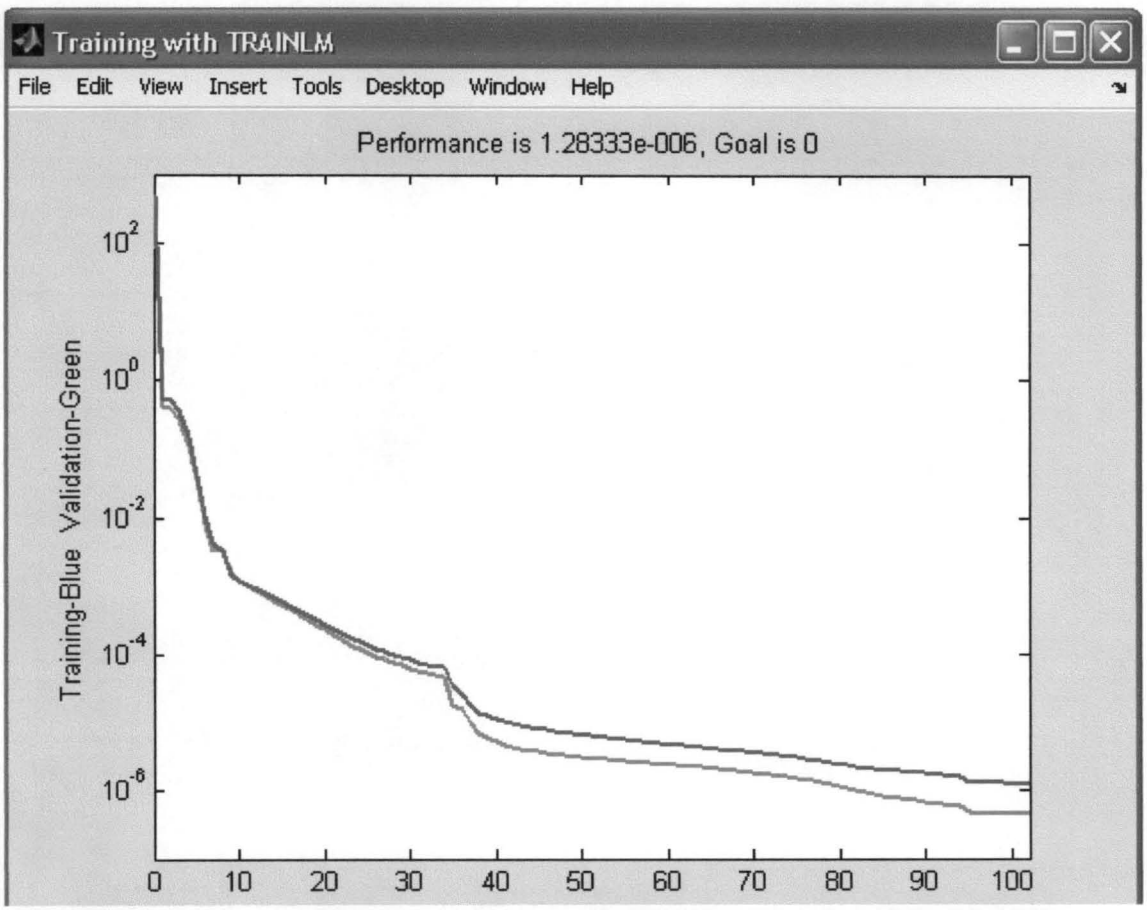


FIGURE 4.12: TRAINING THE IDENTIFICATION CONTROLLER

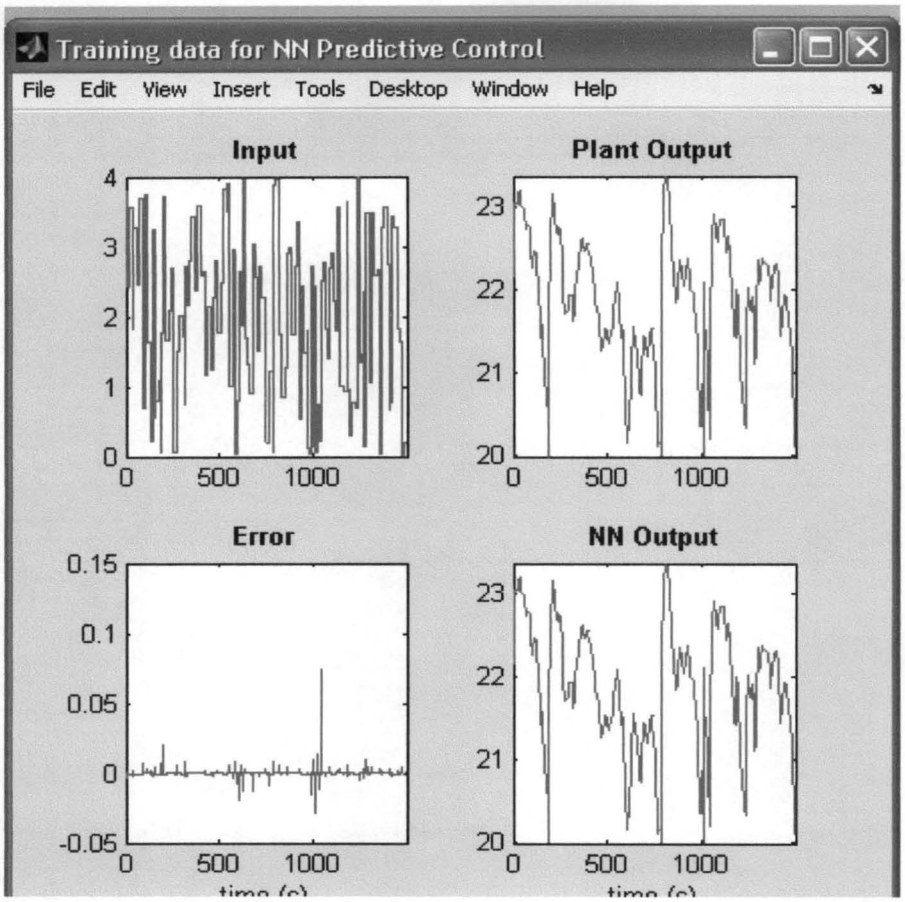


FIGURE 4.13: TRAINING DATA INPUTS AND OUTPUT PLANT, ERROR AND NN OUTPUT SIGNALS

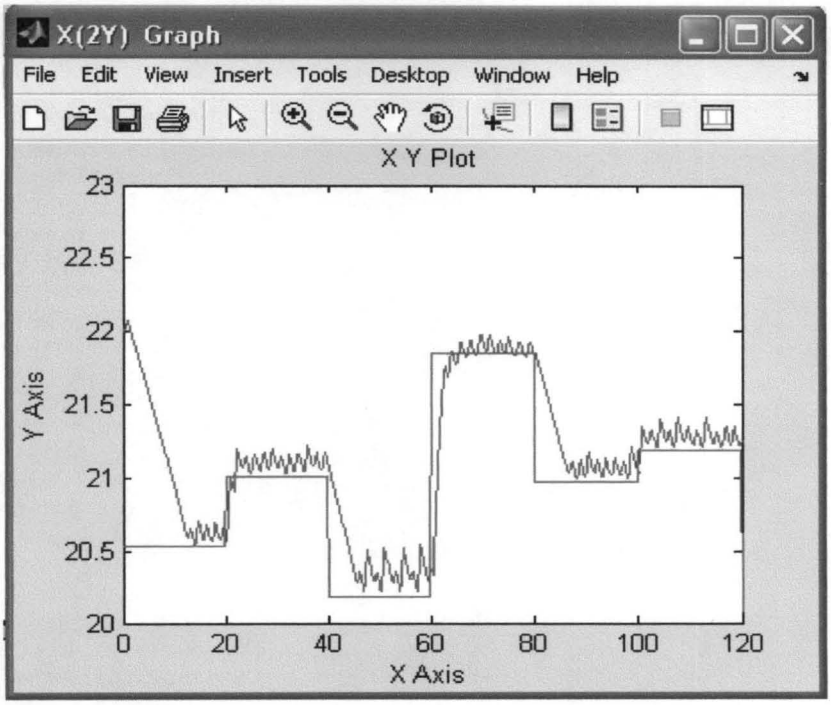


FIGURE4.14: THE PLANT OUTPUT AND REFERENCE SIGNAL DISPLAYED

APPENDIX A

The programmes generated by the neural network system for the outputs

```
>> predcstr
TRAINLM, Epoch 0/200, MSE 1.25774e-006/0, Gradient 82.5739/1e-010
TRAINLM, Epoch 1/200, MSE 2.98939e-007/0, Gradient 3.53103/1e-010
TRAINLM, Epoch 2/200, MSE 2.94471e-007/0, Gradient 30.1914/1e-010
TRAINLM, Epoch 3/200, MSE 2.90718e-007/0, Gradient 0.0710876/1e-010
TRAINLM, Epoch 4/200, MSE 2.90207e-007/0, Gradient 0.0107947/1e-010
TRAINLM, Epoch 5/200, MSE 2.89765e-007/0, Gradient 0.0361354/1e-010
TRAINLM, Epoch 6/200, MSE 2.89236e-007/0, Gradient 0.0691151/1e-010
TRAINLM, Epoch 7/200, MSE 2.88734e-007/0, Gradient 3.91579/1e-010
TRAINLM, Validation stop.

??? Invalid handle object.

Error in ==> nncontrol\private\nnident at 1583
      f2=get(fig2,'userdata');

Error in ==> nncontrolutil at 20
  feval(command,varargin{:});

??? Error while evaluating uicontrol Callback.
```

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 SUMMARY AND CONCLUSION

We have successfully modeled the mechanism governing the functionality of the brain. To achieve this goal, biological neurons were described in sufficient detail to permit meaningful modeling. The brain has been viewed as a network of billons of simple computing devices, each operating on its own set of inputs and transmitting its output to specific sets of other computing devices in the network.

The highly complex interconnections of neurons produce the computation power of the brain. Artificial neural network mimicking the computation richness of the brain implanting our hybrid. Proportional Integral Derivative Algorithm was successfully used to provide solution (maintaining) the volume of the mixture in the tank, checking unnecessarily high inlet flow rate and ensuing the stability of the system for the continuous stirred tank reactor (CSTR) problem, which is nonlinear and dynamic in nature. The results of this work strengthen the fact that neural networks can be used for fault detection and diagnosis purposes. This work also shows that this Proportional Integral Derivative (PID) controller provides an alternative attractive method for solving the Continuous Stirred Tank Reactor (CSTR) problems.

In this research, we were able to

(I) come up with a proportional integral derivative controller which simulates and provides an alternative solution to the continuous stirred tank reactor problem.

(II) provide a model system that can be easily extended to solve multiple fault detection in system with non linear dynamics like the continuous stirred tank reactor.

(III) design a very stable transfer functions (controller) that was used to determine the behaviors of the control variables in the problem.

(IV) established that the Continuous Stirred Tank Reactor problem is a type of optimal control problem and hence a cost function was formulated for the system in terms of the parameters of interest (i.e concentration

C and volume V). We can therefore say that artificial neural networks can be successfully used to solve problems, process control and fault detection and diagnosis problems. The controller under consideration follows a second order reaction as a result; this controller system may be able to handle problems of higher order.

5.2 RECOMMENDATIONS FOR FURTHER RESEARCH INVESTIGATION

Since neural network model offers very promising approach to building truly Intelligent Systems which can provide good optimal solutions for the control problems, we hereby make the following recommendations.

(I) Research candidates should be made to explore the full potential of this new computing technique to solve other industrial problems.

(II) Further research should be directed towards Continuous Stirred Tanks Reactor problem whose reaction is of higher order.

REFERENCES

Aarts. E and Korst J. (1989); Simulated Annealing and Boltmann Machines: A Storhastic Approach to combinatorial Optimization and Neural Computing, Wiley, New York.

Aldrich C. Moolman D.Ward Van Deventer J.S (1995); monitoring and control of Hydrometallugical process with self-organizing Adaptive Neural Net Sysstems, comput. Chem.. Engng Vol 19, PP S803-S808.

Almeida L.B (1987) A learning Rule for Asynchronous Perceptrons with Feedback in a combinatrional environment. Proceedings of the First IEEE intern. Conf. On Neural Networks San Diego Vol. 2, PP 609-18

Alvager T.,T.J. Smith, and F. Vijia; (1994); The use of Artificial Neural Networks in Biomedical Technologies: An introduction; Biomedical instrumentation and technology, Vol. 28, No. 4 PP 315-322.

Anderson B.D.O. and Moore J.B. (1979); Optimal filtering, prentice-Hall, Engle wood Cliffs, New jersey.

Angeniol B; Dela Croix Vaubois G. and Le-Texier J.Y (1988); Self-organizing feature Maps and the traveling Salesman problem. Neural Networks, Vol. 1, PP 289-93.

Balakrishnan. S.N. and R.D Weil, (1996); "Neuro control: A literature suritey". Mathematical and computer modeling, Vol 23, No. 1-2, PP 101-117.

Bateson R.N. (1993); Introduction to control system technology, 4th edition Macmillan publishing coy. PP 33-38.

Bose N.K. and Liang P. (1996); Neural network Fundamentals with Graph, Algorithms and Applications; McGraw- Hill, Inc New York.

Branko S. and the IRIS Group: (1991); Neural and intelligent System integration; John Willy & Son Inc

Broom head D.S. and Lowe D. (1988); Multivariable functional interpolation and Adaptive Neural Networks. Complex Systems 2:321-355.

Brositow C. and Tong M. (1978); The structure and dynamics of inferential control systems ALCHET.J 24, 429.

Cain B.J (1990) An improved probabilistic Neural Network and its performance relative to other models. In Application of Artificial Neural Networks Proc. SPIE Vol. 1294, PP 354-365

Carpenter G. and Rosen D.B. (1992); Fuzzy ART; Fast stable learning and categorization of Analog input pattern by an Adaptive resonance System, Neural Networks Vol. 4, PP 759-71.

Carpenter G.A Grossberg & (1987). A massively parallel architecture for a self- organizing neural pattern recognition machine, computer vision graphics and image processing, Vol. 37, PP 54-55

Carpenter G.A and Grossberg S. (1988); The Art of Adaptive Pattern recognition by self-organization Neural Networks. IEEE computer, PP 77-88

Castellano E.N, Mc Cain C.A and Noble F.N (1978); Digital control of a Distillation system, chem., Engr Prog. 74(4), 56.

Cellander, A.D.R. Hartree and A. porter (1936); Time Lag in a control system Phi. Trans. Roy. Soc. London, Ser. A, 235: 415

Chua L.O and Yang L. (1988); Cellular Neural networks: Theory. IEEE Trans. Circuit and Systems, 35, 1257-1272.

Chua L.O; Roska T. and Venetianer P.L (1993); The CNN is Universal as the Turing machine. IEEE Trans. Circuit and systems 1-fundermental theory and Application 40, 289-291

Cohen M.A and Grossberg S. (1983); Absolute Stability of global pattern formation and Parallel memory Storage by Competitive Neural Networks IEEE trans. System; Man and Cybernetics 13, 815-826.

Cooper. B.S. (1994); "Selected Application of neural Networks in telecommunication systems" Australian Telecommunication research; Vol 28, No. 2.PP.9-20.

Croall, I. F. and Mason J.P., (1991); Industrial Applications of Neural Networks in project ANNIE Handbook, Springer- Verlag, Berlin.

Cybenko. G (1989); Approximation by superposition of a sigmoidal function. Math. Control Signals system, Vol. 2, PP 33-314.

Daniels J.W. (1971); The Approximate Minimization of Functional. Prentice-hall.

Elman J.L. (1991); Distributed Representations Simple Recurrent Networks and Grammatical Structure, machine learning, Vol. 7.PP. 750-6.

Fukushima, K. (1988); A Neural Network for Visula pattern recognition. IEEE. Computer Vol. 21, No. 3.PP. 65-75

Fukushima k. and Myaki, (1982); Neocognition: A New algorithm for pattern Recognition tolerant of deformation and Shift in position. Pattern recognition, Vol. 15, No. 6PP. 455-60

Fukushima, K and Wake N; (1991); Handwritten Alphabetic character. Recognition by the Neocognition, IEEE transaction on Neural Network Vol. 2, No.3, PP.355-65.

Fukushima K. (1989); Analysis of the process of visual pattern Recognition by the Neocognitron and Neural Networks Vol. 2, PP 413-420

Garry Y. (1987); BYTES; Neural Network Heuristics. PP 183-191

Giles C.L and Maxwell T. (1987); Learning, invariance and Generalization in High- Order Neural Networks. Applied optics. Vol.26,No.23, P4972

Goita K. et al; (1994); Literature Review of Artificial Neural Networks and Knowledge based systems for image analysis and interpretation of data in remote sensing Canadian journal of Electrical and computer Engineering, Vol. 19; No. 2, PP53-61

Grebe J.J; R.H. Boundy and R.W. Cermark (1993); The control of chemical processes, Trans. Am inst chem Engrs; 29:211

Grossberg S. (1968); Embedding Fields. A theory of learning with physiological implications. Journal of mathematical psychology, Vol. 6, PP 209-39.

Grob-Hardt, R. and Laux. T. (2003). Stem cell regulation in the shoot meristem. Journal of cell science 116(9), 1659-1666

Hasdorf Lawrence (1976); Gradient Optimization and Non-Linear control. John wiley & sons, New York. PP45-55,219-225.

Hasnain, S.K. (2001); Neural Network and its application in biomedical engineering III[rd] Annual Hamdard Symposium of Hamdard college of medicine and Dentistry & Hamdard University hospital, Karachi.

Haykins S. (1994); Neural Network-A comprehensive foundation, IEEE Press Macmillan, New York.

Hecth-Nielson R. (1987); Kolmogorov's Mapping Neural Network Existence Theorem, Preceding of the first IEEE Int conf on Neural Network Vol. 3 PP.112-4.

Hecht-Nielson R. (1987); Nearest Matched filter classification of spatiotemporal. Patterns. Applied optics 26. 1892-1899.

Himmelblau D.M. (1978); Fault detection and diagnosis in chemical and petrochemical processes Elsevier.

Hinton G.E. (1989); Connectionist Learning Procedure Artificial Intelligence Vol. 40 pp 185-234

Hinton G.E and Sejnowsk T.J. (1983); Analyzing Cooperative Computation. Proceeding of the fifth Annual Conference of the Cognitive Science Society, Rochester NY, PP448-53

Hopfield J.J. (1982); Neural Network and physical Systems with Emergent Collective computational Abilities, proceedings of the National Academy of Science vol.179 pp 2554-8

Hopfield J.J. and Tank D.N. (1985); Neural computation of Decisions in Optimization Problems. Biological cybernetics Vol. 52, PP 141-52.

Horng T., Wang C. and Alexopoulos N.G. (1993); Microstrip Design Using Neural Networks MTT-S Int. Microwave Symp. Dig, pp413-416.

Hornik K, Stinchombe M., and White H. (1989); Multilayer feed forward Network are universal approximators. Neural Networks, 2:359-366

Http:/www. Emsl.pnl.gov:2080/docs/cie/neural.homepage.html Ivanoff A. (1934); Theoretical Foundations of the Automatic Regulation of Temperature J. Inst Fuel, 7:117

Jabocs R.A. (1988); Increased Rates of Convergence Through Learning Rate Adaptation, Neural Networks Vol.1 No. 4, PP1-2,17-25.

Kanade, T., Reed M.L. and Weiss L.E, (1994); New Technologies and Applications in Robotics, Communication of the ACM, Vol. 1.37 PP.58-67.

Kangas J.A. Kohonen T.K. and Laaksonen J.T. (1990); Variants of self-organizing maps, IEEE Transaction on Neural Networks 1(1): 93-99.

Kestenbaum A., Shinnar R. and Thau F.E. (1976); Design Concepts for process Control, Ind Eng. Chem., Proc. Des. Dev, 15,2.

Kohonen T (1988a); The neural Phonetic Typewriter, IEEE Computer, 21:11-22.

Kohonen T. (1984b); Self Organization and Associative Memory First Edition, Springer-Verleg Berlin-Heidelberg

Kohonen T (1989c); Self Organization and Associative Memory Third Edition, Springer-Verleg Berlin-Heideiberg

Lconard J.A. and Kramer M.A. (1991); Radial Basis Function Networks for classifying process faults. IEEE Control System 11, 31-38.

Luenberger D.G (1969) Optimization by Vector Space Method, Wiley

Luyben W.L. (1973); Process Modeling, Simulation, and Control For Chemical Engineers. McGraw-Hill Kogakusha, Ltd.

McAvoy T.J., Hsu E and Lowenthal S. (1972); Dynamics of pH in Controlled Stirred Tank Reactor, Ind. Eng. Chem. Process Des. Develop 11:1,68-70

Matlab Neural Network Toolbox Version. 5.2

Microsoft Encarta Encyclopedia (2003).

Moody J. and Darken C.J (1989); Fast Learning, in Networks of Locally-Tuned processing Units. Neural computation 1:281-294.

Muhammad A. (2001); Computerized patient Monitoring System for ICU patient care, Final year project, Sir Syed University of Engineering and Technology, Karachi, Pakistan.

Parker D.B. (1985); Learning Logic Technical Report TR-47, Centre for Computational Research in Economics and Management Sciences, MIT

Pearlmutter B.A. (1988); Dynamic Recurrent Neural Networks Reports CMU-CS88-191 School of Computer Science Carnegie-Mellon University.

Peterson K.L (1992); Counter propagation neural networks in the modeling and prediction of Kovats indices for substituted phenols. Anal. Chem. 62,379-386.

Pellionisz A.J. (1990); About the Geometry Intrinsic to neural nets. Proc. Internat. Joint Conference on Neural Networks 15.

Pelionisz A.J. (1990); The Geometry of Brain functions, Tenzor Network Theory, Cambridge University Press

Peter Harriot. (1981); Process Control. Tata McGraw-Hill Publishing Company Ltd.

Perry R.H and D.W. Green (1997); Chemical Engineering Handbook 7th edition McGraw Hill International Edition, Singapore. 8:4-5

Pineda F.J (1987); Generalization of Back- Propagation to Re current Neural Networks phys Rev Letters 59:2229-2232

Pineda F.J. (1988); Dynamics and Architecture for Neural Computation J. Complexity 4:216-240

Pineda F.J. (1989), Recurrent Back-propagation and the Dynamic Approach to Adaptive Neural Computation. Neural Computation; 1:161-172

Reddy D.C., Ghosh K. and Vardhan. V.A. (1989); Identification and Interpretation of manufacturing processes patterns through Neural Network, Mathematical Comp. Modeling journal. Vol. 25, No.5, PP15-36.

Rosenblatt F. (1962); On the convergence of reinforcement procedures in simple perceptrons Technical Report VG1196-G-;Cornell Aeronautical Laboratory.

Rumelhart D.E., Hinton G.E. and Williams R.J. (1986); Learning Representations by back propagating errors. Nature, 323-533-536.

Shink F.G. (1988); Process Control System: Application, Design and Tuning. Third edition McGraw Hill publishing Company. PP246-247

Smetanin, Y.G. (1995); Neural Networks as system for pattern Recognition: A Review. Pattern Recognition and Image Analysis, Vol. 5,2, PP245-293.

Specht D.F. (1990); Probabilistic Neural Network and the polynomials Adaline as a Complementary techniques for the classification IEEE Trans. Neural Networks 1(1):111-121.

Sorsa T. and Koivo H.N. (1991); Neural Networks in process fault diagnosis, IEEE Trans. Sys. Man and Cyber, 21,815-825.

Staib W.E. and Staib B.R. (1993); The Intelligent Arc Furnace Controller; A Neural network Electrode position Optimization System for the Electric Arc Furnace. Proc of the Intern. Joint Conf. On neural Networks New York.

Stephanopoulos G. (1982); Optimization of Closed-Loop Responses, Process Control, Vol. 2 T.F. Edgar (ed); AIChE Modular Instruction. American Institute of Chem. Engrs. New York.

Stephanopoulos G. (1984); Chemical Process Control: An Introduction to theory and practice. Prentice Hall International, Inc London

Szu. II (2003). Bio digest-Stem Cells. Neural Network Society Newsletter 1(1), 1-2.

Szu H. and Hwang W.L (2003); Self-Supervised Back propagation in Stem Cells, Newsletter of the IEEE Neural Networks Society. Vol. 1 (3).

Tank D.W. and Hopfield J.J. (1987); Neural computation by time compression. Proc. National Academy of Science. USA, 84: 1896-1900

Thrope, C.M. Hebert, T. Kanade and S. Shafer (1991); Towards Autonomous Driving: The CMU Navlab. IEEE Expert, PP31-34.

Venkatasubramannian V., Vaidyanthan R. and Yamarnoto Y. (1990), Process Fault Detection and Diagnosis using Neural Networks. Compute. Chem. Engng. 14, 699-712

Watanabe K., Abe M., Kubota M. and Himmelblau D.M. (1989); Incipient fault diagnosis of chemical processes via artificial neural networks, AIChE. J. 35, 1803-1812.

Watanabe K., Hirota S., Hou L., and Himmelau D.M. (1994); Diagnosis of multiple simultaneous fault via Hierarchal artificial neural networks, AIChE.J 40,839-848.

Werbos P.J. (1988); Generalization of Back propagation with Application to a Recurrent Gas Market Model, Neural Networks Vol. 1, No 4 pp 339-56

Werbos P.J. (1974); Beyond Regression: New tools for Prediction and Analysis in the Behavioral Sciences, Ph. D Thesis, Harvard university.

Widrow B. and Hoff M.E. (1960); Adaptive Switching Circuit (1960) IRE WESCON Convention Record New York.

Williams R.J and Peng J. (1990); An Efficient Gradient-Based Algorithms for On-line Training of Recurrent Network Trajectories. Neural Computation vol 2: pp 490-501

Williams R.J. and Zipser D. (1989); Experimental Analysis of the real time recurrent learning Algorithms, Connection Science, 1:87-100.

Williams R.J. and Zipser D. (1989); A learning algorithm for continually learning fully recurrent neural networks. Neural Computation 1:270-280

Whitley J.R. and Davis J.F., (1994); A Similarity-Based Approach to Interpretation Sensor data Using Adaptive Resonance Theory. Compute Chem. Engrg. 18, PP 637-661,

Zaabab, A.H., Q. J. Zhang and M. Nakhla, (1994); "Analysis and Optimization of Microwave Circuit and Devices Using Neural Network Models, MTT-S Int. Microwave Symp. Dig., PP393-396

Zhang J. and Julian A. (1994); A sequential training strategy for locally Recurrent Neural Networks.

Zhang, Q.J., and G,L. Creech (1999); Special Issue on Application of Artificial Neural Networks to RF and Microwave Design, International Journal of RF AND Microwave Computer Aided Engineering, Vol. 9, NY: Wiley.

Zipser D. (1989) A subgrouping strategy that reduces complexity and speeds up learning in Recurrent Networks, Neural Computation, VOL. 1, PP 552-8.