Design and Implementation of Intelligent Control Schemes for pH Neutralization Process

THESIS

Submitted in partial fulfillment

of the requirements for the degree of

DOCTOR OF PHILOSOPHY

by

PARIKSHIT KISHOR SINGH

(2007PHXF432P)

Under the Supervision of

PROF. SUREKHA BHANOT

DR. HARE KRISHNA MOHANTA

BITS Pilani

Pilani | Dubai | Goa | Hyderabad

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

2015

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

CERTIFICATE

This is to certify that the thesis entitled **"Design and Implementation of Intelligent Control Schemes for pH Neutralization Process"** and submitted by **Parikshit Kishor Singh,** ID No **2007PHXF432P** for award of **Ph.D. Degree** of the Institute embodies original work done by him under my supervision.

(Signature of the Supervisor)

PROF. SUREKHA BHANOT

Professor

Birla Institute of Technology & Science, Pilani

Pilani - 333031 (Rajasthan) INDIA

Date:

Place: Pilani

(Signature of the Co-supervisor)

DR. HARE KRISHNA MOHANTA

Assistant Professor

Birla Institute of Technology & Science, Pilani

Pilani - 333031 (Rajasthan) INDIA

Date:

Place: Pilani

Dedicated

To

My Beloved and Inspiring Parents

Shrimati Sunaina Devi

and

(Late) Shri Harendra Kumar Singh

ABSTRACT

Modern process plant automation requires, in lower to higher hierarchy, digital control system, advanced process control schemes, management information system and optimization techniques. Intense global competition, profit based business strategies, rapidly changing socio-economic conditions, demands of better quality control, increased safety concerns and stringent environmental norms are prompting many process industries to automate their operations for optimum performance and productivity. These requirements demand the advanced control system to be accurate, robust, reliable, efficient, optimal, adaptive, and intelligent. Therefore, there is a continuing need for research on optimized advanced process control schemes.

Highly nonlinear behavior and time varying parameters of pH process makes it a benchmark for modeling and control of nonlinear processes. pH measurement and control plays an important role in nutrition, agriculture, food processing, pharmaceutical and biopharmaceutical manufacturing, industrial fermentation and brewery industry, thermal power plant, iron and steel industry, and many other process applications such as wastewater and industrial effluent treatments. This thesis describes, with extensive experimentation and simulation, three aspects of a pH neutralization process: (i) Dynamic modeling, (ii) Intelligent Control, and (iii) Optimization. The neutralization of strong acid (Hydrochloric acid, HCl) and strong base (Sodium Hydroxide, NaOH) streams for experimentation part is carried out in the multifunctional Process Control Teaching System (PCT40) with Process Vessel accessory (PCT41) and pH Probe accessory (PCT42) of Armfield® Ltd., United Kingdom. The pH neutralization system is interfaced with Laboratory Virtual Instrumentation Engineering Workbench (LabVIEW®) for communication, control and display.

In this work, two approaches to the dynamic modeling have been used, namely first principle modeling and system identification. The modeling based on first principles uses laws of conservation of mass, and physical and chemical laws applicable to the pH process. This, however, requires simplifying assumptions which limit the accuracy of developed model. System identification, on the other hand, aims at the development of a mathematical model using experimental data obtained from Armfield pH neutralization system. Artificial Neural Network (ANN) model based on experimental data has exhibited good accuracy. Calibrations of pH

sensor and pump actuators are also incorporated in model development based on both first principles and ANN. The dynamic pH model has been used for various simulation studies involving servo and regulatory operations in conventional and intelligent pH control schemes, and in optimization of pH controller parameters.

Recent advances in control methodologies are focused on developing intelligent control based on computational intelligence paradigms. For conventional control of pH neutralization system, an accurate model of the complex and time-varying nonlinear processes is required which is very difficult to achieve. Fuzzy logic control is a practical common sense based alternative for control of such nonlinear processes since it incorporates the method for constructing nonlinear controllers based on heuristic experience. This thesis compares performance variables, such as Integral of Squared Errors (ISE), and maximum overshoot or undershoots, of optimized conventional Proportional-Integral-Derivative (PID) and fuzzy control techniques for servo and regulatory operations. The present work describes finding optimum parameter settings of the pH controller using various search and optimization techniques such as Genetic Algorithm (GA), Differential Evolution (DE), and Particle Swarm Optimization (PSO). The thesis also describes the convergence of above optimization techniques.

Since pH neutralization of a strong acid and strong base process is highly nonlinear, the fuzzy logic controller parameters alter as the operating conditions changes, requiring it to be retuned. To incorporate self-tuning mechanism, a fuzzy self-tuning controller has been designed, changing its output scaling factor according to error and change in error. The self-tuned fuzzy logic controller has been found to operate over wide range with satisfactory performance. Results of optimized fuzzy logic controller and self-tuned fuzzy logic controller for servo and regulatory operations have also been compared.

ACKNOWLEDGEMENTS

Foremost, I would like to express my humble gratitude and sincere thanks to my supervisors Prof. Surekha Bhanot and Dr. Hare Krishna Mohanta for their valuable guidance, encouragement, suggestions, and moral support throughout the period of this research work. It has been a privilege for me to work and learn under them.

Next I would like to express my deep sense of satisfaction to BITS Pilani, Deemed University, for providing all the necessary facilities and support to complete the research work. My special thanks to Prof. V. Sambasiva Rao, Acting Vice Chancellor, BITS Pilani, and Prof. Ashoke Kumar Sarkar, Director, BITS Pilani, Pilani campus, for giving me an opportunity to pursue my research successfully. I am thankful to Prof. G. Sundar, Director, Off-Campus Programmes & Industry Engagement, for motivating me throughout the duration of this degree programme.

I am thankful to Prof. Sanjay Kumar Verma, Dean, and Prof. Hemant Ramanlal Jadhav, Associate Dean, Academic Research Division (Ph.D. Programme) for providing all necessary guidelines and extending full support which were crucial for the completion of this thesis.

Thanks are due to Prof. S. Gurunarayanan, Dean, Work Integrated Learning Programmes Division, Admissions and Instruction Division, and Prof. Niranjan Swain, Dean, Practice School Division for their support regarding facilitation of industrial visits for better understanding of thesis objectives. I would like to thank Prof. S.C. Sivasubramanian, Dean, Administration, and Prof. J.P. Mishra, Unit Chief, Information Processing Centre for encouragement and support direly needed during difficult phase of thesis writing.

I am thankful to members of Doctoral Advisory Committee, Prof. Sudeept Mohan and Prof. Hitesh Dutt Mathur, for their critical comments, which helped me to improve the quality of manuscript.

Thanks are due to Prof. Anu Gupta, Head of Department, Prof. Vinod Kumar Chaubey, Prof. Hari Om Bansal, Prof. Navneet Gupta, and other faculty colleagues of Dept. of Electrical & Electronics Engineering for their constant motivation and encouragement.

I am thankful to Prof. Suresh Gupta, Head of Department, Dept. of Chemical Engineering, for granting me uninhibited permission to use complete facilities at Process Control Laboratory. I convey my thanks to Prof. Arvind Kumar Sharma, and other faculty colleagues of Dept. of Chemical Engineering for their constant motivation and encouragement.

I sincerely thank Prof. Bharti Khungar, Prof. Ajay Kumar Sah, and Dr. Paritosh Shukla, Dept. of Chemistry, for helping me to understand different Chemistry related concepts.

I wish to acknowledge the support provided by Dept. of Science & Technology, Ministry of Science & Technology, Govt. of India under Fund for Improvement of S&T Infrastructure in Higher Educational Institutions (FIST) scheme to Dept. of Chemical Engineering, BITS Pilani, Pilani campus. The experimental validation of modeling and control strategies has been carried out on FIST sponsored Process Control Teaching System (PCT40) plus Process Vessel accessory (PCT41) plus pH Probe accessory (PCT42) of Armfield® Ltd., United Kingdom.

I also acknowledge technical support received from professionals of Armfield Ltd., UK, especially John Quirk, Ann Knott, and Tom Morton. I sincerely thank Armfield Ltd., UK, for providing a pair of silicone tubing free of charge as a complementary gesture.

I am thankful to Dr. Vimal Bhanot for showing right path and directions during many crucial phases in my life at BITS Pilani, Pilani campus.

I express my regards to my friends, especially Dr. G. Muthukumar and Dr. Satish Kumar Dubey, my students, especially Mr. Vinit Bansal, Ms. Ayushi Kansal and Ms. Surabhi Mahata, all other faculty colleagues of BITS Pilani especially Prof. Ajit Pratap Singh, Prof. Shamsher Bahadur Singh, Prof. P. Srinivasan, Prof. B.K. Rout, Prof. P.R. Deepa, Mr. K. Venkatasubramanian, Dr. Rajeev Taliyan and Dr. Navin Singh, and all staff members of BITS Pilani, Pilani campus especially Mr. Birdi Chand, Mr. Babu Lal Saini, Mr. Jangvir, Mr. Subodh Kumar Azad, Mr. Ashok Saini, Mr. Jeevan Lal Verma, Mr. Om Prakash Shekhawat, Mr. Hazari Lal Saini, Mr. Mahendra Kumar Sharma and Mr. Krishna Gopal Daiya for their support and cooperation. I would also like to thank research scholars Mr. Somesh Mishra, Mr. Saswat Kumar Pradhan and Mr. Tapas Kumar Patra for making the time spent at Process Control Laboratory so memorable.

Lastly, in everyday life, the portion of Earth exposed to Sun is said to have day, whereas the remaining portion is said to have night. Similarly behind all accomplishments there are equal sacrifices. I am thankful to Almighty God for giving His divine blessings to my wife Mrs. Rashmi Singh, my daughter Ms. Ananya Singh, and my son Mr. Janmejay Singh. I deeply value the cooperation and support received from my mother in-law Shrimati Saraswati Devi, father in-law Shri Luvleen Kumar Singh, sisters Mrs. Punam Singh and Mrs. Priyanka Kumari Singh, brothers Mr. Prashant Kumar Singh, Mr. Rahul Kishor Singh and Mr. Parashar Kishor Singh, brothers in-law Dr. Abhimanyu Kumar Singh and Mr. Chandrashekhar Kumar Singh, and other family members.

Parikshit Kishor Singh

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
ANFIS	Adaptive Network based Fuzzy Inference System
ANN	Artificial Neural Network
ArmIFD	Armfield Interface Device
ARMAX	Auto Regressive Moving Average eXogenous
ARX	Auto Regressive eXogenous
BIS	Bureau of Indian Standards
BP	Back-Propagation
CI	Computational Intelligence
COG	Centre of Gravity
CPCB	Central Pollution Control Board
CSTR	Continuous Stirred-Tank Reactor
DE	Differential Evolution
DLL	Dynamic Link Library
EA	Evolutionary Algorithm
FIS	Fuzzy Inference System
FLC	Fuzzy Logic Control
FOPDT	First-Order Process with Dead Time
GA	Genetic Algorithm
GD	Gradient-Descent method with constant learning rate
GDA	Gradient-Descent method with Adaptive learning rate
GDAM	Gradient-Descent method with Adaptive learning rate and Momentum

GDM	Gradient-Descent method with constant learning rate and Momentum
ID	Internal Diameter
IMC	Internal Model Control
ISE	Integral of Squared Errors
LM	Levenberg-Marquardt algorithm
LMN	Local Model Network
MPC	Model Predictive Control
MRAC	Model Reference Adaptive Control
PCT40	Armfield® Process Control Teaching System
PCT41	Armfield® Process Vessel Accessory
PCT42	Armfield® pH Sensor Accessory
PD	Proportional-Derivative
PI	Proportional-Integral
PID	Proportional-Integral-Derivative
PSO	Particle Swarm Optimization
RBF	Radial Basis Function
RCD	Residual Current Device
SR	Servo-Regulatory
STR	Self-Tuning Regulator
UOD	Universe Of Discourse

CHAPTER 1

INTRODUCTION

In recent years, there is a major spurt in modernization of industrial plants through process automation since the new competitive business strategy is based on pricing, production, scheduling and delivery-time (Lederer & Li, 1997). Modern process plants aim for increased productivity, better product quality and growing profit in order to remain competitive in global economy. Process automation is essential for economical plant operation through efficient techniques for energy utilization and waste minimization. Also modern process industries must meet obligation to laws concerning increased safety levels and reduction in environmental pollution.

Control system design is greatly influenced by amount of nonlinearity present within process. If nonlinearity encountered is very mild, a linear model can adequately represent the process and a classical controllers such as Proportional-Integral-Derivative (PID) or Proportional-Integral (PI) based on linear control theory provide satisfactory control over a wide operating range. In presence of appreciable amount of nonlinearities, however, such linear models are ineffective since even small disturbances can force process away from the operating point. A good way to compensate processes with known nonlinearities and operating condition variations is use of adaptive control techniques. An adaptive control system automatically adjusts its parameter using feedforward, feedback or both strategies to compensate for corresponding variations in the properties of the process (Shinskey, 1979; Cohen & Friedmann, 1974). Adaptive control of nonlinear process using linear controller requires adjustable controller parameters and a mechanism such as gain scheduling for adjusting parameters (Åström & Wittenmark, 2008). To deal with severe nonlinearities and address concerns of varying operating conditions and parameter variations in modern process plant, development of rigorous dynamic plant model is required.

Highly nonlinear behavior and time varying parameters of pH process makes it a benchmark for modeling and control of nonlinear processes. Nonlinear processes can be modeled using two ways, namely mathematical modeling based on first principles approach and system identification based on experimental input-output data. Dynamic modeling based on first

principle, also known as '*white-box'* models, uses laws of conservation of mass, and physical and chemical laws applicable to the pH process. McAvoy et al. (1972) proposed use of material balances and electroneutrality relations on component ions to derive a simple dynamic pH equations in Continuous Stirred-Tank Reactor (CSTR) for single streams of strong base and weak acid process. Few other first principle based pH modeling techniques are also developed using concept of reaction invariants (Gustafsson & Waller, 1983) and strong acid equivalent (Wright & Kravaris, 1991). First principle based dynamic models developed with idealistic assumptions such as perfect mixing and absence of measurement noise do not represent true and realistic behavior of process. Therefore, nonlinear adaptive control schemes based on '*white-box*' models are ineffective in modern process plant (McAvoy, 1972; Gustafsson, 1985; Wright et al., 1991).

System identification based dynamic '*grey-box*' and '*black-box*' models can be designed using experimental data obtained from the process. The '*grey-box*' modeling technique requires some insight into the system apart from the experimental data. Among various nonlinear models based on *'grey*-box' techniques, Wiener, Hammerstein, and Wiener-Hammerstein models based on block-structured approaches using linear and nonlinear elements are quite popular (Norquay et al., 1999; Fruzzetti et al., 1997; Park et al., 2006). However, fundamental limitations of '*grey-box*' model based nonlinear adaptive control strategies are choice of suitable nonlinear structure and its order. Moreover, inability to accurately estimate model parameter often restricts their applications.

The '*black-box*' modeling technique is based on input-output behavior of the system without any knowledge of system configuration. Over last two decade, Artificial Neural Network (ANN) has been the most popular and successful *'black-box*' modeling technique with wide range of nonlinear system applications. ANN based modeling is inspired by biological neural networks and it comprises a set of interconnected nonlinear processing element known as artificial neuron. ANN has an excellent ability to learn nonlinear dynamics of a complex process because of its inherent parallel and distributed configuration (Bhat & McAvoy, 1989). ANN based model predictive control techniques have been extensively developed for pH neutralization process (Draeger et al., 1994; Krishnapura & Jutan, 2000; Åkesson et al., 2005).

The focus of advanced control methodologies now a day is to develop intelligent control algorithm based on Computational Intelligence (CI) paradigms. The intelligent controllers have capability of self-organizing and take appropriate control actions in case of any change in process conditions, much like human nature of first think and then act. The recent focuses of CI paradigms are ANN, fuzzy logic, Evolutionary Computation such as Genetic Algorithm (GA) and Differential Evolution (DE), Swarm Intelligence such as Particle Swarm Optimization (PSO).

Fuzzy logic based on definition of fuzzy set theory proposed by Zadeh (Zadeh, 1965) deals with an ambiguous and imprecise class of objects which are characterized by membership functions with membership degrees assigned between 0 and 1. Fuzzy logic introduced concept of linguistic variables, fuzzy conditional statements and Fuzzy Inference System (FIS) to analyze an ill-defined complex systems and decision processes. Though fuzzy logic brought an unconventional shift in nature of computing based on words and perceptions, it faced however initial resistances from scientists and researchers. Almost a decade later synthesis of Fuzzy Logic Control (FLC) schemes, famously known as Mamdani type FLC, for a small boiler steam engine combination could be realized (Mamdani & Assilian, 1975). Self-organizing fuzzy controller having capability to modify its control rule base were also developed (Procyk & Mamdani, 1979). Further an alternative and simpler Sugeno type FLC scheme was also applied for system identification and control (Takagi & Sugeno, 1985). Fuzzy logic based pH control has been developed to obtain intelligent equivalent of the conventional counterpart such as PI, PID and sliding-mode, and model predictive control techniques (Babuska et al., 2002; Tzafestas & Papanikolopoulos, 1990; Chen & Chang, 1998; Cho et al., 1999).

Evolutionary algorithms (EAs) are population based search techniques in which optimal solution is reached on the basis of Darwin's theory of biological evolution. As per Darwinian theory, the principle of natural selection favors those species for survival and further evolution which are fittest. Over number of years, various but independent types of EAs were developed by many scientists and researchers with an aim of utilizing them for optimal solution of various engineering problems. However, GA conceived by Holland and its variants developed by his associated team members received wide attention (Holland, 1992; Goldberg, 1989). Many variants of GA contributed immensely to scientific and engineering applications. GA is also

applied for parameters optimization of various controllers (Yeo & Kwon, 2004; Karr & Gentry, 1993; Oh et al., 2004).

DE is a stochastic EA in which optimization function parameters are represented as floating-point variables (Storn & Price, 1996; Price, 1996). The performance of DE in optimization of many real-valued, multi-modal functions is found to be superior in comparison with many other evolutionary optimization methods. Also many DE variants have been developed for real-valued objective function optimization problems (Storn, 1996). DE has also found applications in industrial automation and control (Sickel et al., 2007; Syed & Abido, 2013).

PSO is a population based stochastic search technique which simulates the movement of organisms such as bird flocking or fish schooling (Kennedy & Eberhart, 1995). The main feature of PSO is mutual and social cooperation of individual particles where they take a decision on basis of current and previous exchanged information with their neighboring particles in population. Many researchers have used particle swarm algorithm in optimization problems (Kennedy, 1997; Shi & Eberhart, 1998).

From above discussion, it can be seen that problem of pH control in neutralization processes is an important topic for research having application in many industrial batch and continuous processes for servo and regulatory operations. In this thesis we have proposed some new pH model and control schemes based on nonlinear dynamics of the neutralization process. Extensive simulation has been done using ANN based dynamic pH model to compare performance of optimized conventional PID and fuzzy logic controllers for servo and regulatory operations using MATLAB®. Optimization of PID and fuzzy logic controllers are carried out using evolutionary and swarm intelligence algorithms namely GA, DE and PSO. The experimental validation of optimized FLC and self-tuned FLC schemes are carried out using LabVIEW® on multifunctional Armfield® Process Control Teaching System (PCT40) with Process Vessel accessory (PCT41) and pH Probe accessory (PCT42) at Process Control Laboratory of Department of Chemical Engineering, BITS Pilani, Pilani campus.

1.1 Objectives of Thesis

The main objectives of thesis are as follows:

(a) Development and validation of dynamic modeling of Armfield pH neutralization system PCT40, PCT41, PCT42

(i) To develop calibration equations for pH sensor and pump actuators.

(ii) To develop first principle and ANN based dynamic models.

(iii) To evaluate performances of dynamic models.

(b) Design and experimental implementation of optimized control of pH neutralization process for servo and regulatory operations

(i) To design PID control and Fuzzy Logic Control (FLC) schemes.

(ii) To optimize PID and FLC schemes using global optimization techniques namely GA, DE, and PSO.

(iii) To evaluate performances of control schemes and optimization methods.

(c) Design and implementation of Self-tuned FLC scheme of pH neutralization process for servo and regulatory operations

(i) To design Self-tuned FLC scheme.

(ii) To evaluate performances of optimized FLC and Self-tuned FLC schemes.

1.2 Organization of Thesis

This thesis is organized in seven chapters including the present chapter.

Chapter 2 on literature survey extensively covers topics such as modeling, control and optimization for pH neutralization process. This chapter begins with definition and characteristics of pH, in order to give a brief introduction about the subject. Also, significance of

pH monitoring and control has been presented clearly. Towards the end of the chapter, gaps in the existing research work have been presented.

Chapter 3 gives detailed description of Armfield pH neutralization system PCT40, PCT41, PCT42. It also describes its interfacing with LabVIEW, and calibration procedure of important components of the system.

Chapter 4 describes development and validation of dynamic pH model using first principle and ANN.

Chapter 5 describes design, simulation and experimental validation of optimized PID and FLC schemes for servo and regulatory operations of pH neutralization process. The optimized controller parameters are evaluated by application of optimization algorithms namely GA, DE, and PSO.

Chapter 6 describes design and experimental validation of Self-tuned FLC for servo and regulatory operations of pH neutralization process. It also describes performance comparison of optimized FLC and Self-tuned FLC schemes.

Finally Chapter 7 highlights the main conclusions of the thesis and provides recommendations for future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

The far-reaching applications of pH measurement in modern commerce and industry necessitated development of controllers that permit pH processes to be regulated automatically. As explained in Chapter 1, monitoring and control of pH neutralization process is a complex and challenging problem. Many researchers have attempted to solve it using conventional, adaptive and intelligent strategies. Present chapter starts with a brief description of pH characteristics and its significance. Detailed literature survey on the pH neutralization process covers established concepts and techniques in dynamic modeling, control and optimization. Based on literature survey, motivation for the research has been elaborated towards end of this chapter.

2.2 pH and its Characteristics

The acidity of an aqueous solution is determined by $[H^+]$ in the solution (Bates, 1965). Because $[H^+]$ in an aqueous solution is typically quite small, Sørensen in 1909 proposed use of 'hydrogen ion exponent (pH)' to conveniently express $[H^+]$ as shown in equation (2.1).

$$[H^+] = 10^{-pH} \tag{2.1}$$

pH, therefore, is defined as

$$pH = -\log[H^+] \tag{2.2}$$

The pH of a solution is measured using a pH meter and its value depends upon strength and concentration of aqueous solution. The pH is a measure of acidic, basic or neutral nature of an aqueous solution. Generally, the pH scale ranges from 0 to 14. At 25 °C, aqueous solutions with pH value less than 7 are said to be acidic, those with pH value greater than 7 are said to be basic and those with pH value equal to 7 are said to be neutral.

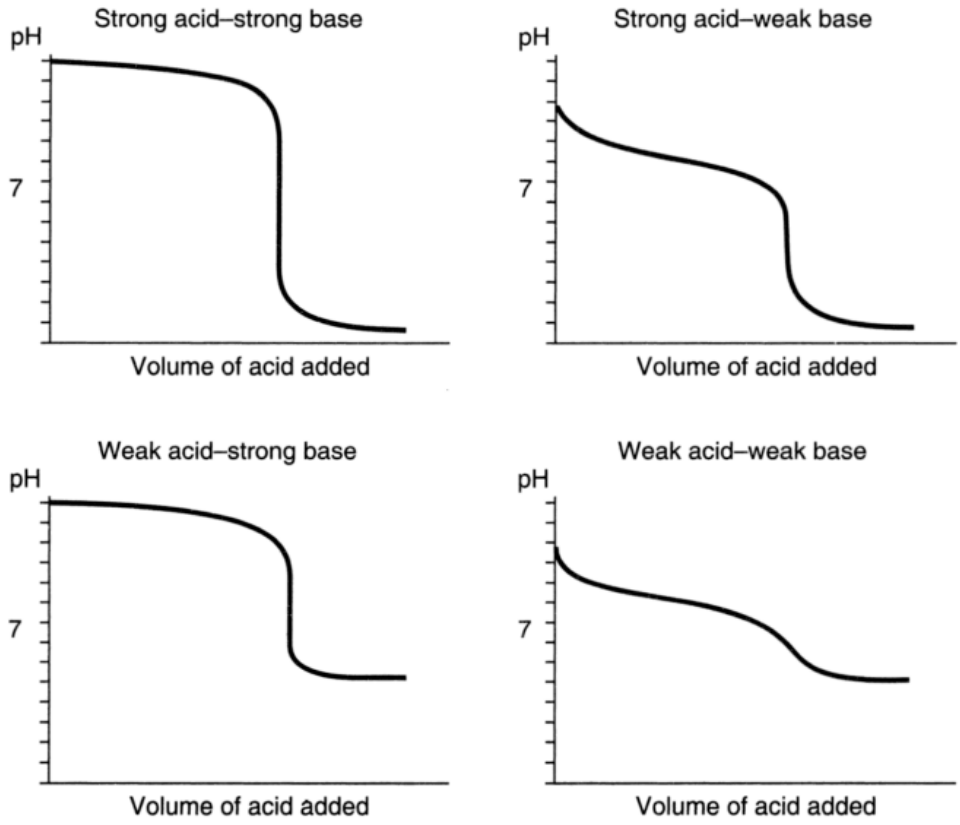


Figure 2.1 Titration Curves for different types of acids and bases (*Courtesy: http://intranet.tdmu.edu.ua/*)

A quantitative acid-base reaction is often called a neutralization reaction. For example, when just enough base is added to exactly react with the acid in a solution, we say the acid has been neutralized.

Titration is a quantitative volumetric analysis technique for determining the amount of an analyte by using a standard reagent. Titration method involves delivery of a measured volume of a solution of known concentration i.e. titrant into a solution containing the substance being analyzed i.e. analyte. The point in the titration where enough titrant has been added to react exactly with the analyte is called the equivalence point. The progress of neutralization reaction is often monitored by plotting the pH of the solution being analyzed as a function of the amount of titrant added. Such a static plot is called a titration curve, as shown in Figure 2.1.

In a neutralization reaction, titration curve provides first information report about nature of acids and bases such as strong or weak, monoprotic or polyprotic, and concentrated or dilute. At the

equivalence point, the neutralization process exhibits maximum nonlinearity. For the strong acid-strong base system, the gain at the equivalence point is extremely high and it occurs at neutral pH. Controlling such system near neutral pH would place very high demands both on the accuracy of the control system and on the rangeability of the reagent delivery system. Clearly, the weak acid-weak base system is easier to control because of the lower gain near neutral point. However, it is important to note that the equivalence point does not always coincide with neutral point. In case of polyprotic acid and monoprotic base neutralization, there are multiple equivalence points.

A buffered solution is one that resists a change in its pH when either hydroxide ions or protons are added. The most important practical example of a buffered solution is our blood, which can absorb the acids and bases produced in biologic reactions without changing its pH. Buffering is an important aspect in strong acid-weak base and weak acid-strong base neutralization processes. Generally, a buffered solution may contain a weak acid and its salt, or a weak base and its salt.

2.3 Significance of pH Control

The pH measurement and control are of vital significance in daily life as well as modern process industries. A constant pH for blood is vital because cells can survive only in a very narrow pH range of 7.35 to 7.45. The pH of human blood should be 7.388 at 38 °C (Semple et al., 1962). The pH of natural milk is around 6.8. The Indian Standard Drinking Water - Specification (Second Revision) IS 10500:2012 from Bureau of Indian Standards (BIS) specifies the pH value to be maintained between 6.5 to 8.5 (Public.Resource.Org, Inc., 2012).

In agriculture, soil pH is considered as an important variable as it controls many chemical processes that take place during growth of crops. The desirable soil pH for optimal plant growth varies among corps and it also affects nutrient content in the corps. The water used in irrigation must have pH in the range 6 to 9 (Venkateswarlu, 1996).

Food industry extensively uses processing, packaging, storage and transportation of agro products provided they meet Food Safety and Standards Regulations, 2011 (Food Safety and

Standards Authority of India, 2011). The packaged foods and drinks are acidic to keep their usable life prolonged and their pH values are limited by BIS guidelines.

Pharmaceutical formulations are often buffered and maintain a constant pH in order to minimize drug degradation and, improve patient comfort and efficacy of delivery (Pandit, 2007). In biopharmaceutical industry pH control is essential for biopharmaceutical products since metabolic changes within micro-organisms can change the pH of their environment and hence the process conditions. In addition many biopharmaceutical processes such as culture control, stem cell and protein aggregation require strict pH monitoring and control (Wilkins, 2011).

Fermentation is an essential step in brewery industry. pH measurements in the fermentor is used to indicate the progress of the fermentation process, ultimately influencing the growth of the culture, its cellular metabolism and quality of the final product (Humphrey & Deindoerfer, 1961).

In thermal power plants, pH of boiler feedwater is kept in the range 9 to 9.5 to reduce corrosion of carbon steel, and cooling tower water should have pH value of around 7.5 to 8.2 in presence of Hypobromous acid (HOBr) (Venkateswarlu, 1996). Studies at Bethlehem's Sparrows Point Plant in United States of America suggests the water pH range as 6.8 to 7 for cooling of blast furnaces and open hearths in iron and steel industry (Walling & Otts, 1967).

The law of conservation of mass states that matter can neither be created nor can be destroyed but can change its form. Nature is the ultimate source of enormous quantity of raw water required in various industries and nature is the ultimate destination for almost same quantity of polluted water discharged from the very same industries. It is therefore of utmost importance to apply wastewater treatment on these industrial effluents to maintain ecological balance (Ahmed et al., 2010; Rodrigues et al., 2014). The Central Pollution Control Board (CPCB) of India has been entrusted with various powers including inspection of wastewater treatment installations and frequently monitors quality of industrial effluents discharged to water, air and soil. As per CPCB standards for emission or discharge of various environmental pollutants from various industries, the pH of effluents must lie within the range 6.5 to 8.5 (Central Pollution Control Board, 2007).

2.4 Overview of Modeling and Control of pH Neutralization Process

For a control system to operate satisfactorily, it must have the abilities of measurement, comparison, correction and control. Early trend in industrial pH measurement and control shows progression from manual to automatic operations based on analog instrumentation (Stock, 1991). The digital revolution of late 1950s to late 1970s brought major change in industrial operation from mechanical, pneumatic and vacuum tube based electronic technology to first transistor, followed by integrated circuit based digital technology. With adoption and proliferation of digital computer by various process industries, a new era of modern automatic process control dawned (Bennett, 1996). Due to better efficiency, accuracy and reliability, modern automatic process control applications increased rapidly over a very wide range of process engineering field, from the manufacturing of precision devices used for miniaturized systems to that of massive equipments used for the industrial processes (Williams, 1966; Williams, 1970). Among numerous additional benefit of modern process control are: higher production rates, improved productivity, increased profit, better product quality, efficient energy utilization, economical plant operation, minimization of waste materials, increased safety levels and reduction in environmental pollution. Increased computational power of digital computer also led to development of process modeling and simulation for the design and optimization of conventional, adaptive and intelligent control applications in modern process industries.

2.4.1 First Principles based Modeling and Adaptive Control

Initially, study of pH process dynamics in CSTR formed the basis for development of First-Order Process with Dead Time (FOPDT) model based control (Harriot, 1964). Based on FOPDT model, Harriott (1964) illustrated estimation of proportional controller gain in feedback configuration. Shinskey (1979) divided the titration curve into three zones as per severity of the nonlinearity. The piecewise linear proportional controller assigned low gain in most severe nonlinear region and high gain in remaining two regions. Further the width of the low gain zone adjusted adaptively using a feedforward controller. Mellichamp et al. (1966a) addressed the time varying characteristics of pH process through a periodic estimate of the process gain and used a PID controller to study the performance of adaptive identification system (Mellichamp et al., 1966b).

Equation (2.2) shows that a change in hydrogen ion concentration by factor of ten accounts for unity change in pH value. The simplistic FOPDT models were inadequate to represent severe nonlinearity of a typical pH process which called for development of a more rigorous and generalized dynamic model. McAvoy et al. (1972) proposed use of material balances on the species present in acetic acid (CH_3COOH)-sodium hydroxide (NaOH) solution and dissociation constants, thus avoiding direct material balances on hydrogen ion or hydroxyl ion. McAvoy (1972) compared performances of Ziegler-Nichols tuned PID and Proportional-Derivative (PD) controllers, and time optimal control using experimental as well as theoretical studies. Experimental studies pointed out that PD has comparable performance with time optimal control whereas PID performance is poor. Use of simplistic formulation made model proposed by McAvoy et al. (1972) popular and it is used in development of various modern control strategies for single acid-base streams (Yeo & Kwon, 1999; Venkateswarlu & Anuradha, 2004; Tan et al., 2005).

In another approach, researchers used reaction invariants variables which are not affected by the chemical reactions taking place in a process (Asbjørnsen, 1972). For example, in the process reaction $A \rightarrow B$, if the total concentration variable $[C_A+C_B]$ remains unchanged, then this variable is called as reaction invariant (Waller & Mäkilä, 1981). The control strategy based on reaction invariant needs *a priori* knowledge of the reaction invariant dynamics of the process. In addition, some knowledge of the chemical contents is also desirable. Gustafsson & Waller (1983) proposed partitioning of concentration vector based dynamic model into two parts for systems with arbitrary number of acid and base streams: one part based on reaction invariants expressing the thermodynamic state of the system and another part based on reaction variants describing the chemical reactions. Reaction invariants based dynamic model are used to demonstrate adaptive pH control by estimating the process gain in the form of a nonlinear function of pH (Gustafsson & Waller, 1983; Gustafsson, 1985). Development of dynamic pH model based on reaction invariants also led to design and implementation of many modern control strategies (Hall & Seborg, 1989; Babuska et al., 2002).

Wright & Kravaris (1991) developed a linear and non-adaptive approach for design of nonlinear controllers for pH neutralization processes based on strong acid equivalent of the system. The method derived a reduced-order model based on rigorous mathematical description of the

dynamic pH process in terms of titration curves. The strong acid equivalent is used as a state variable in this reduced model and it can be calculated on-line from pH measurements using the nominal titration curve of the inlet process stream. A linear PI controller in terms of the strong acid equivalent is then used to close the loop. Wright et al. (1991) demonstrated the performance of the strong acid equivalent control method using experimental results for the Hydrochloric acid (HCl)-NaOH and CH_3COOH-NaOH systems.

2.4.2 Nonlinear Adaptive Control

Nonlinear dynamics of pH neutralization process lead to development of many variants of adaptive control, popular ones are Gain-scheduling, Model Reference Adaptive Control (MRAC) and Self-Tuning Regulator (STR). Gain-scheduling is based on determination of process operating conditions and then accordingly change the controller parameters in order to compensate process variations. Lin & Yu (1993) proposed a framework for automatic tuning and gain-scheduling based on modified Gulaian-Lane titration curve model. Application of gain-scheduled autotune variation method on a relay feedback experiment resulted simultaneously in the ultimate gain and ultimate frequency required for autotuning. Chan & Yu (1995) applied gain-scheduled PI controller to demonstrate nonlinear pH control. The gain-scheduled autotune variation method is used to estimate the titration curve and PI controller parameters. Klatt & Engell (1996) combined feedback linearization and trajectory based gain-scheduling technique to obtain nonlinear pH control of CH_3COOH-NaOH system. Use of dual techniques preserved the advantages and overcame shortcoming of both concepts. Zhang (2001) proposed neuro-fuzzy network models based gain scheduling strategy in which the process operation is partitioned into several fuzzy operating regions, and within each region, a local linear model is used to model the process. The global model output is obtained through defuzzification. The process knowledge is used to train the network. Finally nonlinear controller is developed by combining several local linear controllers that are tuned on the basis of the local model parameters. Nyström et al. (2002) approximated pH neutralization process as a linear parameter-varying system using a set of velocity-form linearizations. Application of gain-scheduled controller using state estimator and stationary Riccati equation resulted in performance improvement in terms of increased control accuracy.

MRAC uses a reference model of the process which tells how the process output should ideally respond to the command signal. Although MRAC is a good alternative to PID, but it has to be tuned for each particular process and the tuning depends on the presence of lag, delay and other factors. For processes not well known, the controller must be tuned experimentally and it could be a disadvantage from a commercial or business point of view. Palancar et al. (1996) used it for the pH neutralization of wastewater streams containing CH_3COOH and Propionic acid (CH_3CH_2COOH) with NaOH stream.

STRs are intended to control systems with unknown but either constant or slowly varying parameters. STRs are generally composed of three parts: a parameter estimator, a linear controller and a block which determines the controller parameters from the estimated parameters (Åström et al., 1977). Proudfoot et al. (1983) demonstrated superior performance by self-tuning PI controller as compared to fixed parameter based PI controller. Babuska et al. (2002) achieved desired accuracy using fuzzy self-tuning PI controller. Alpbaz et al. (2006) applied self-tuning PID controller for neutralization of limestone with Sulphuric acid (H_2SO_4) using Auto Regressive Moving Average eXogenous (ARMAX) based system model, pseudo-random binary sequence as forcing function in order to identify dynamics of the process and Bierman algorithm for model parameters evaluation.

2.4.3 Internal Model Control (IMC)

IMC is based on the knowledge of a supposed model of the process (Garcia & Morari, 1982). The performance of the control system can be compensated by its robustness to process modification or modeling errors (Corriou, 2008). Rivera et al. (1986) reported IMC based PID controller parameters for different process models. Choi & Rhinehart (1987) simulated an adaptive IMC strategy based on a reduced phenomenological model of the wastewater neutralization process. The reduced phenomenological model of the controller considers wastewater as a single fictitious acid of unknown concentration and of unknown Gibbs free energy of dissociation. Choice of few adjustable model parameters led to faster parameterization and matches the model to measured process data. Chen et al. (1996) proposed IMC strategy based on fuzzy neural network. Narayanan et al. (1997) applied nonlinear IMC in combination with strong acid equivalent and a robust nonlinear control law which resulted in improved

disturbance rejection and servo control capabilities. Brown et al. (1997) presented a hybrid learning approach for Local Model Networks (LMNs) comprising Auto Regressive eXogenous (ARX) local models and normalised Gaussian basis functions. Based on LMNs of the nonlinear plant, IMC based controller is derived analytically. Edgar & Postlethwaite (2000) simulated servo and regulatory actions of a nonlinear fuzzy relational model based IMC strategy for a multi-variable pH system. Toivonen et al. (2003) simulated performance of IMC to nonlinear pH neutralization process described by linear parameter-varying models using velocity-based linearizations. Kim et al. (2012) presented a nonlinear IMC procedure for a pH process modeled as a stable Wiener system, which composed of a stable linear system followed by a static nonlinearity.

2.4.4 ANN based Modeling and Model Predictive Control (MPC)

ANN an important technique in computational intelligence domain made its importance felt in chemical engineering application in late 1960s, but ANN is applied for pH system identification during late 1980s only (Bhat & McAvoy, 1989). From 1990s onwards, researchers started exploring ANN for predictive control too (Koivisto et al., 1991). Kwok et al. (1994) utilized a modified Elman neural network to construct control systems for industrial pH processes. Draeger et al. (1994) used neural network based pH model in the extended Dynamic Matrix Control algorithm. Cheng & Himmelblau (1995) used an internal recurrent neural network model for identification of pH neutralization process with unknown dead time. Lightbody et al. (1997) demonstrated higher performance of B-spline neural network compared to multilayer perceptron for online adaptation of pH neutralization process model. Agarwal (1997) organized systematically neural network based control schemes into a multi-level classification using their essential functional features. Pottmann & Seborg (1997) proposed a Radial Basis Function (RBF) based predictive control strategy for pH neutralization process. The predictive controller provided better tracking and disturbance rejection as compared to conventional PI controller. Palancar et al. (1998) designed a pH control system based on combination of direct and inverse ANN model. Kuo & Melsheimer (1998) reported time-lag recurrent RBF neural network model for predictive control of a pH neutralization system. Yeo & Kwon (1999) developed a neural PID controller and tested it experimentally for pH neutralization process. In the neural PID control method, the neural network is trained on the basis of the control errors and the tuning parameters

of the PID controller are obtained as the outputs of the neural network. Hagan & Demuth (1999) published a tutorial on varieties of neural network based MPC techniques. Krishnapura & Jutan (2000) reported adaptive neural network control influenced by PID controller for pH neutralization process. The algorithm has few number of weights similar to that found in a PID controller. The new structure, with its very few weights, overcomes the problem of excessive tuning parameters. Zheng & Wang (2002) proposed a neural network based model free control technique for pH neutralization process. Hadjiski et al. (2002) identified the relative properties of eight different neural network control structures for pH neutralization process. Based on extensive simulation results, a Hammerstein plant model is recommended. Oh & Pedrycz (2002) investigated polynomial neural networks performance for pH neutralization process. Polynomial neural network has a self-organizing neural network architecture whose topology develops through learning. In particular, the number of layers of the network is not fixed in advance but evolves as learning progresses. Åkesson et al. (2005) reported increased computational efficiency by modeling pH neutralization process with a set of models constructed by velocity-based linearization. The resulting quasi-linear models also simplifies the estimation of the system state from the measured outputs. The on-line computational burden associated with the controller calculation is reduced by using a neural network function to approximate the optimal MPC strategy. Elarafi & Hisham (2008) presented modeling of the pH neutralization plant using empirical techniques and investigated the performance of an ANN based predictive controller against the more traditional PID controllers. The empirical model was found closest to a second–order with dead time and the predictive controller outperformed the conventional PI or PID controllers.

2.4.5 Nonlinear MPC

Review articles published by Henson (1998), Morari & Lee (1999), and Qin & Badgwell (2003) give a broad overview of many variants of MPC technique and their applications. Many popular MPC strategies incorporated nonlinear process models based on neural network, Wiener, Hammerstein, Volterra series and Laguerre polynomial techniques. Norquay et al. (1997) and Norquay et al. (1999) used Wiener model as an effective way of introducing nonlinearity to a control problem without significantly increasing complexity of the nonlinear MPC. In case of highly nonlinear pH neutralization process, Wiener model based MPC was shown to posses

excellent set-point tracking and disturbance rejection capabilities, as compared to linear MPC and PID schemes. Fruzzetti et al. (1997) reported simulation studies of Hammerstein models as part of nonlinear MPC strategy for a pH neutralization process. Gómez et al. (2004) utilized input-output data from a first principles simulation model of the pH neutralization process for subspace-based identification of a black-box Wiener-type model. The proposed Wiener model is used as the internal model in a MPC. Shahraeini et al. (2006) applied standard quadratic programming algorithm to optimize Wiener model. Shafiee et al. (2006) developed a piecewise linear Wiener model for nonlinear MPC of pH neutralization process. Arefi et al. (2006) applied neural network for Wiener model based identification of pH neutralization process consisting of acidic, basic and buffer streams. Oblak & Škrjanc (2007) proposed to combine a fuzzy-system approximation of the output mapping with the linear dynamics to calculate the model-output prediction. Further, Oblak & Škrjanc (2010) reported continuous-time Wiener MPC of a pH process based on a piecewise linear approximation. Wenfeng et al. (2009) developed a nonlinear model predictive technique for pH control of a rolling wastewater effluent. Hermansson et al. (2010) developed control strategy in which Bayesian weight calculator to combined set of piecewise linear models results in a single linear model describing the pH neutralization system. In addition, other nonlinear MPC methods such as those based on Wiener–Laguerre and Volterra series models are also reported in the literature (Mahmoodi et al., 2009; Díaz-Mendoza & Budman, 2010).

2.4.6 Fuzzy Logic based Intelligent Control

Zadeh (2008) discussed about unconventional perspectives of fuzzy logic, namely graduation, granulation, precisiation and the concept of a generalized constraint. Zadeh summarized that in large measure, the real-world is a fuzzy world and to deal with fuzzy reality we need fuzzy logic. Therefore, in coming years, fuzzy logic is likely to grow in visibility, importance and acceptance. Zadeh (1973) also introduced fuzzy logic based approach for the complex systems which are not well-defined. Mamdani (1974) published fuzzy logic based control, popularly known as Mamdani controller, for steam plants. Mamdani (1977) applied fuzzy logic to control the boiler pressure of the steam plants. Mamdani also introduced concept of self-organizing controller whose primary property is to self-tune the fuzzy controller settings depending on the process conditions. In addition, Mamdani in collaboration with fellow researchers contributed immensely

to development of fuzzy logic controller for varieties of dynamic systems (King & Mamdani, 1977; Procyk & Mamdani, 1979). Takagi & Sugeno (1985) suggested another approach, popularly known as Sugeno method for system identification and control. Kwok & Wang (1993) developed a fuzzy pH controller based on PD algorithm whose performance is enhanced using normal integrator and a Smith predictor. Parekh et al. (1994) developed fuzzy logic based in-line control scheme for nonlinear pH process. This relatively simple approach led to wider operating range as well as robust handling of random disturbances. Aoyama et al. (1995) proposed an internal model control scheme using fuzzy neural network for pH neutralization modeling and compared its performance with PID controller. In order to avoid exponential increase in the number of hidden layers, this novel fuzzy neural network structure used hyper ellipsoids. Jang & Sun (1995) unified both neural networks and fuzzy logic modeling concept, and introduced a new technique called Adaptive Network based Fuzzy Inference System (ANFIS). Eikens et al. (1995) compared three different approaches for pH neutralization process using fuzzy neural network structure based on RBF. First method used a generic fuzzy controller in which rule base and membership functions are selected intuitively. For second method, fuzzy system is initialized using error back-propagation algorithm. Third method is on-line adaptive adjustment of membership functions of second method. Chen & Chang (1996) described design methodology of a neural or fuzzy variable structural PID control system for pH process. In this controller, the PD mode is used in the case of large errors to speed up response and PI mode is applied for small error conditions to eliminate steady-state offset. In addition, controller is capable of changing its parameters based on operating conditions. Sing & Postlethwaite (1997) proposed fuzzy relational model based predictive control for highly nonlinear processes such as pH neutralization. Garrido et al. (1997) compared the performance of fuzzy logic based wastewater pH controller with other controller such as general model control, single-acid general model control, multi-acid feedback control and model reference adaptive control. Behera & Anand (1999) modeled pH neutralization system using a fuzzy neural network, in which on Lyapunov synthesis approach, two controllers namely disturbance invariant and model predictive are developed using the dynamic pH model. Ylén (1998) used modified self-organizing controller algorithm by introducing stochastic measures of the controller behavior as additional criteria over and above the traditional performance measures. Cho et al. (1999) reported a MPC scheme based on the Sugeno model of pH neutralization process. Adroer et al. (1999) used a fuzzy logic structure

coupled with a tuning factor, in order to account for variation due to titration curves, unknown water composition, buffering capacity of the systems and the changes in input loading in wastewater neutralization process. Zárate et al. (2001) applied a variable structure fuzzy controller and utilized a Smith predictor to account for the pH process time lag. Leng et al. (2002) created a self-organizing neural network to implement pH system identification. Fuente et al. (2002) and Fuente et al. (2006) divided the pH neutralization process into three fuzzy regions based on high, medium and low process gain. The proposed fuzzy controller is designed to cover entire operating range. Babuska et al. (2002) implemented self-tuning tuning scheme of fuzzy inference system based on the pH value of fermentation process. Babuska & Verbruggen (2003) applied neuro-fuzzy methods for pH process identification. Venkateswarlu & Anuradha (2004) proposed a self adjustable dynamic fuzzy adaptive controller for a weak acid-strong base pH process. It consists of a low-level basic control phase with a minimum rule base and a high-level dynamic learning phase with an updating mechanism to interact and modify the control rule base. The proposed controller performed better than conventional PI and PID controllers. Wan et al. (2004) and Wan et al. (2006) adjusted Mamdani fuzzy controller parameter using a least square algorithm with deadzone. Zeybek and Alpbaz (2005) proposed fuzzy dynamic matrix control scheme for pH control in a dye wastewater plant. Jia et al (2005) used a neuro-fuzzy approach to describe the nonlinearity of the Hammerstein model, thus avoiding various restrictions during use of polynomial approach. Bharathi et al. (2006) compared performances of gain-scheduled PI, neural network and fuzzy logic controllers, applied independently to a pH neutralization process. In order to take the pH region of operation into account, the fuzzy controller also used set-point as its input variable apart from two standard inputs, error and change of error. Oblak & Škrjanc (2006) used fuzzy system approximation technique for Wiener MPC of a neutralization process. Wan & Kamal (2006) applied tuned Type-II fuzzy controller for pH neutralization process. Li et al. (2006) proposed fuzzy generalized predictive control for nonlinear pH neutralization process. Ibrahim & Murray-Smith (2007) applied Mamdani fuzzy based pH controller for a neutralization process pilot plant using feedback and feedforward schemes. Palancar et al. (2007) applied fuzzy logic to PD and PID control schemes for pH control of a neutralization process. Salehi et al. (2009) presented an adaptive fuzzy control scheme for pH neutralization processes which performed better than tuned PI. Liao et al. (2009) presented Type-II Takagi-Sugeno fuzzy logic based modeling on data clustering, and designed

two kinds of predictive controllers based on crisp output and type-reduced set. Jiayu et al. (2009) found performances of fuzzy based PID to be superior than PID or fuzzy logic based control schemes in anaerobic wastewater treatment application. Saji & Sasi (2010) tuned a PI controller for pH process using fuzzy logic based on sliding mode control principle. Karasakal et al. (2013) proposed an on-line tuning method for fuzzy PID controllers via rule weighing mechanism. The effectiveness of the proposed on-line weight adjustment method is demonstrated on a pH neutralization process. Heredia-Molinero et al. (2014) presented a feedback PID-like fuzzy control scheme to deal with instability near the equivalence point in pH neutralization processes. State space analysis of the titration curves and a fuzzy clustering algorithm based on calculating a measure of potential derived from the square distance of the pH data are complementary applied to define the membership structure and the fuzzy sets of the controller.

2.4.7 Evolutionary and Swarm Algorithms based Optimization

In particular, controller designed for process control applications must be optimized for efficient operation. Over last two decade many researchers have utilized global optimization techniques based on evolutionary algorithms such as GA and DE, and swarm algorithm such as PSO for controller optimization.

Even though Holland introduced GA in 1975, its application in engineering optimization started almost a decade later. Wang & Kwok (1992) utilized GA to simulate an optimization mechanism in order to refine the rule base of fuzzy PID controller for a heating process control system. Karr & Gentry (1993) applied GA technique to design adaptive fuzzy logic controller for pH neutralization process. The proposed scheme successfully controlled the pH system in a reasonable time for various disturbances. In order to optimize the parameters of polynomial fuzzy neural network based pH controller, Kim et al. (1996) applied a hybrid genetic optimization approach which combines GA with Nelder and Mead's simplex method of optimization. Chen & Chang (1998) applied GA and Taguchi methods separately in order to obtained optimal fuzzy sliding-mode controller parameters. Simulation results indicated better performance of GA than Taguchi method. Khemliche et al. (2002) designed a GA based fuzzy controller. Oh et al. (2004) and, Oh & Roh (2010) applied computational intelligence based fuzzy controller to nonlinear inverted pendulum. The design approach used was to first tune the

scaling factors of the fuzzy controller, and then estimate them using neuro-fuzzy networks based model. Chou (2006) proposed GA based optimal fuzzy controller design by establishing an index function as the consequent part of the fuzzy control rule. The inputs of the controller, after scaling, are utilized by the index function for computing the output linguistic value. This linguistic value can then be used to map the suitable fuzzy control actions. Roh et al. (2007) applied concept of information granulation driven genetically optimized fuzzy set based polynomial neural networks to pH neutralization process modeling. Valarmathi et al. (2007) proposed GA based optimal fuzzy controller for pH neutralization process. Valarmathi et al. (2008) applied Sugeno fuzzy model for PID parameter estimation and used GA to obtain optimal values. Sharma et al. (2012) proposed a novel technique for automatic exploration of the genetic search space using fuzzy coding based algorithm.

Since its inception in 1996 by Storn and Price, another popular evolution algorithm known as DE has drawn the attention of many researchers all over the world resulting in a lot of its variants exhibiting improved performance. Das & Suganthan (2011) presented a detailed review of the basic concepts of DE, a survey of its major variants and an overview of the significant engineering applications. Pishkenari et al. (2011) utilized DE for the optimization of a fuzzy controller membership functions in path tracking of a mobile robot and compared its performance with respect to GA. Sickel et al. (2007) applied DE in a reference governor to generate optimal set points and gain tuning for the control of a power plant. Comparison of DE performance with PSO resulted in DE slightly outperforming PSO. Syed & Abido (2013) presented simulation analysis for DE based intelligent speed regulator for permanent magnet DC motor. Yu et al. (2008) proposed a nonlinear MPC algorithm based on DE and RBF neural network. For a CH_3COOH-NaOH pH neutralization process, RBF neural network is used for modeling and DE algorithm is used to design optimal predictive controller.

Kennedy and Eberhart introduced PSO as a member of swarm algorithms in 1995. Han et al. (2009) used PSO in order to improve system identification capacity of neural networks for nonlinear dynamic systems. Tang et al. (2010) developed PSO based approach for Wiener model identification. The identification process is carried out in two stages. First, sequences of step signals are supplied to the system to identify the static nonlinear function. Then, the linear dynamic subsystem is identified from the view point of optimization using PSO. Han et al.

(2011) reported adaptive parameters adjustment of a dynamic feedforward neural network using Gaussian PSO in the training process. Sivaraman et al. (2011) compared performances of PI controller based pole placement technique, fuzzy model based on fuzzy c-means and PSO based fuzzy c-means algorithm for a pH process. Aras et al. (2011) conducted simulation and experimental studies for design of multiregional fuzzy logic based controller in pH neutralization process of CH_3COOH-NaOH. Scaled coefficients of the controller were optimized separately using GA and PSO. Although fitness function values obtained via GA and PSO algorithms were very close to each other, but computational complexity of GA is more as compared to PSO.

Finally it is worth mentioning that few researchers, such as Yang et al. (2007), have reported hybrid optimization techniques based on GA, DE and PSO.

2.5 Gaps in Existing Research

The literature survey presented in the previous section shows that for over last six decades, researchers have proposed many pH control schemes using different techniques such as conventional, adaptive and intelligent. However, there are still considerable challenges in dynamic modeling, control and optimization of pH neutralization process.

HCl is an important and widely used chemical in steel pickling process in iron and steel industry, ore processing in mining industry, wastewater treatment in food processing, and neutralization reaction in chemical manufacturing. Literature survey shows that strong acid-strong base neutralization have not been investigated extensively.

Many proposed dynamic pH models and subsequent control schemes are based on weak acid-strong base neutralization process. Surely such formulation, particularly first principles based, will not be exactly applicable to strong acid-strong base such as HCl-NaOH neutralization process. Thus there is need to develop first principle based dynamic pH model which gives performance comparable to experimental observation. As reasoned earlier, first principle based model does not represent all the nonlinear dynamics of pH neutralization system. In particular, the random variations in pH sensor values and process parameter variations cannot be accounted

in first principle model. All these necessitates development of ANN based dynamic pH model using experimental values, which has capability to learn highly nonlinear behavior.

As discussed in literature, controller needs to be optimized for their desired operation. Evolutionary and swarm algorithms provide a methodology for objective function optimization. Reported works in literature do not provide a comprehensive performance comparison of controller parameter optimization using GA, DE and PSO for pH control of strong acid-strong base neutralization process. Moreover, literature survey shows that many reported works are based on simulation studies only and their extensive experimental validations are often lacking. Further self-tuning is an important aspect of intelligent control strategy which has not been thoroughly investigated. This research work, therefore, compares performances through simulation studies and experimental validation of evolutionary and swarm algorithms based controller optimization, and demonstrates online performance of self-tuned fuzzy controller.

2.6 Concluding Remarks

In this chapter we have presented a brief overview of pH characteristics. Also significance of pH control has been illustrated with the help of specific examples in various process applications. Further contributions of various researchers in the field of modeling, control and optimization of pH neutralization process have been presented briefly. In particular, emphasis of literature survey has been given for computational intelligence techniques such as neural networks based process modeling, fuzzy logic based process control and evolutionary algorithms based optimization of controller parameters. Finally the gaps in the existing research are outlined.

CHAPTER 3

DESCRIPTION OF pH NEUTRALIZATION PROCESS

3.1 Introduction

Armfield® Process Control Teaching System (PCT40) with Process Vessel Accessory (PCT41) and pH Sensor Accessory (PCT42) has been used as a pH neutralization system for testing the performance of the models and control strategies developed. pH neutralization system is interfaced with LabVIEW® software for communication, user interface, control and various applications development. In this chapter, first a brief overview of the pH neutralization system and its interfacing with LabVIEW has been discussed. Thereafter, calibration procedure of important components like pH sensor and pumps of the system has been presented in detail.

Figure 3.1(a) Armfield PCT40 with PCT41 and PCT42 as pH neutralization system

3.2 Description of Armfield® pH Neutralization System

Figure 3.1(a) shows the Armfield pH neutralization system at Process Control Laboratory in Department of Chemical Engineering, BITS Pilani, Pilani campus. Its specifications are given in Appendix A1. The PCT40 base unit is designed for implementing control schemes such as feedback, feedforward, ratio and cascade on a variety of process control loops like level, temperature, flow and pressure using manual, On/Off and PID controller modes (Armfield Limited, 2005). The PCT40 base unit consists of moulded plinth, two process vessels, pumps, sensors and actuators as well as a mounting point and electrical connections for the PCT41 accessory. The PCT41, acting as CSTR, expands the capabilities of the PCT40 (Armfield Limited, 2006a). Using PCT42 along with PCT40 and PCT 41, many control strategies can be realized with pH as a process variable (Armfield Limited, 2006b). A brief description of few components is given below with reference to schematic diagram of the pH neutralization system as shown in Figure 3.1(b).

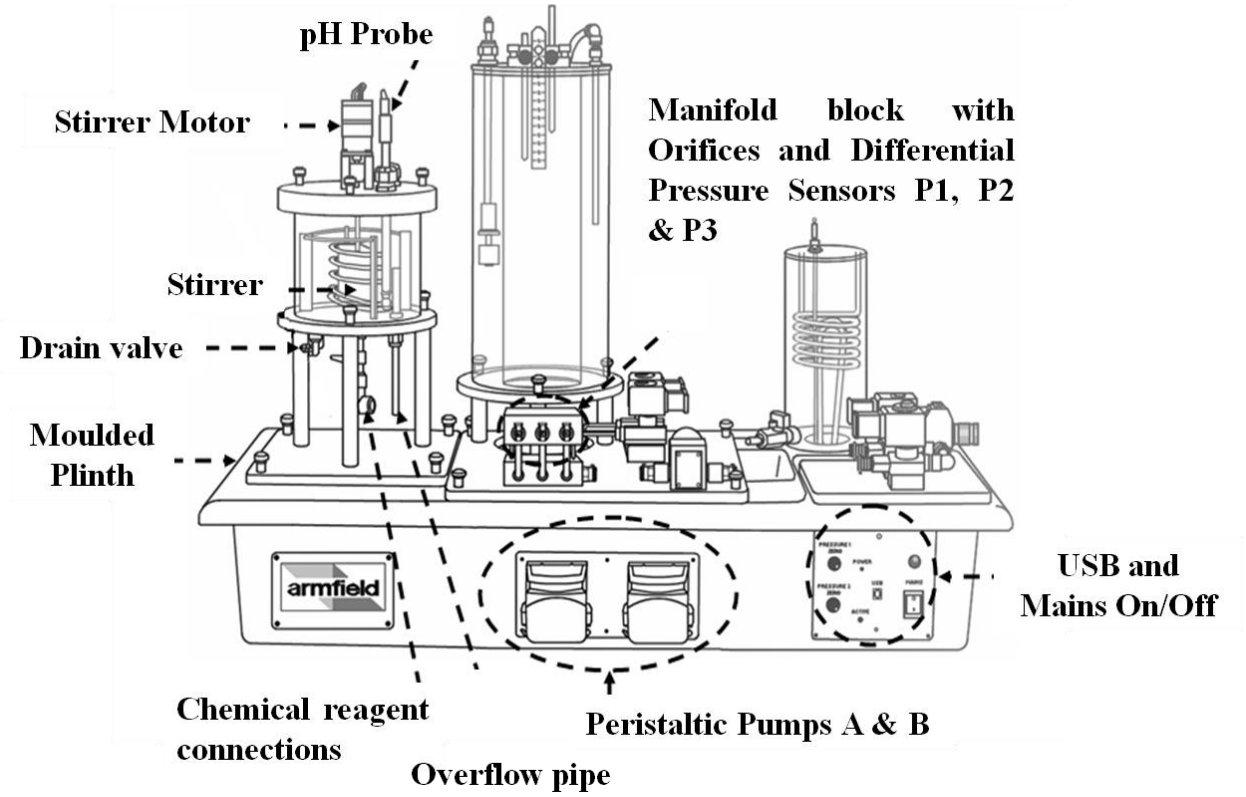


Figure 3.1(b) Schematic diagram of Armfield PCT40 with PCT41 and PCT42 (*Courtesy: Armfield Ltd., UK*)

Moulded Plinth: Base plate for the PCT41 accessory unit is located on a moulded plinth and secured using nuts. A mains power inlet socket and sensor connections for the accessory are located at the rear of the plinth, which has Residual Current Device (RCD) as circuit breakers for electrical safety. The left hand end of the plinth includes a drainage channel with a drain valve located at the end of the channel. The right hand end of the plinth contains a 60-pin Input/Output (I/O) connector whose pin description has been given in Appendix A1.

USB and Mains On/Off: Front of the plinth incorporates mains On/Off switch and a USB socket for connection to a computer.

Peristaltic Pumps A & B: Peristaltic pumps are the simplest pump, with no valves, seals or glands to clog or corrode. The fluid contacts only the bore of a tube, eliminating the risk of pump contaminating fluid, or the fluid contaminating pump. Watson-Marlow® manufactured peristaltic pumphead of type 313D has three rollers whose specifications are mentioned in Appendix A4. Silicone tubing with 3.2 mm internal diameter (ID) and 1.6 mm wall thickness has been used with the apparatus. Pumps A and B are located on left-hand and right-hand sides respectively on the front of the plinth. In this research work, pump A has been used for carrying HCl solution and pump B has been used for carrying NaOH solution.

Manifold block with Orifices and Differential Pressure Sensors P1, P2 & P3: The manifold block incorporates three orifices with associated differential pressure sensors from Honeywell® that can be used to measure flowrate. A brief specification detail of differential pressure sensors has been given in Appendix A5. The orifice associated with pressure sensors P1 and P2 is 1.9 mm diameter and suited to the low flow rates used in the PCT41. During use, the silicone tube from the peristaltic pump is connected to the small quick release fitting at the front and the silicone tube from the process vessel is connected to the ferrule at the rear so that the fluid flowing to the reactor vessel passes through the orifice. The orifice associated with P3 is 3.7 mm diameter and suited to the higher flowrates used on PCT40.

Chemical reagent connections: Aqueous chemical solutions used in experiments are pumped into the process vessel through two connectors in the base. The solutions should be supplied from suitable containers using the 3.2 mm ID flexible silicone tubing. The tubes passing through the peristaltic pumps A and B are connected to the two forward-facing self-sealing fittings on the

manifold block. A second pair of tubes connects the self-sealing fittings at the back of the manifold block to the reagent connections at the base of the process vessel.

pH Probe: The pH probe PCT42 is a combination electrode from Hanna Instruments®. The specifications of pH sensor have been given in Appendix A6. Glands in the lid of the PCT41 process vessel house the pH sensor. pH sensor must be dipped maximum in the process vessel. This may be achieved by loosening the glands, making the required height adjustment, and re-tightening the glands. The pH sensor is connected to socket at the back of the plinth.

Stirrer: A motor-driven stirrer works in conjunction with a baffle arrangement to provide efficient mixing and heat transfer.

Stirrer Motor: The motor for the stirrer is mounted on the lid of the process vessel. The stirrer motion can be remotely controlled through user program installed on PC connected to the equipment.

Overflow pipe: An overflow pipe or stand pipe within the PCT41 process vessel controls the maximum level in the vessel.

Drain valve: When the PCT41 process vessel is not being used, content solution can be drained through a valve in the base of the vessel.

3.3 Interface of Armfield pH Neutralization System with LabVIEW®

Armfield pH neutralization system is provided with a software package to facilitate the device interfacing with computer through Universal Serial Bus (USB). The device driver is installed on 32-bit Microsoft Windows® XP operating system. The system32 directory contains following device driver files: ARMUSB.INF, ARMFIELDLTDTHERMUSB.INF, THERMUSB.SYS, ARMUSB.SYS, and ArmIFD.DLL. The first two files tell the computer how to recognize the data acquisition card, also called Interface Device (IFD), installed within the base unit PCT40 when the neutralization system is plugged in to the computer. The next two files are the IFD drivers for the USB interface. The last file is a Dynamic Link Library (DLL) which is used to pass data between the user program and the IFD driver through USB interface. Based on four

types of I/O data, as given in Appendix A1 and briefly described below, user can access data logger for the IFD driver through four basic function calls to DLL file.

Analog Inputs (from PCT40 to computer): There are 12 analog input channels corresponding to Pins 1 to 12, each with 0 to 5 V signals digitized into a 12-bit number. The interface will pass a value between 000000000000_B to 011111111111_B to the computer.

Analog Outputs (from computer to PCT40): There are 2 analog output channels corresponding to Pins 22 and 24, each with 0 to 5 V signals taken from a 12-bit number. Here computer must pass a value between 0_D to 2047_D to the interface.

Digital Inputs (from PCT40 to computer): There are 8 digital input channels corresponding to Pins 28 to 31 and 33 to 36. The interface will pass either 0 or 1 to the computer.

Digital Outputs (from computer to PCT40): There are 8 digital output channels corresponding to Pins 38 to 41 and 43 to 46. Here computer must pass either 0 or 1 to the interface.

In this research work, LabVIEW 12.0 has been used to develop applications for calibration, identification, control and optimization of Armfield pH neutralization system. LabVIEW communicates with the Armfield pH neutralization system by accessing appropriate I/O data of the DLL file, using standard call library function node. To use standard call library function node, path, name and prototype of the function and its parameters need to be specified.

3.4 pH Neutralization System Calibration

System calibration is an important step for dynamic modeling and system identification. This section presents calibration of main components of pH neutralization system like peristaltic pumps, differential pressure sensors, pump flowrates and pH sensor.

3.4.1 Calibration of Peristaltic Pumps A and B

Peristaltic pump needs adjustment in order to use flexible silicone tube having 3.2 mm ID and 1.6 mm wall thickness, as shown in Figure 3.2. Peristaltic pumps have two adjusting screws which needs to be put in the uppermost position so that they do not clamp the tube. Then silicone

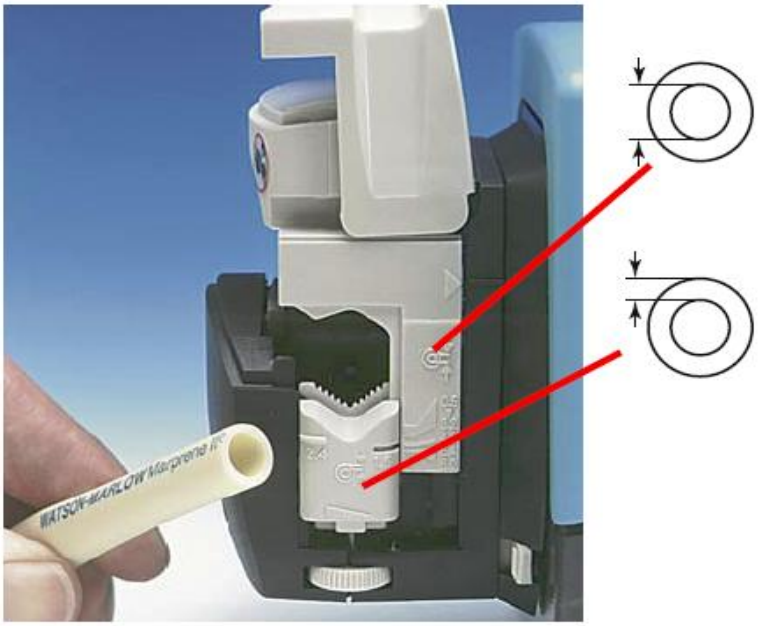


Figure 3.2 Adjusting Watson-Marlow peristaltic pump for silicone tubing (*Courtesy: Watson-Marlow Fluid Technology Group*)

tube is placed on the rollers and each screw is adjusted until it makes contact with the tube. Thereafter the front cover is pulled down to run the pump. The tube should remain stationary. If it moves, then both adjusters must be screwed down further at the same time to prevent the tube from moving. Once the tube is static then pump is ready to be used.

3.4.2 Calibration of Differential Pressure Sensors

Calibration of differential pressure sensors involves adjustment of corresponding 'zero' and 'span' PCT40 potentiometers in signal conditioning circuit by keeping the pump speed at 0% and 90% of maximum value. Appendix A1 provides identification numbers for these PCT40 Variable Resistor (VR) which can be accessed behind the black cover plate on the left-hand side of the module plinth.

Flowchart for data sampling of the calibrated differential pressure sensors corresponding to speed of pump A (S_a) and speed of pump B (S_b) respectively has been given in Figure 3.3(a). Figure 3.3(b) shows the LabVIEW block diagram for capturing pressure versus speed readings at sampling time (T_s) of 1 second. Following are the important points about Figure 3.3(b).

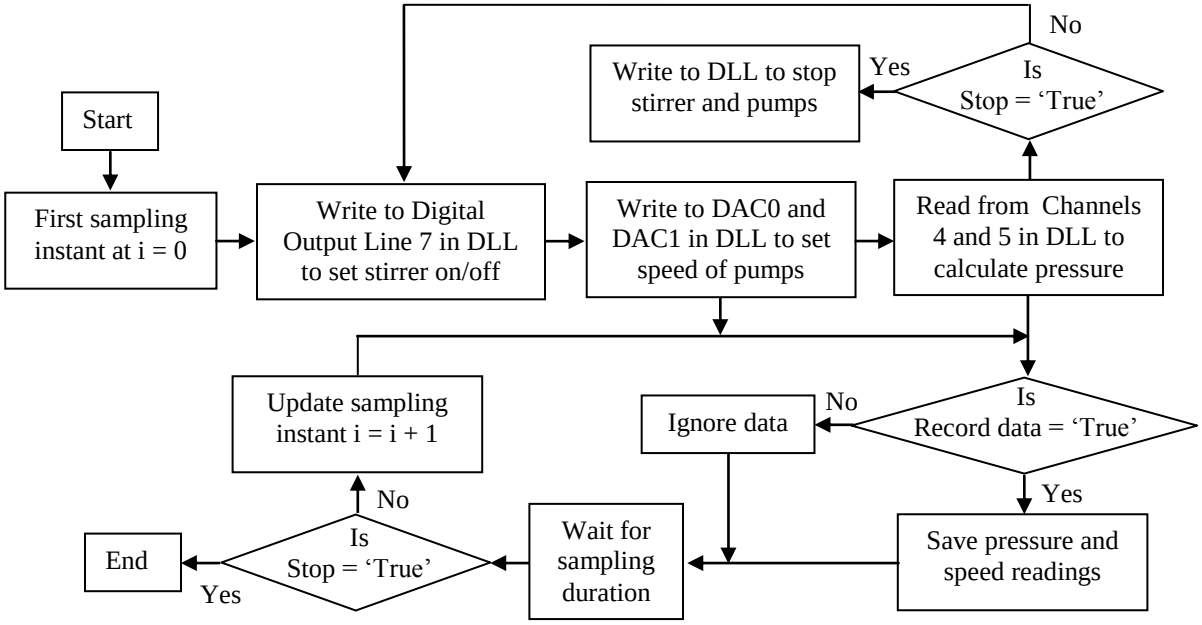


Figure 3.3(a) Flowchart for data sampling in a calibrated differential pressure sensor

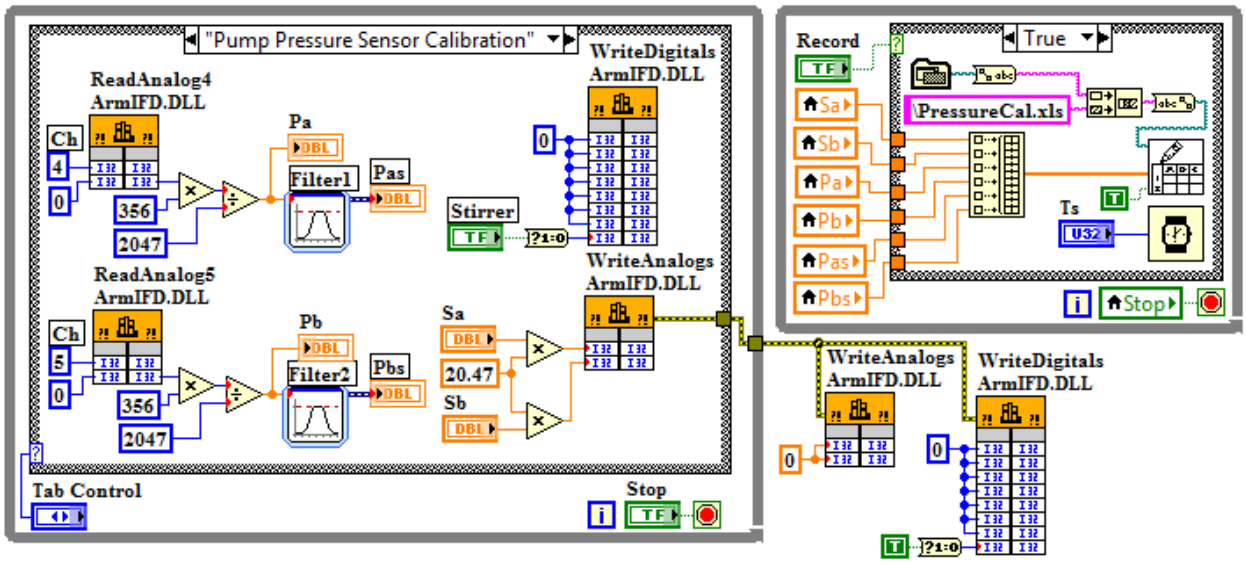


Figure 3.3(b) LabVIEW block diagram for data sampling of calibrated differential pressure sensors

(i) Block named 'WriteDigitalsArmIFD.DLL' is a call library function node whose configuration and setting for DO0 has been shown in Appendix A7. The parameters setting for digital output lines DO1 to DO7 are similar to DO0. DO0 to DO7 represent eight digital output lines 0 to 7 respectively which can be assigned binary values either 0 or 1. Here, DO0 to DO6 are assigned integer value 0 which means they are unused and On/Off status of stirrer is controlled using DO7 through a Boolean to binary interface.

(ii) 'WriteAnalogsArmIFD.DLL' block is a call library function node whose configuration and setting for DAC0 has been shown in Appendix A7. The parameters setting for analog output DAC1 is similar to DAC0. Since DAC0 and DAC1 can be assigned decimal values between 0 to 2047 and the speed of pumps A and B varies from 0% to 100% the pump speed values are multiplied with factor 2047/100 before giving to DAC0 and DAC1.

(iii) 'ReadAnalog4ArmIFD.DLL' and 'ReadAnalog5ArmIFD.DLL' are call library function nodes whose configuration has been shown in Appendix A7. Value4 from Channel4 and Value5 from Channel5 correspond to differential pressure sensors P1 and P2 respectively in which returned decimal values varies from 0 to 2047. Since pressure readings varies from 0 to 356 mm for speed variation in pumps A and B, the parameters Value4 and Value5 are scaled using factor 356/2047.

(iv) Differential pressure sensors corresponding to peristaltic pumps A and B respectively exhibit large variation in instantaneous pressures 'P_a' and 'P_b' respectively for given pump speeds. This variation however can be reduced by applying an external smoothing filter.

(v) Smoothing filter is configured as a moving average filter with a rectangular window of 15 samples. 'P_{as}' and 'P_{bs}' are the averaged pressure readings from peristaltic pumps A and B respectively.

(vi) Experimentation is carried out under the condition that process vessel PCT41 is completely filled with water.

Figures 3.4(a) and 3.4(b) show the instantaneous and average pressure readings in pump A and B for 5% step changes in pump speed. Clearly both the response curves are nonlinear. Further it can be seen that for pump speeds less than 20%, pressure variations are very less. This is because both the pumps A and B are almost stationary due to inertial friction for speeds less than 18%.

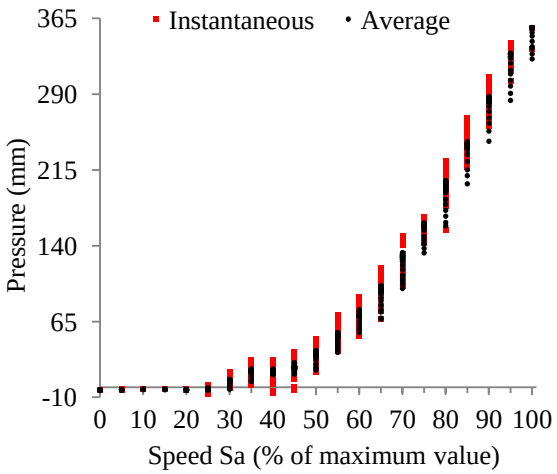


Figure 3.4(a) Response curve of calibrated differential pressure sensor P1 in pump A

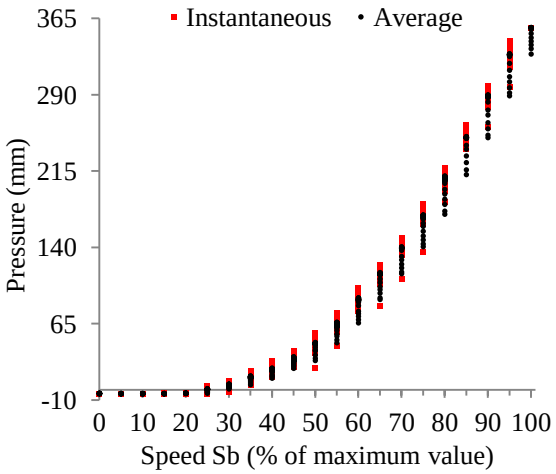


Figure 3.4(b) Response curve of calibrated differential pressure sensor P2 in pump B

3.4.3 Calibration of Peristaltic Pumps Flowrate

Flowrate (F) calculation of pumps A and B involves determination of time duration (T_d) to pump known volume (V_s) of water into completely filled process vessel, at constant speed. For pump speed values equal to or higher than 40%, the known volume of water is 1000 mL; otherwise, the known volume of water is 250 mL. Use of large volume ensures better accuracy in the flowrate calculation. Also, determining time duration in milliseconds further improves the accuracy of flowrate calculation. Flowchart and LabVIEW block diagram for flowrate calculation are shown in Figures 3.5(a) and 3.5(b) respectively. Following are the important points about Figure 3.5(b).

(i) Description and configuration of 'WriteDigitalsArmIFD.DLL' and 'WriteAnalogsArmIFD.DLL' blocks as discussed in section 3.4.2.

(ii) Flowrates F_a and F_b of pumps A and B respectively are measured using same set of values for S_a and S_b.

(iii) The experimentation is carried out under the condition that process vessel PCT41 is completely filled with water.

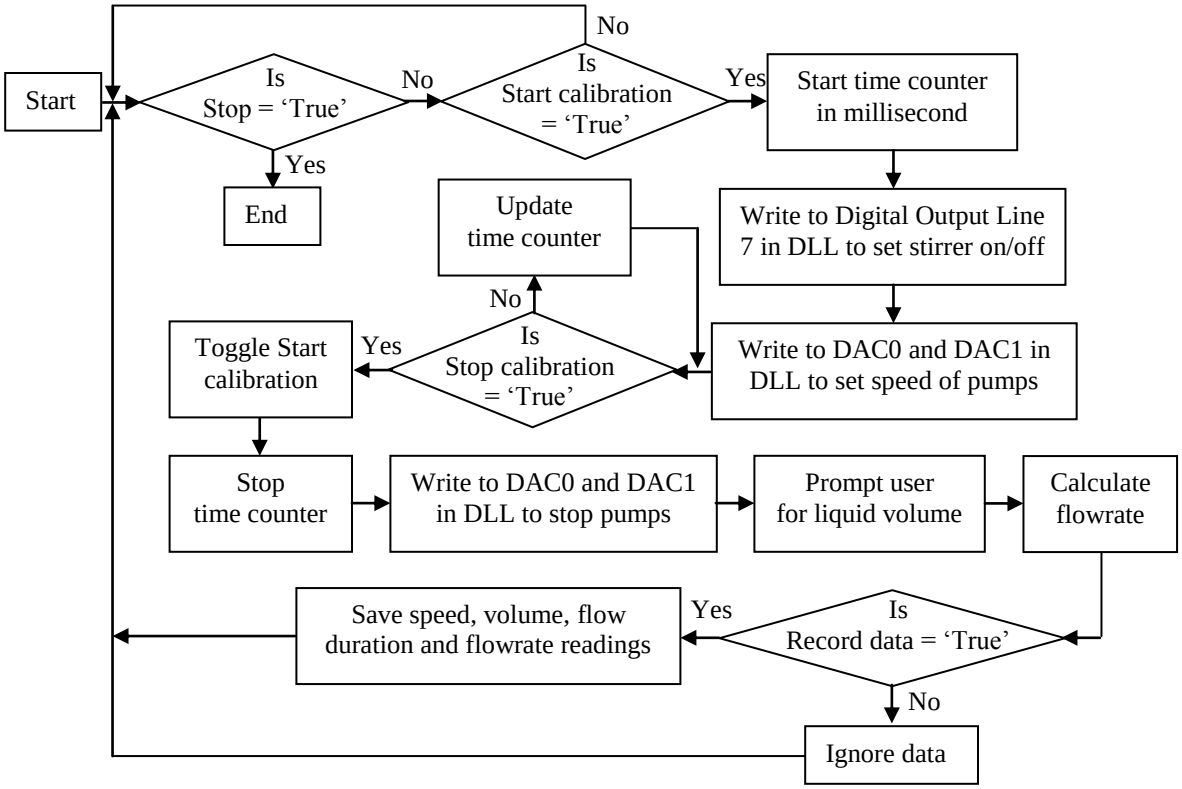


Figure 3.5(a) Flowchart for peristaltic pump flowrate calculation

Figure 3.6 shows the combined plots of F_a against S_a and F_b against S_b. A linear regression analysis results in equations (3.1) to (3.4) when used to estimate the pumps A and B flowrate for various values of their speed. The statistical coefficient R^2 is 0.9985 and 0.9984 for pumps A and B respectively.

$$F_a = 0 \qquad \text{for } 0 \le S_a < 18 \qquad (3.1)$$

$$F_a = 0.0599\ S_a - 0.8761 \qquad \text{for } 18 \le S_a \le 100 \qquad (3.2)$$

$$F_b = 0 \qquad \text{for } 0 \le S_b < 18 \qquad (3.3)$$

$$F_b = 0.068\ S_b - 0.9251 \qquad \text{for } 18 \le S_b \le 100 \qquad (3.4)$$

where flowrates and speeds are expressed in mL/s and % of maximum value respectively.

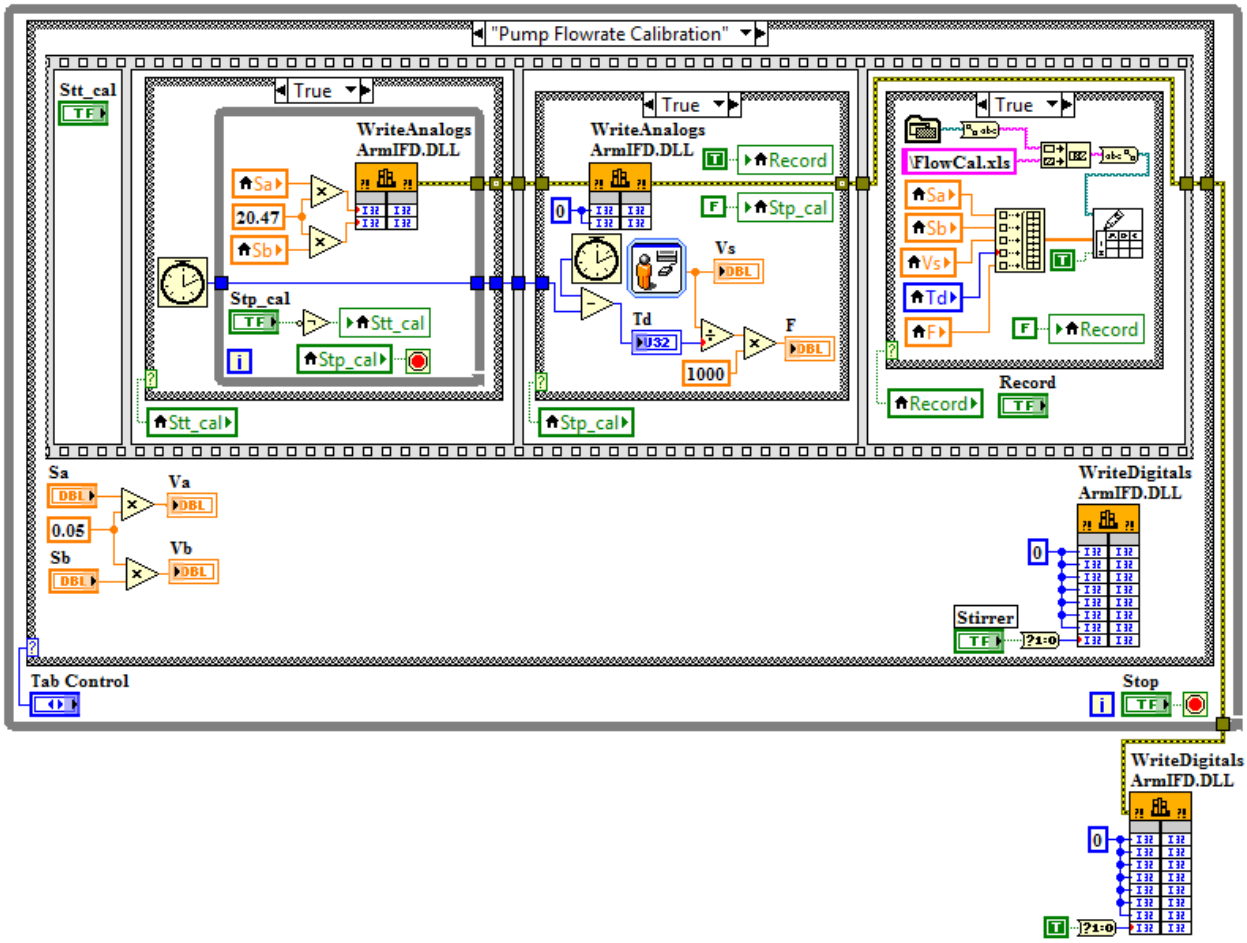


Figure 3.5(b) LabVIEW block diagram for peristaltic pump flowrate calculation

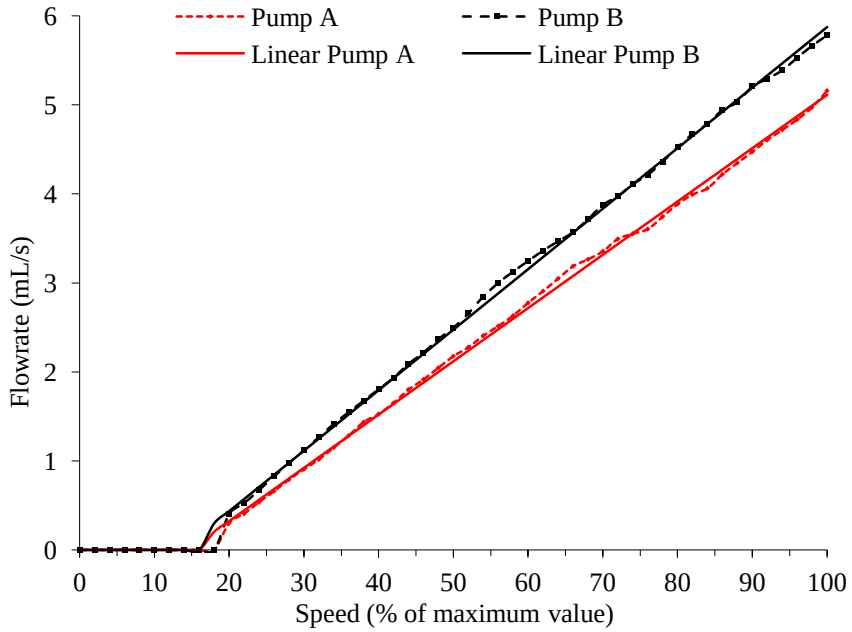


Figure 3.6 Flowrate calibration curve for peristaltic pumps

3.4.4 Calibration of pH Sensor

pH sensor output voltage V_{pH} in volts varies when it is dipped into different standard pH buffer solutions. Appendix A2 gives details of buffer capsules used to prepare 100 mL of buffer solutions having standard pH values as 4, 7 and 9.2. Flowchart and LabVIEW block diagram for pH sensor data collection at sampling time (T_s) of 1 second corresponding to standard buffer solution under stirred condition has been given in Figures 3.7(a) and 3.7(b) respectively. In Figure 3.7(b), 'ReadAnalog11ArmIFD.DLL' is a call library function node whose configuration has been shown in Appendix A7. Parameter Value11 from Channel11 corresponds to pH sensor in which returned decimal values varies from 0 to 2047. Since pH sensor output varies from 0 to 5 V, the parameter Value11 is scaled using factor 5/2047 in order to obtain V_{pH}.

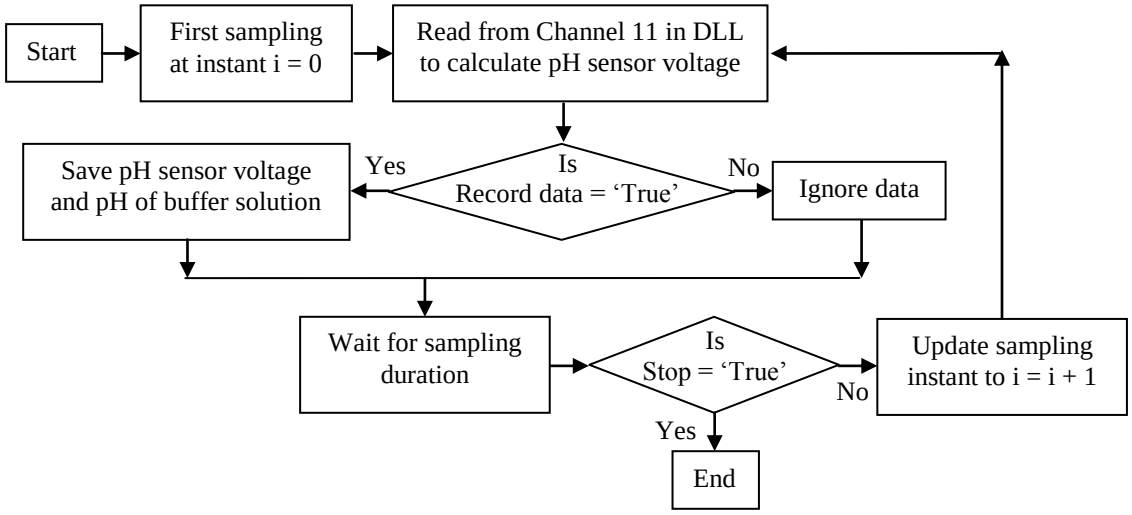


Figure 3.7(a) Flowchart for data sampling of pH sensor

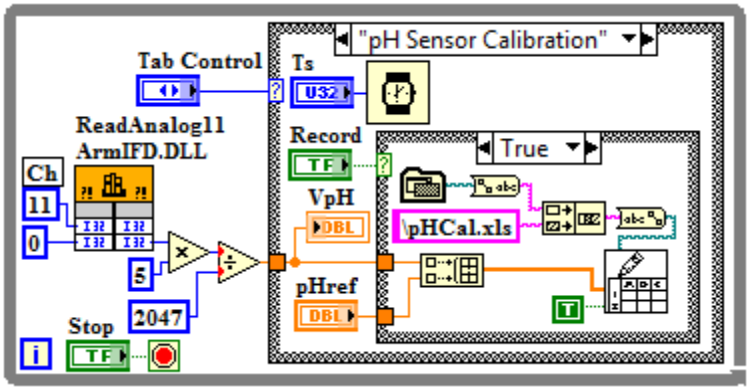


Figure 3.7(b) LabVIEW block diagram for data sampling of pH sensor

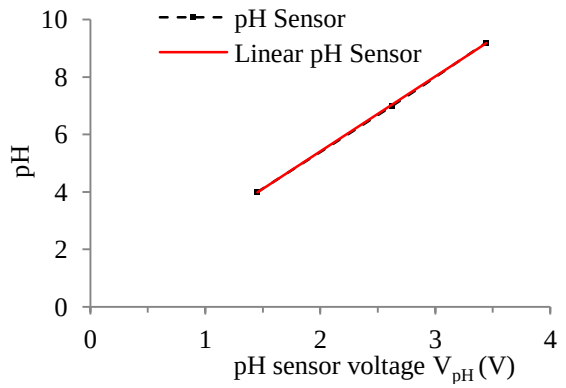


Figure 3.8(a) Static calibration for pH sensor

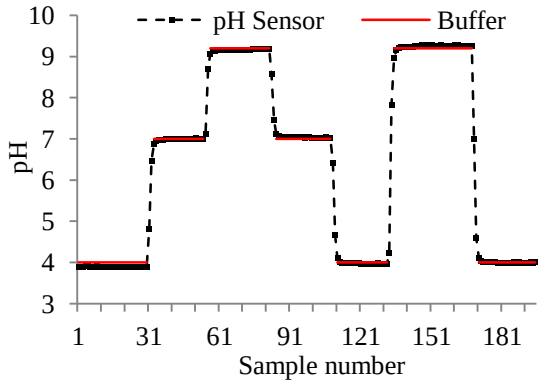


Figure 3.8(b) Dynamic response of pH sensor

Figure 3.8(a) is shows static calibration plot of pH sensor output voltage against standard pH buffer solutions. A linear regression analysis gives an estimate of pH based on V_{pH} as shown in equation (3.5), with statistical coefficient R^2 as 0.9998.

$$pH = 2.6114\ V_{pH} + 0.1868 \qquad (3.5)$$

Figure 3.8(b) shows the dynamic response of pH sensor. Dynamic response is carried out by transferring the pH sensor from one standard buffer solution to another and then stirring it. Calibration equation (3.5) can be used to estimate pH from V_{pH}. Approximately two second time is elapsed on transferring the pH sensor from one buffer to another and then stirring it and we can conclude from Figure 3.8(b) that there is negligible delay in pH sensor response. In this research work, we will consider entire process lag to be associated with its mixing dynamics.

3.5 Concluding Remarks

In this chapter necessary details about Armfield PCT42, PCT41 and PCT40 hardware and software which formed basic pH neutralization system on which experimentation work is carried out are provided. Also calibration procedure of differential pressure sensors, peristaltic pumps and pH sensor has been described for pH neutralization system. Data for calibration is obtained by interfacing LabVIEW with pH neutralization system, and calibration equations for pump flowrates and pH sensor have been developed, which will be used for dynamic modeling, identification and control of pH neutralization system.

CHAPTER 4

DYNAMIC MODELING OF pH NEUTRALIZATION PROCESS

4.1 Introduction

An understanding of the dynamic behavior of processes is important from both process design and process control perspectives. The process model can be utilized to analyze and study various issues concerning the dynamic response of process. Dynamic modeling of the process is invariably required in identification and control problems. Nonlinear processes such as pH neutralization are complex and require many assumptions to be made for deriving the first principles based mathematical model to emulate the process behavior. Moreover, mathematical modeling needs an understanding of physics and chemistry of process, and process parameters are often estimated based on empirical input-output data. Nonlinear process modeling using ANN overcomes limitations related to limited knowledge about the process dynamics through its implicit learning based on empirical input-output data. In addition, ANN has advantage of parallel computation, making it fast after learning is complete, which makes it an attractive proposition in dynamic process modeling. In this chapter, design of dynamic modeling based on first principles and ANN for Armfield® PCT42 in conjunction with PCT41 and PCT4 has been carried out, and implemented using MATLAB® software. Also, performances of both models are compared using experimental data collected on above laboratory model of pH neutralization process.

4.2 First Principles based Dynamic Modeling of pH Neutralization Process

pH is defined as the negative logarithm of hydrogen ion concentration. However material balances on hydrogen ion would be extremely difficult as dissociation of water and the resultant slight change in water concentration is to be accounted for. This is especially true if one is interested in almost neutral solutions, as often is the case in industries. McAvoy et al. (1972) presented a rigorous method of deriving dynamic equations for pH neutralization of NaOH with CH_3COOH in CSTR. The proposed method avoids the difficulty of making a direct balance on

hydrogen ion by making material balances on sodium and acetate ions, using acetic acid and water equilibrium relationships, and electroneutrality equations. The resulting dynamic pH equations are considerably complex as compared with previously proposed models. Step response testing on an experimental CSTR verified the accuracy of derived model.

Figure 4.1 shows a brief schematic diagram of Armfield pH neutralization system. Chapter 3 describes Armfield pH neutralization system, namely PCT42 in conjunction with PCT41 and PCT40, in detail. The pH neutralization process takes place in CSTR with perfect mixing and constant volume. The CSTR has two influent streams, HCl as 'titration stream' called feed A and NaOH as 'process stream' called feed B, and one 'effluent stream'. Preparation of acidic and basic solutions and calculation of their concentrations are given in Appendix A8. Table 4.1 summarizes specifications of Armfield pH neutralization system which are found to be comparable with those of McAvoy (1972) experimental set up. Both systems involve pH neutralization of single acid-base streams under stirred condition and, they have comparable volume and flowrates. One major difference is that Armfield pH neutralization system involves strong acid-strong base reaction whereas McAvoy (1972) experimental set up involves weak acid-strong base reaction. Partially this difference has been tried to be bridged using reduced acid-base concentrations. Therefore, McAvoy et al. (1972) strategy has been used for dynamic pH model development.

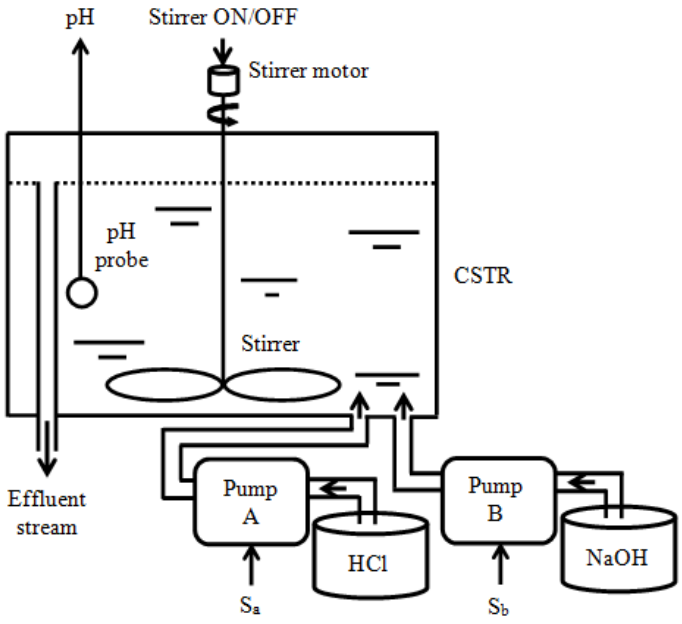


Figure 4.1 Schematic description of pH neutralization system

Table 4.1 pH neutralization system specifications

Parameter	Specification
Process vessel volume (V_s)	2000 mL
pH of raw water	6.7121
pH of HCl (pH_a)	1.75
Concentration of HCl (C_a)	0.01778 mol/L
pH of NaOH (pH_b)	12.1
Concentration of NaOH (C_b)	0.01259 mol/L
Useful range for speed of pump A (S_a) and speed of pump B (S_b)	18 to 100%
Equivalent flowrate of pump A (F_a)	0.2021 to 5.1139 mL/s
Equivalent flowrate of pump B (F_b)	0.2989 to 5.8749 mL/s
Voltage range of pH sensor	0 to 5 V
Equivalent pH reading	0.1868 to 13.2438
Sampling interval (T_s)	1 s

4.2.1 Mathematical Formulation of Dynamic pH Model

Dynamic model of pH neutralization process involves material balances on selective ions. The general equation for the conservation of material for pH process may be written as follows:

$$\begin{bmatrix} Rate\ of\ accumulation \\ of\ non-reactant \\ species\ within\ CSTR \end{bmatrix} = \begin{bmatrix} Rate\ of\ flow \\ of\ non-reactant \\ species\ into\ CSTR \end{bmatrix} - \begin{bmatrix} Rate\ of\ flow \\ of\ non-reactant \\ species\ out\ of\ CSTR \end{bmatrix}$$

Applying material balance on chloride ion $[Cl^-]$:

$$V_s \frac{dx_a}{dt} = F_a C_a - (F_a + F_b) x_a \qquad (4.1)$$

where x_a is the concentration (mol/L) of acid component $[Cl^-]$ in the effluent stream.

Applying material balance on sodium ion $[Na^+]$:

$$V_s \frac{dx_b}{dt} = F_b C_b - (F_a + F_b) x_b \quad (4.2)$$

where x_b is the concentration (mol/L) of base component $[Na^+]$ in the effluent stream.

The equilibrium relationship for water is:

$$K_w = [H^+][OH^-] \quad (4.3)$$

where K_w is the dissociation constant of water (10^{-14}).

Since the acid-base reaction is neutral in nature, from the electroneutrality condition,

$$[Na^+] + [H^+] = [Cl^-] + [OH^-] \quad (4.4)$$

Since all of the $[Cl^-]$ ion comes from the HCl and all of the $[Na^+]$ ion comes from the NaOH,

$$x_a = [Cl^-] \quad (4.5)$$

$$x_b = [Na^+] \quad (4.6)$$

Using Eq. (4.4), Eq. (4.5) and Eq. (4.6),

$$x_b + [H^+] = x_a + [OH^-] \quad (4.7)$$

Eliminating $[OH^-]$ from Eq. (4.7) and using Eq. (4.3),

$$[H^+]^2 - (x_a - x_b)[H^+] - K_w = 0 \quad (4.8)$$

Since $[H^+] = 10^{-pH}$ (4.9)

Substituting Eq. (4.9) in Eq. (4.8), we get

$$(10^{-pH})^2 - (x_a - x_b)\, 10^{-pH} - K_w = 0 \quad (4.10)$$

Solving Eq. (4.10), we get

$$10^{-pH} = \left(\frac{x}{2} + \sqrt{\frac{x^2}{4} + K_w} \right) \quad (4.11)$$

$$\Rightarrow \mathrm{pH} = -\log_{10}\left(\frac{\mathrm{x}}{2} + \sqrt{\frac{\mathrm{x}^2}{4} + \mathrm{K_w}}\right) \tag{4.12}$$

where $\mathrm{x} = (\mathrm{x_a} - \mathrm{x_b}) = 10^{-\mathrm{pH}} - 10^{\mathrm{pH}-14}$ (4.13)

Finally we have following observations about dynamic pH model:

If $\mathrm{x} = 0$ i.e. $\mathrm{x_a} = \mathrm{x_b}$, then $\mathrm{pH} = 7$ (neutral).

If $\mathrm{x} > 0$ i.e. $\mathrm{x_a} > \mathrm{x_b}$, then $\mathrm{pH} < 7$ (acidic).

If $\mathrm{x} < 0$ i.e. $\mathrm{x_a} < \mathrm{x_b}$, then $\mathrm{pH} > 7$ (basic).

4.2.2 Validation of Model Output with Experimental Results

Model validation is carried out to determine the accuracy with which the developed pH model is able to represent the actual neutralization process,. For validation purpose, we have chosen the neutral point pH = 7 with initial speeds of pump A and B as S_{a0} = 35% and S_{b0} = 38.5% respectively. The validation process involves following steps.

Step 1: The step responses of the simulated dynamic pH model and actual pH neutralization process are obtained for sampling duration of 300 seconds, by keeping pump A speed unchanged at S_{a0}, and applying a step change in pump B speed ΔS_b i.e. the pump B is speed is maintained at $S_{b0} + \Delta S_b$.

Step 2: The mean of squared errors (MSE) are calculated where error is defined as the difference between simulated model output minus experimentally obtained output.

Figure 4.2(a) shows the flowchart for step response of pH neutralization process, and Figure 4.2(b) shows the block diagram of LabVIEW implementation for the same. Figure 4.3 shows the various plots of simulated model output and experimental output for S_{a0} = 35%, S_{b0} = 38.5%, and ΔS_b = 41.5%, 31.5%, 21.5%, 11.5%, 1.5%, -3.5%, -8.5%, -18.5%. On the basis of observation of experimental responses, simulated dynamic model has been assigned a dead time of three sampling instants i.e. 3 second for all values of ΔS_b.

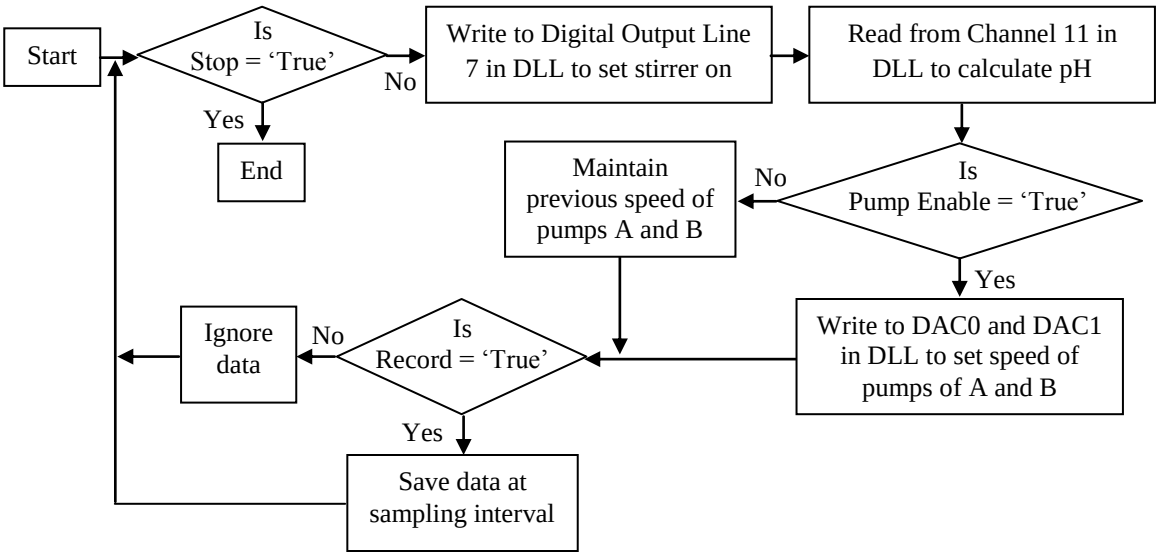


Figure 4.2(a) Flowchart for step response of pH neutralization system

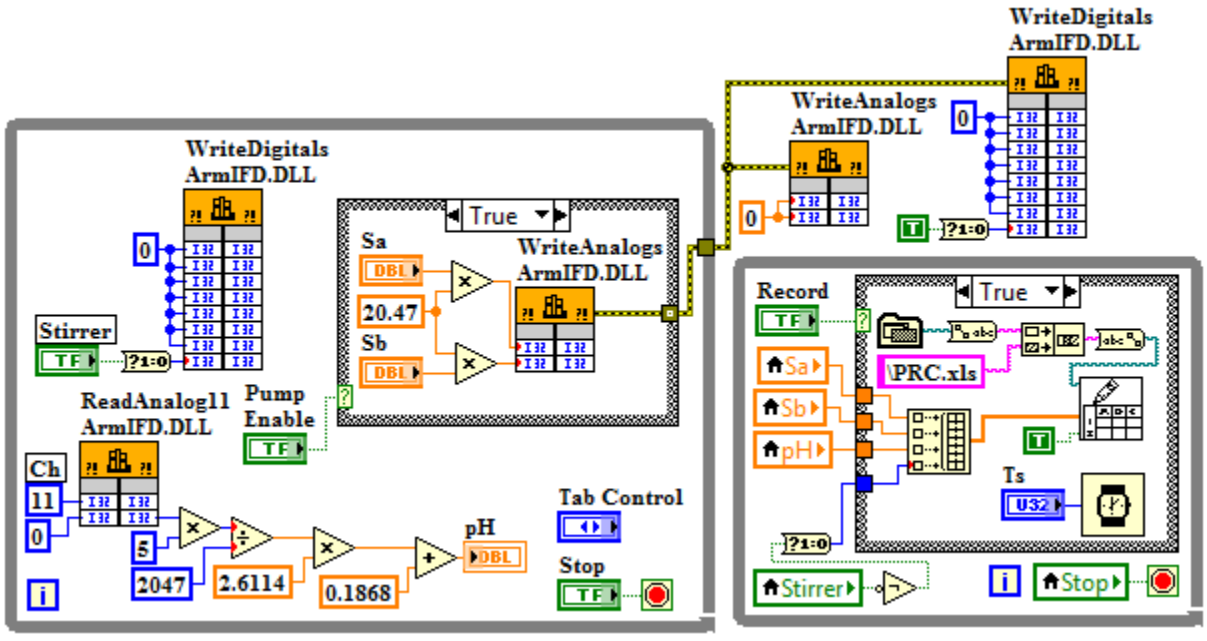


Figure 4.2(b) LabVIEW block diagram for step response of pH neutralization system

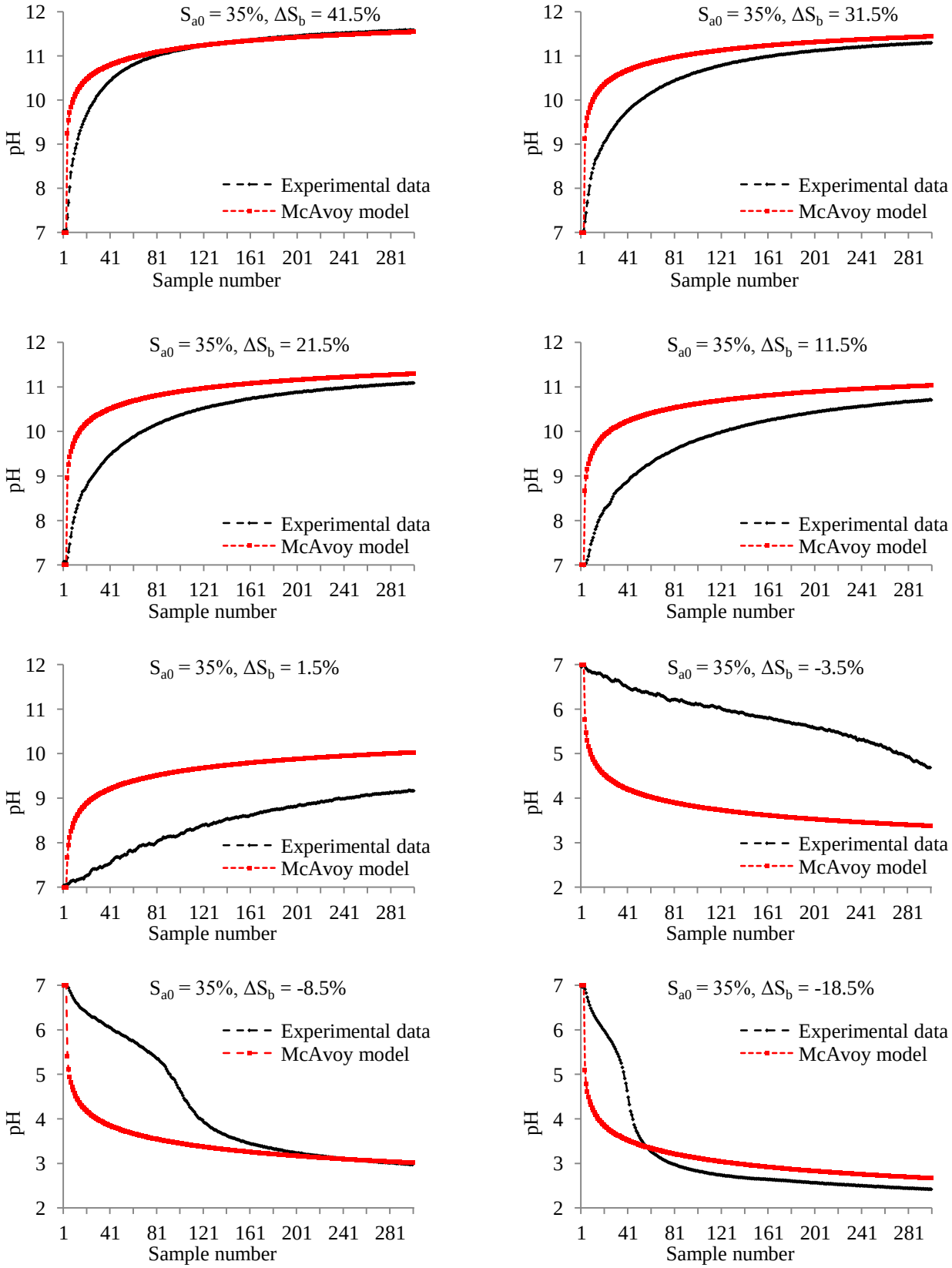


Figure 4.3 Step response of pH neutralization system for various speed values of pump B

Table 4.2 shows the performance of simulated pH model in terms of following factors: Initial error at sampling instant just after dead time, Final error at the end of sampling duration, Maximum error magnitude, Sampling instant for maximum error magnitude and MSE. Following conclusions can be drawn about the first principles based dynamic pH model.

(i) For ΔS_b = 41.5%, simulated model has initial error in pH of 1.90754 at 4^{th} sampling instant which is maximum, final error in pH of 0.03937 at 300^{th} sampling instant and MSE of 0.121253. For decreasing but positive values of ΔS_b, model error decreases at 4^{th} sampling instant, as sampling progresses model error decreases after reaching maximum, final error at 300^{th} sampling instant increases and MSE increases.

(ii) For ΔS_b = -18.5%, simulated model has initial error in pH of -1.81912 at 4^{th} sampling instant, minimum error in pH of -2.185088 at 11^{th} sampling instant, final error in pH of 0.25085 at 300^{th} sampling instant and MSE of 0.546655. For increasing but negative values of ΔS_b, model error increases at 4^{th} sampling instant, as sampling progresses model error increases after reaching minimum, final error at 300^{th} sampling instant decreases and MSE increases.

From validation results, it is evident that the McAvoy et al. (1972) based dynamic pH model does not represent the behavior of pH neutralization process. Therefore, it is proposed to use dynamic feedforward neural network for development of pH neutralization process model.

Table 4.2 Comparative performance of first principles based pH model at pH = 7

S_{a0} (%)	S_{b0} (%)	ΔS_b (%)	Dead-time (s)	Initial error in pH at 4^{th} sampling instant	Final error in pH at 300^{th} sampling instant	Magnitude of maximum error in pH	Sampling instant for maximum error	MSE for 300 samples
35	38.5	41.5	3	1.90754	0.03937	1.90754	4	0.121253
35	38.5	31.5	3	1.88196	0.1454	1.9783	5	0.34869
35	38.5	21.5	3	1.78968	0.20641	1.96263	5	0.440231
35	38.5	11.5	3	1.68844	0.32885	2.07862	7	0.745791
35	38.5	1.5	3	0.60141	0.862	1.68641	34	1.534779
35	38.5	-3.5	3	-1.19104	-1.30833	2.358881	68	4.277416
35	38.5	-8.5	3	-1.59576	0.04235	2.247414	28	1.364233
35	38.5	-18.5	3	-1.81912	0.25085	2.185088	11	0.546655

4.3 Artificial Neural Network (ANN) based Dynamic Modeling of pH Neutralization Process

4.3.1 Basics of ANN

ANN offer near ideal solutions to system identification modeling for complex and nonlinear where the physical processes are poorly understood and/or are highly complex. ANN is a simplified mathematical model based on the highly complex neural network of the human brain. Like the human brain, ANN is also designed to learn by example and past experience. ANN is composed of a large number of highly interconnected processing elements, called artificial neurons, working in unison to solve specific problems. ANN is configured for a specific application, such as function approximation, pattern recognition, data classification, etc. Learning in biological systems involves adjustments of the synaptic connections that exist between the neurons. This is true for ANN which adjusts weights of links between artificial neurons for learning.

The first artificial neuron was formulated in 1943 by neurophysiologist Warren McCulloch and logician Walter Pitts. Figure 4.4(a) shows a modern single input artificial neuron. The scalar input x is multiplied by the scalar weight w to form wx, one of the terms that is sent to the summer. The other input, 1, is multiplied by a bias b and then passed to the summer. The summer output n, often referred to as the net input, goes to an activation function f, which produces the scalar neuron output a. Typically the activation function is chosen by the user and the parameters w and b are adjusted by some learning rule so that the neuron input/output relationship meets some specific goal. Activation function, also known as transfer function, may be linear or nonlinear in nature.

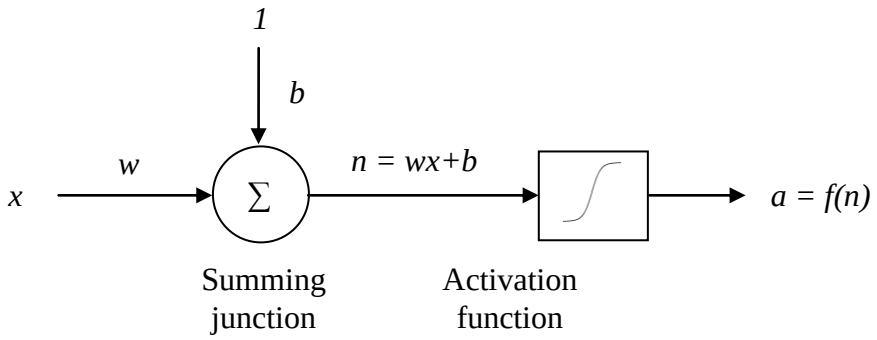


Figure 4.4(a) Single input artificial neuron

Among variety of activation functions, popular ones are hard limiter, linear, log-sigmoid and tan-sigmoid. In fact the first artificial neuron by McCulloch and Pitts used hard limiter as activation function. The sigmoid transfer function is widely accepted due to its continuously differentiable and monotonically increasing properties. A connection of neurons using this transfer function is capable of mapping highly nonlinear relations. In the present thesis, we have used tan-sigmoid activation function for all neurons. Figure 4.4(b) shows the tan-sigmoid activation function and Equation (4.14) gives its mathematic representation.

$$f(n) = \frac{1 - e^{-2n}}{1 + e^{-2n}} \tag{4.14}$$

Psychologist, Frank Rosenblatt, invented the first artificial neural network in 1958, called it the *perceptron model*. Figure 4.4(c) shows the representation of basic perceptron model which is equivalent to a multiple input, single artificial neuron. The individual inputs x_1, x_2, x_3 are each weighted by corresponding weights $w_{1,1}$, $w_{1,2}$, $w_{1,3}$ respectively. The neuron has a bias b. Mathematically, variable n in vector form is represented as shown in Equation (4.15).

$$n = [w_{1,1} \quad w_{1,2} \quad w_{1,3}] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + b \tag{4.15}$$

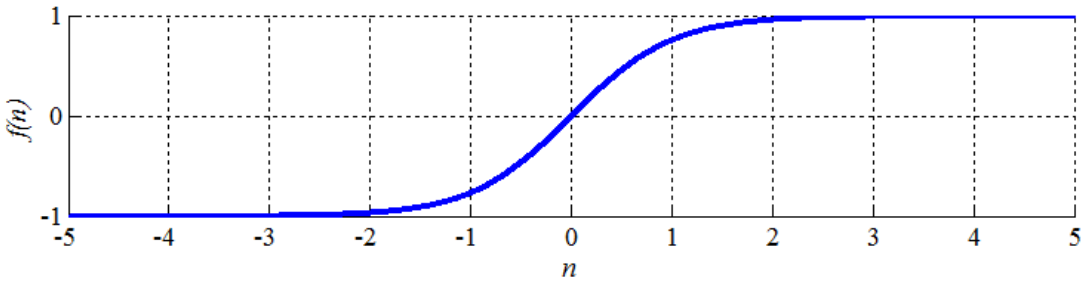


Figure 4.4(b) Tan-sigmoid activation function

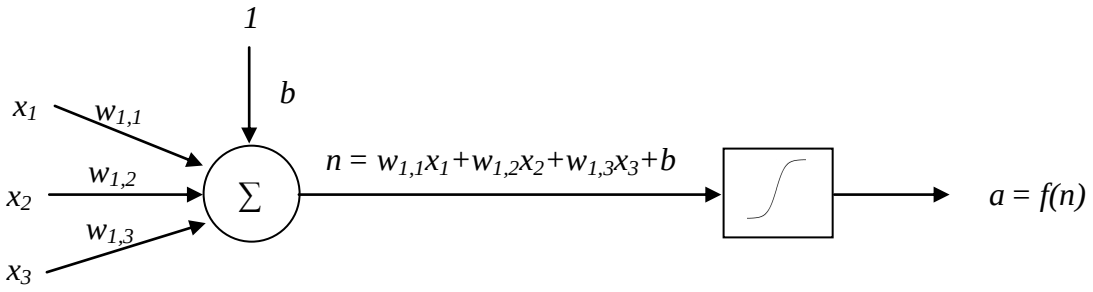


Figure 4.4(c) Single neuron perceptron model of ANN

Figure 4.4(d) shows a single layer perceptron network with multiple inputs and multiple neurons. It can be noted that each of the three inputs is connected to each of the neurons. Mathematically, we can represent net input in matrix form as shown in Equation (4.16). Additionally the network input-output can be expressed in vector form as shown in Equations (4.17) and (4.18). A single layer perceptron is not very useful because of its limited mapping ability. In general, we have multilayer perceptron network consisting of one or more hidden layers and an output layer of computational neurons. Figure 4.4(e) shows a multilayer perceptron model for multiple inputs with various hidden layers and an output layer.

$$\begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ w_{2,1} & w_{2,2} & w_{2,3} \\ w_{3,1} & w_{3,2} & w_{3,3} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{4.16}$$

$$\mathbf{n} = \mathbf{WX} + \mathbf{b} \tag{4.17}$$

$$\mathbf{a} = f(\mathbf{n}) \tag{4.18}$$

where $\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$, $\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ w_{2,1} & w_{2,2} & w_{2,3} \\ w_{3,1} & w_{3,2} & w_{3,3} \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$, $\mathbf{n} = \begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix}$, $\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$

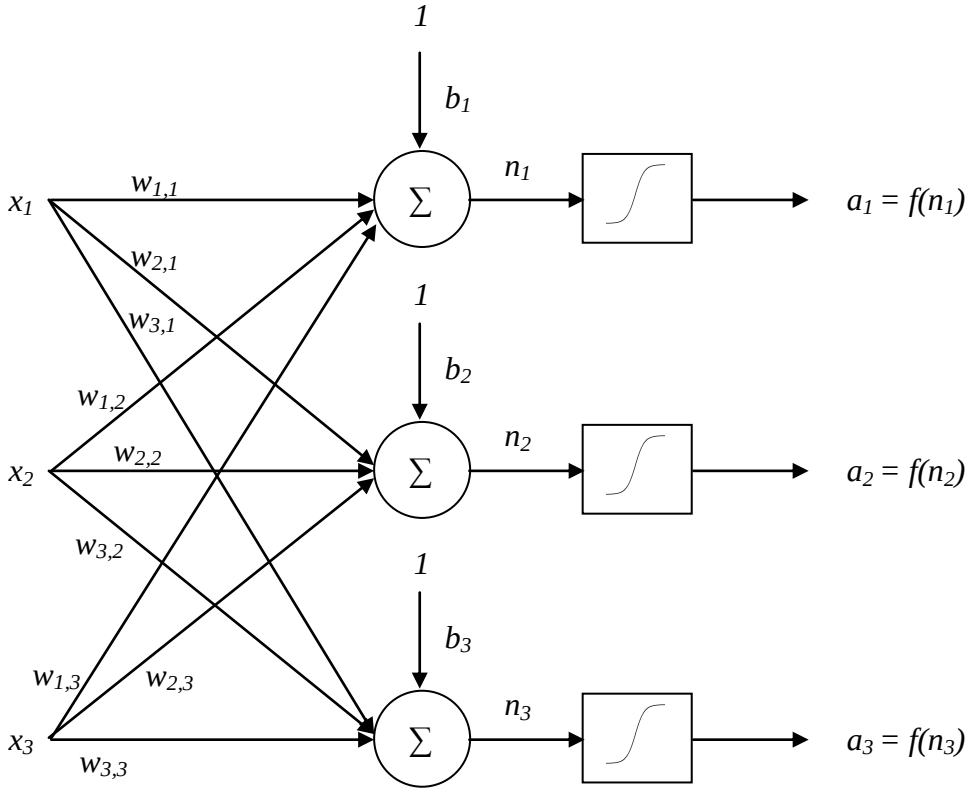


Figure 4.4(d) Multiple neuron perceptron model of ANN

ANNs are broadly classified based on their architecture and learning rules. According to architecture, ANNs are classified into two main categories: Static network and Dynamic network. Static networks have no feedback elements and contain no previous inputs/outputs, thus allowing signals to travel only in non-recurrent, feedforward manner. Static network output can be directly calculated from the input through feedforward connections. In dynamic networks, the output depends not only on the current inputs to the network, but also on the previous input and output states of the have network. Therefore dynamic networks can be further divided into two sub-categories: those that have only feedforward connections such as tapped delay line (TDL) method based architecture, and those that have feedback connections such as recurrent networks. The TDL approach uses the current and past values of the system inputs and outputs as the inputs to a standard feedforward network, and the network is trained to predict the next value of the system output. The recurrent feedback approach employs a recurrent neural network which introduces dynamics into the architecture of the network by feeding back the output of some or all of the neurons, via a weighted connection to the inputs of some or all of the neurons.

Learning rules or training algorithms in ANNs implement an adaptive procedure for modifying its weights and biases, and resulting neural network is said to be trained in order to perform the desired task. According to learning rules, ANNs are classified into three broad categories: Supervised learning, Reinforcement learning and Unsupervised learning. In supervised learning, the neural network is provided with a input-target output training data set and the network learns by comparing the actual output with the target or desired output value for given inputs. Similar to supervised learning, reinforcement learning too receives feedback from its environment, except that the information received is not exact but critic in nature such as 'yes' or 'no'. In unsupervised learning, the network is not provided with target values and the network learns by performing clustering operating on the input data set.

Supervised learning rules operate in either incremental mode or batch mode. In incremental mode, each input-output pair from training data set is sequentially presented to the network, and network weights and biases are updated in sequential manner based on corresponding training error. Once all input-output pairs are presented to the neural network, it is said to have completed an *epoch*. In batch mode, entire training data set are presented to network, and network weights and biases are updated based on total training error obtained on presenting the entire data set.

In this thesis, dynamic feedforward neural network architecture based on tapped delay line method and supervised learning rule in batch mode has been used.

4.3.2 Basics of Back-Propagation Learning Algorithms in ANN

The back-propagation (BP) learning algorithm, developed by Rumelhart et al. (1985), is the most popular supervised training algorithm used in order to train a feedforward neural network and it provided a major breakthrough for the neural network research. The basic algorithm used gradient-descent or steepest-descent method with constant learning rate. However, the basic algorithm was too slow for most practical applications. Thus, over a number of years, researchers have presented several variations of BP algorithm that provided significant speed-up and make the algorithm more practical. In this section, the basic BP algorithm and its important variants in batch mode for feedforward or multilayer perceptron architecture of ANN as shown in Figure 4.4(e) is described (Hagan et al., 2014).

4.3.2.1 Gradient-Descent method with constant learning rate (GD)

The basic BP algorithm in batch mode based on GD can be outlined using following steps.

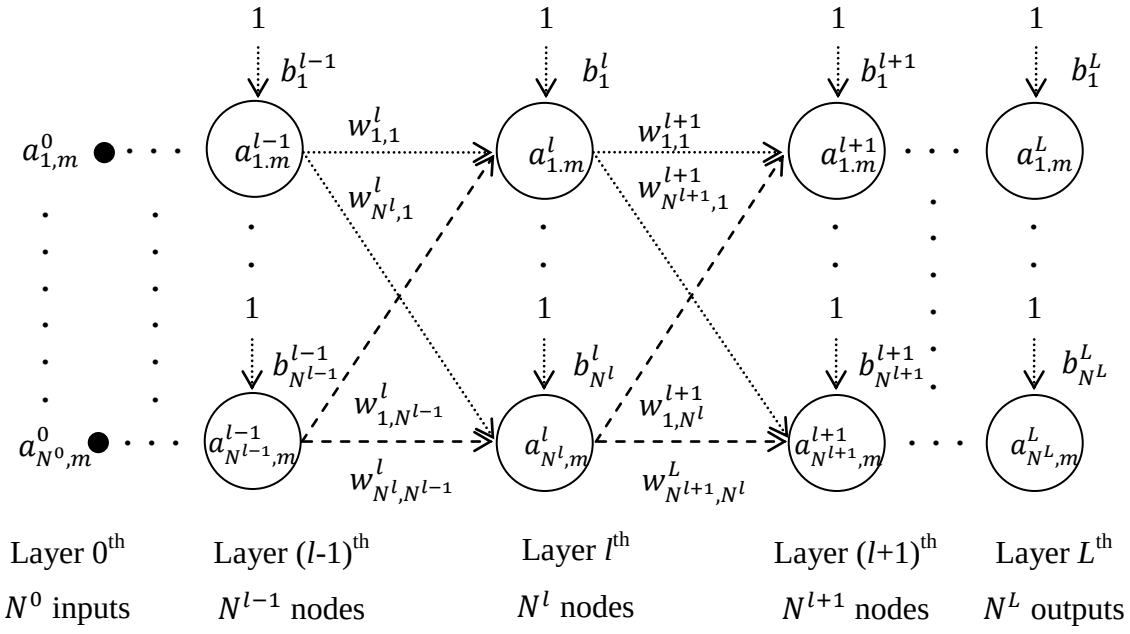


Figure 4.4(e) Feedforward or multilayer perceptron architecture of ANN

Step 1. Initialize all weights $w_{i,j}^{l}$ and biases b_i^l for layer l = 1 to L with random values within input range.

Step 2. Initialize current epoch as k = 1.

Step 3. Calculate the output $a_{i,m}^{l}$ for all neurons in layer l = 1 to L for all M training data set. All neurons have tan-sigmoid activation function as shown in equation (4.14).

$$n_{i,m}^{l} = \sum_{j=1}^{N^{l-1}} \left(a_{j,m}^{l-1} w_{i,j}^{l}\right) + b_i^l \tag{4.19}$$

$$a_{i,m}^{l} = f\left(n_{i,m}^{l}\right) \tag{4.20}$$

Step 4. Compute error at all neurons at the output layer $l = L$ for all M training data set.

$$e_{i,m}^{L} = t_i^L - a_{i,m}^{L} \tag{4.21}$$

Step 5. The BP algorithm minimizes performance index defined as half of MSE for N^L neurons at output layer L for all M training data set.

$$E_T^L = \frac{1}{2M} \sum_{m=1}^{M} \sum_{i=1}^{N^L} \left(e_{i,m}^{L}\right)^2 \tag{4.22}$$

According to GD based BP algorithm, the weights and biases in layer $l = L$ are updated as follows:

$$w_{i,j}^{L}(k+1) = w_{i,j}^{L}(k) - \alpha \frac{\partial E_T^L}{\partial w_{i,j}^{L}} \tag{4.23}$$

$$b_i^L(k+1) = b_i^L(k) - \alpha \frac{\partial E_T^L}{\partial b_i^L} \tag{4.24}$$

where i = 1 to N^L and j = 1 to N^{L-1}.

Using equation (4.22), we have components of error gradient at the output layer $l = L$ as follows.

$$\frac{\partial E_T^L}{\partial w_{i,j}^L} = \frac{1}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(e_{i,m}^L\left(\frac{\partial e_{i,m}^L}{\partial w_{i,j}^L}\right)\right) \tag{4.25}$$

$$\frac{\partial E_T^L}{\partial b_i^L} = \frac{1}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(e_{i,m}^L\left(\frac{\partial e_{i,m}^L}{\partial b_i^L}\right)\right) \tag{4.26}$$

Using equation (4.21), we have

$$\frac{\partial e_{i,m}^L}{\partial w_{i,j}^L} = -\frac{\partial a_{i,m}^L}{\partial w_{i,j}^L} \tag{4.27}$$

$$\frac{\partial e_{i,m}^L}{\partial b_i^L} = -\frac{\partial a_{i,m}^L}{\partial b_i^L} \tag{4.28}$$

Using chain rule, we can write

$$\frac{\partial a_{i,m}^L}{\partial w_{i,j}^L} = \left(\frac{\partial a_{i,m}^L}{\partial n_{i,m}^L}\right)\left(\frac{\partial n_{i,m}^L}{\partial w_{i,j}^L}\right) \tag{4.29}$$

$$\frac{\partial a_{i,m}^L}{\partial b_i^L} = \left(\frac{\partial a_{i,m}^L}{\partial n_{i,m}^L}\right)\left(\frac{\partial n_{i,m}^L}{\partial b_i^L}\right) \tag{4.30}$$

Using equations (4.14) and (4.20), we have

$$\frac{\partial a_{i,m}^L}{\partial n_{i,m}^L} = 1 - \left(a_{i,m}^L\right)^2 \tag{4.31}$$

Using equation (4.19), we have

$$\frac{\partial n_{i,m}^L}{\partial w_{i,j}^L} = a_{j,m}^{L-1} \tag{4.32}$$

$$\frac{\partial n_{i,m}^L}{\partial b_i^L} = 1 \tag{4.33}$$

Using equations (4.23), (4.25), (4.27), (4.29), (4.31) and (4.32), we have

$$w_{i,j}^{L}(k+1) = w_{i,j}^{L}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(e_{i,m}^{L}\left(1-\left(a_{i,m}^{L}\right)^2\right)a_{j,m}^{L-1}\right) \tag{4.34}$$

Using equations (4.24), (4.26), (4.28), (4.30), (4.31) and (4.33), we have

$$b_{i}^{L}(k+1) = b_{i}^{L}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(e_{i,m}^{L}\left(1-\left(a_{i,m}^{L}\right)^2\right)\right) \tag{4.35}$$

Using equations (4.21) and (4.34), we have

$$w_{i,j}^{L}(k+1) = w_{i,j}^{L}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(\left(t_{i}^{L}-a_{i,m}^{L}\right)\left(1-\left(a_{i,m}^{L}\right)^2\right)a_{j,m}^{L-1}\right) \tag{4.36}$$

Using equations (4.21) and (4.35), we have

$$b_{i}^{L}(k+1) = b_{i}^{L}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(\left(t_{i}^{L}-a_{i,m}^{L}\right)\left(1-\left(a_{i,m}^{L}\right)^2\right)\right) \tag{4.37}$$

Using equations (4.36) and (4.37), we have updated weights and biases in terms of error sensitivity $S_{i,m}^{L}$ as follows.

$$w_{i,j}^{L}(k+1) = w_{i,j}^{L}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(S_{i,m}^{L}a_{j,m}^{L-1}\right) \tag{4.38}$$

$$b_{i}^{L}(k+1) = b_{i}^{L}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(S_{i,m}^{L}\right) \tag{4.39}$$

$$S_{i,m}^{L} = \left(t_{i}^{L}-a_{i,m}^{L}\right)\left(1-\left(a_{i,m}^{L}\right)^2\right) \tag{4.40}$$

Step 6. According to GD based BP algorithm, the weights and biases in layer $l = L$-1 are updated as follows:

$$w_{i,j}^{L-1}(k+1) = w_{i,j}^{L-1}(k) - \alpha\frac{\partial E_{T}^{L}}{\partial w_{i,j}^{L-1}} \tag{4.41}$$

$$b_i^{L-1}(k+1) = b_i^{L-1}(k) - \alpha \frac{\partial E_T^L}{\partial b_i^{L-1}} \tag{4.42}$$

where i = 1 to $N^{L-1}$and j = 1 to N^{L-2}.

Using equation (4.22), we have

$$\frac{\partial E_T^L}{\partial w_{i,j}^{L-1}} = \frac{1}{M} \sum_{m=1}^{M} \sum_{i=1}^{N^{L-1}} \left(e_{i,m}^L \left(\frac{\partial e_{i,m}^L}{\partial w_{i,j}^{L-1}} \right) \right) \tag{4.43}$$

$$\frac{\partial E_T^L}{\partial b_i^{L-1}} = \frac{1}{M} \sum_{m=1}^{M} \sum_{i=1}^{N^{L-1}} \left(e_{i,m}^L \left(\frac{\partial e_{i,m}^L}{\partial b_i^{L-1}} \right) \right) \tag{4.44}$$

Using equation (4.21), we have

$$\frac{\partial e_{i,m}^L}{\partial w_{i,j}^{L-1}} = -\frac{\partial a_{i,m}^L}{\partial w_{i,j}^{L-1}} \tag{4.45}$$

$$\frac{\partial e_{i,m}^L}{\partial b_i^{L-1}} = -\frac{\partial a_{i,m}^L}{\partial b_i^{L-1}} \tag{4.46}$$

Using chain rule, we have

$$\frac{\partial a_{i,m}^L}{\partial w_{i,j}^{L-1}} = \left(\frac{\partial a_{i,m}^L}{\partial n_{i,m}^L} \right) \left(\frac{\partial n_{i,m}^L}{\partial a_{i,m}^{L-1}} \right) \left(\frac{\partial a_{i,m}^{L-1}}{\partial n_{i,m}^{L-1}} \right) \left(\frac{\partial n_{i,m}^{L-1}}{\partial w_{i,j}^{L-1}} \right) \tag{4.47}$$

$$\frac{\partial a_{i,m}^L}{\partial b_i^{L-1}} = \left(\frac{\partial a_{i,m}^L}{\partial n_{i,m}^L} \right) \left(\frac{\partial n_{i,m}^L}{\partial a_{i,m}^{L-1}} \right) \left(\frac{\partial a_{i,m}^{L-1}}{\partial n_{i,m}^{L-1}} \right) \left(\frac{\partial n_{i,m}^{L-1}}{\partial b_i^{L-1}} \right) \tag{4.48}$$

Using equations (4.14) and (4.20), we have

$$\frac{\partial a_{i,m}^{L-1}}{\partial n_{i,m}^{L-1}} = 1 - \left(a_{i,m}^{L-1} \right)^2 \tag{4.49}$$

Using equation (4.19), we have

$$\frac{\partial n_{i,m}^L}{\partial a_{i,m}^{L-1}} = w_{i,j}^L \tag{4.50}$$

$$\frac{\partial n_{i,m}^{L-1}}{\partial w_{i,j}^{L-1}} = a_{j,m}^{L-2} \tag{4.51}$$

$$\frac{\partial n_{i,m}^{L-1}}{\partial b_{i}^{L-1}} = 1 \tag{4.52}$$

Using equations (4.31), (4.41), (4.43), (4.45), (4.47), (4.49), (4.50) and (4.51), we have

$$w_{i,j}^{L-1}(k+1) = w_{i,j}^{L-1}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^{L-1}}\left(e_{i,m}^{L}\left(1-\left(a_{i,m}^{L}\right)^2\right)\left(1-\left(a_{i,m}^{L-1}\right)^2\right)a_{j,m}^{L-2}w_{i,j}^{L}\right) \tag{4.53}$$

Using equations (4.31), (4.42), (4.44), (4.46), (4.48), (4.49), (4.50) and (4.52), we have

$$b_{i}^{L-1}(k+1) = b_{i}^{L-1}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^{L-1}}\left(e_{i,m}^{L}\left(1-\left(a_{i,m}^{L}\right)^2\right)\left(1-\left(a_{i,m}^{L-1}\right)^2\right)w_{i,j}^{L}\right) \tag{4.54}$$

Using equations (4.21) and (4.53), we have

$$w_{i,j}^{L-1}(k+1) = w_{i,j}^{L-1}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^{L-1}}\left(\left(t_{i}^{L}-a_{i,m}^{L}\right)\left(1-\left(a_{i,m}^{L}\right)^2\right)\left(1-\left(a_{i,m}^{L-1}\right)^2\right)a_{j,m}^{L-2}w_{i,j}^{L}\right) \tag{4.55}$$

Using equations (4.21) and (4.54), we have

$$b_{i}^{L-1}(k+1) = b_{i}^{L-1}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^{L-1}}\left(\left(t_{i}^{L}-a_{i,m}^{L}\right)\left(1-\left(a_{i,m}^{L}\right)^2\right)\left(1-\left(a_{i,m}^{L-1}\right)^2\right)w_{i,j}^{L}\right) \tag{4.56}$$

Using equations (4.55) and (4.56), we have updated weights and biases in terms of error sensitivity $S_{i,m}^{L-1}$ as follows.

$$w_{i,j}^{L-1}(k+1) = w_{i,j}^{L-1}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^{L-1}}\left(S_{i,m}^{L-1}a_{j,m}^{L-2}\right) \tag{4.57}$$

$$b_{i}^{L-1}(k+1) = b_{i}^{L-1}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^{L-1}}\left(S_{i,m}^{L-1}\right) \tag{4.58}$$

$$S_{i,m}^{L-1} = S_{i,m}^{L}\left(1 - \left(a_{i,m}^{L-1}\right)^2\right) w_{i,j}^{L} \tag{4.59}$$

Step 7. In general, according to GD based BP algorithm, the weights and biases in layer $l = L$-1 to 1 are updated as follows:

$$w_{i,j}^{l}(k+1) = w_{i,j}^{l}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^l}\left(S_{i,m}^{l} a_{j,m}^{l-1}\right) \tag{4.60}$$

$$b_{i}^{l}(k+1) = b_{i}^{l}(k) + \frac{\alpha}{M}\sum_{m=1}^{M}\sum_{i=1}^{N^l}\left(S_{i,m}^{l}\right) \tag{4.61}$$

$$S_{i,m}^{l} = S_{i,m}^{l+1}\left(1 - \left(a_{mi}^{l}\right)^2\right) w_{i,j}^{l+1} \tag{4.62}$$

Step 8. Check the stopping criteria such as maximum number of epoch, minimum error performance function and minimum gradient. If stopping criteria is satisfied, end the training process. If stopping criteria is not satisfied, perform step 9.

Step 9. Perform steps 3 to 8 for next epoch.

4.3.2.2 Gradient-Descent method with constant learning rate and Momentum (GDM)

The GD based BP algorithm is very slow if the learning rate is small and oscillates widely if the learning rate is too large. One very efficient and commonly used method that allows a larger learning rate without oscillation is by adding a momentum factor (γ) to the basic gradient-descent method. According to GDM based BP algorithm, the weights and biases of layer L are updated as shown in equations (4.63) and (4.64) respectively, and the weights and biases of layer $l = L$-1 to 1 are updated as shown in equations (4.65) and (4.66) respectively.

$$w_{i,j}^{L}(k+1) = w_{i,j}^{L}(k) + \frac{\alpha}{M}(1-\gamma)\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(S_{i,m}^{L} a_{j,m}^{L-1}\right) + \gamma\left(w_{i,j}^{L}(k) - w_{i,j}^{L}(k-1)\right) \tag{4.63}$$

$$b_{i}^{L}(k+1) = b_{i}^{L}(k) + \frac{\alpha}{M}(1-\gamma)\sum_{m=1}^{M}\sum_{i=1}^{N^L}\left(S_{i,m}^{L}\right) + \gamma\left(b_{i}^{L}(k) - b_{i}^{L}(k-1)\right) \tag{4.64}$$

$$w_{i,j}^{l}(k+1) = w_{i,j}^{l}(k) + \frac{\alpha}{M}(1-\gamma)\sum_{m=1}^{M}\sum_{i=1}^{N^l}\left(S_{i,m}^{l}a_{j,m}^{l-1}\right) + \gamma\left(w_{i,j}^{l}(k) - w_{i,j}^{l}(k-1)\right) \quad (4.65)$$

$$b_{i}^{l}(k+1) = b_{i}^{l}(k) + \frac{\alpha}{M}(1-\gamma)\sum_{m=1}^{M}\sum_{i=1}^{N^l}\left(S_{i,m}^{l}\right) + \gamma\left(b_{i}^{l}(k) - b_{i}^{l}(k-1)\right) \quad (4.66)$$

where $S_{i,m}^{L}$ and $S_{i,m}^{l}$ are given by equations (4.40) and (4.62).

From equations (4.63) and (4.64), it is clear that momentum term takes into account the effect of past changes also. Addition of momentum factor also accelerates convergence of gradient-descent based BP algorithm. Typical value of momentum factor lies between 0 and 1. When the momentum factor is 0, the weights and biases are updated using gradient-descent based BP algorithm. When the momentum factor is 1, the new updates in weights and biases is same as the last update since gradient term contribution becomes zero.

4.3.2.3 Gradient-Descent method with Adaptive learning rate (GDA)

In order to speed up convergence of basic BP algorithm, the learning rate is adjusted during the course of training. An important aspect is, however, to determine when to change the learning rate and by how much. The rules to incorporate adaptive learning rate in gradient-descent based BP algorithm after weights and biases are updated in k^{th} epoch, according to equations (4.38), (4.39), (4.40), (4.60), (4.61) and (4.62), are as follows.

Rule 1. If the performance index value, $E_T^L(k+1)$, computed using (k+1)th epoch weights and biases, is greater than $E_T^L(k)$, and $E_T^L(k+1)$ is less than maximum performance increase (δ) times $E_T^L(k)$, then the weights and biases update in k^{th} epoch is accepted but the learning rate is unchanged.

Rule 2. If $E_T^L(k+1)$ is greater than $E_T^L(k)$, and $E_T^L(k+1)$ is greater than δ times $E_T^L(k)$, then the weights and biases update in k^{th} epoch is discarded and the learning rate is multiplied by learning rate decrement factor (α_d).

Rule 3. If $E_T^L(k+1)$ is less than $E_T^L(k)$, then the weights and biases update in k^{th} epoch is accepted and the learning rate is multiplied by learning rate increment factor (α_i).

4.3.2.4 Gradient-Descent method with Adaptive learning rate and Momentum (GDAM)

The rules to incorporate adaptive learning rate in gradient-descent based BP algorithm with momentum after weights and biases are updated in k^{th} epoch, according to equations (4.63), (4.64), (4.40), (4.65), (4.66) and (4.62), are as follows.

Rule 1. If $E_T^L(k+1)$ is greater than $E_T^L(k)$, and $E_T^L(k+1)$ is less than δ times $E_T^L(k)$, then the weights and biases update in k^{th} epoch is accepted but the learning rate is unchanged. If γ has been previously set to zero, it is reset to its original value.

Rule 2. If $E_T^L(k+1)$ is greater than $E_T^L(k)$, and $E_T^L(k+1)$ is greater than δ times $E_T^L(k)$, then the weights and biases update in k^{th} epoch is discarded and the learning rate is multiplied by α_d. If γ has been previously set to non-zero value, it is set to zero value.

Rule 3. If $E_T^L(k+1)$ is less than $E_T^L(k)$, then the weights and biases update in k^{th} epoch is accepted and the learning rate is multiplied by α_i. If γ has been previously set to zero, it is reset to its original value.

4.3.2.5 Levenberg-Marquardt algorithm (LM)

Gradient-descent method based BP algorithm and its variants, such as GD, GDM, GDA and GDAM, are essentially first-order ANN learning algorithms. In these learning algorithms, the weights and biases update is proportional to the error gradients i.e. the first-order partial derivative of the error function. These algorithms tend to be slow and often get trapped in local minima. Therefore, second-order algorithms, such as Newton method, have been developed which calculate weights and biases update on the basis of Hessian matrix (second-order partial derivatives of error function). These make the algorithm computationally expensive and therefore slow, especially for networks involving large number of weights and biases. LM algorithm like the quasi-Newton method is designed to approach second-order training speed without having to compute Hessian matrix (Hagan & Menhaj, 1994).

The LM algorithm in batch mode can be outlined using following steps.

Step 1. Initialize all weights $w_{i,j}^{l}$ and biases b_{i}^{l} for layer l = 1 to L with random values within input range.

Step 2. Initialize current epoch as k = 1.

Step 3. Calculate the output $a_{i,m}^{l}$ for all neurons in layer l = 1 to L for all M training data set, as shown in equations (4.19) and (4.20) respectively. All neurons have tan-sigmoid activation function as shown in equation (4.14).

Step 4. Compute error at all neurons at the output layer $l = L$ for all M training data set, as shown in equation (4.21).

Step 5. The LM algorithm minimizes performance index defined as half of MSE for N^L neurons at output layer L for all M training data set, as shown in equation (4.22). Equation (4.22) can be rewritten as in equation (4.67) and (4.68). Equation (4.70) represents the transpose of error vector $\mathbf{e^L(x)}$ at output layer. Equation (4.71) represents the transpose of parameter vector $\mathbf{x}$ consisting of all layer weights and biases.

$$E_T^L = \frac{1}{2M}\Big(\left(e_{1,1}^L\right)^2 + \left(e_{2,1}^L\right)^2 + \cdots + \left(e_{N^L,1}^L\right)^2 + \left(e_{1,2}^L\right)^2 + \left(e_{2,2}^L\right)^2 + \cdots + \left(e_{N^L,2}^L\right)^2 + \left(e_{1,3}^L\right)^2 + \cdots + \left(e_{N^L,(M-1)}^L\right)^2 + \left(e_{1,M}^L\right)^2 + \left(e_{2,M}^L\right)^2 + \cdots + \left(e_{N^L,M}^L\right)^2\Big) \tag{4.67}$$

$$E_T^L = \frac{1}{2M}\sum_{i=1}^{R}(v_i)^2 \tag{4.68}$$

$$R = N^L \times M \tag{4.69}$$

$$\left(\mathbf{e^L(x)}\right)^{\boldsymbol{T}} = \left[v_1\ v_2\ \ldots v_{N^L}\ v_{N^L+1}\ v_{N^L+2} \ldots v_{2N^L}\ v_{2N^L+1} \ldots v_{(M-1)N^L}\ v_{(M-1)N^L+1}\ v_{(M-1)N^L+2} \ldots v_R\right]$$

$$= \left[e_{1,1}^L\ e_{2,1}^L\ \ldots\ e_{N^L,1}^L\ e_{1,2}^L\ e_{2,2}^L \ldots\ e_{N^L,2}^L\ e_{1,3}^L \ldots\ e_{N^L,(M-1)}^L\ e_{1,M}^L\ e_{2,M}^L \ldots\ e_{N^L,M}^L\right] \tag{4.70}$$

$$\mathbf{x^T} = \left[x_1\ x_2\ \ldots\ x_{N^1\times N^0}\ x_{N^1\times N^0+1} \ldots x_{(N^0+1)\times N^1}\ x_{(N^0+1)\times N^1+1} \ldots\ x_{r-N^L}\ x_{r-N^L+1}\ \ldots x_r\right]$$

$$= \left[w_{1,1}^1\ w_{1,2}^1\ \ldots\ w_{N^1,N^0}^1\ b_1^1\ \ldots\ b_{N^1}^1\ w_{1,1}^2 \ldots\ w_{N^L,N^{L-1}}^L\ b_1^L \ldots\ b_{N^L}^L\right] \tag{4.71}$$

$$r = N^1 \times (N^0 + 1) + N^2 \times (N^1 + 1) + \cdots + N^L \times (N^{L-1} + 1) \tag{4.72}$$

According to LM algorithm, the parameter vector in k^{th} epoch are updated as follows:

$$\mathbf{x}(k+1) = \mathbf{x}(k) - \left(\mathbf{J}^{\boldsymbol{T}}(\mathbf{x}(k))\,\mathbf{J}(\mathbf{x}(k)) + \mu(k)\,\mathbf{I}\right)^{-1}\mathbf{J}^{\boldsymbol{T}}(\mathbf{x}(k))\,\mathbf{e}^{\boldsymbol{L}}(\mathbf{x}(k)) \tag{4.73}$$

$$\mathbf{J}(\mathbf{x}) = \begin{bmatrix} \frac{\partial v_1}{\partial x_1} & \frac{\partial v_1}{\partial x_2} & \cdots & \frac{\partial v_1}{\partial x_r} \\ \frac{\partial v_2}{\partial x_1} & \frac{\partial v_2}{\partial x_2} & \cdots & \frac{\partial v_2}{\partial x_r} \\ \vdots & \vdots & \vdots\ \vdots\ \vdots & \vdots \\ \frac{\partial v_R}{\partial x_1} & \frac{\partial v_R}{\partial x_2} & \cdots & \frac{\partial v_R}{\partial x_r} \end{bmatrix} = \begin{bmatrix} \frac{\partial e^L_{1,1}}{\partial w^1_{1,1}} & \frac{\partial e^L_{1,1}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e^L_{1,1}}{\partial b^L_{N^L}} \\ \frac{\partial e^L_{2,1}}{\partial w^1_{1,1}} & \frac{\partial e^L_{2,1}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e^L_{2,1}}{\partial b^L_{N^L}} \\ \vdots & \vdots & \vdots\ \vdots\ \vdots & \vdots \\ \frac{\partial e^L_{N^L,M}}{\partial w^1_{1,1}} & \frac{\partial e^L_{N^L,M}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e^L_{N^L,M}}{\partial b^L_{N^L}} \end{bmatrix} \tag{4.74}$$

The key step in LM algorithm is the computation of the Jacobian matrix as defined in equation (4.74). The elements of Jacobian matrix are expressed as

$$[\mathbf{J}(\mathbf{x})]_{u,c} = \frac{\partial v_u}{\partial x_c} = \frac{\partial e^L_{z,m}}{\partial x_c} \tag{4.75}$$

$$u = N^L \times (m-1) + z \tag{4.76}$$

where u = 1 to R, c = 1 to r, m = 1 to M, z = 1 to N^L.

If parameter x_c represents weights $w^l_{i,j}$ where i = 1 to N^l, j = 1 to N^{l-1}, $l = L$, then, using equations (4.19), (4.20) and (4.21), we have

$$\frac{\partial e^L_{z,m}}{\partial w^L_{i,j}} = \left(\frac{\partial e^L_{z,m}}{\partial n^L_{i,m}}\right)\left(\frac{\partial n^L_{i,m}}{\partial w^L_{i,j}}\right) = \left(-\frac{\partial a^L_{z,m}}{\partial n^L_{i,m}}\right)\left(\frac{\partial n^L_{i,m}}{\partial w^L_{i,j}}\right) = \left(-\frac{\partial a^L_{z,m}}{\partial n^L_{i,m}}\right) a^{L-1}_{j,m} \tag{4.77}$$

If parameter x_c represents biases b^l_i where i = 1 to N^l, $l = L$, then, using equations (4.19), (4.20) and (4.21), we have

$$\frac{\partial e^L_{z,m}}{\partial b^L_i} = \left(\frac{\partial e^L_{z,m}}{\partial n^L_{i,m}}\right)\left(\frac{\partial n^L_{i,m}}{\partial b^L_i}\right) = \left(-\frac{\partial a^L_{z,m}}{\partial n^L_{i,m}}\right)\left(\frac{\partial n^L_{i,m}}{\partial b^L_i}\right) = \left(-\frac{\partial a^L_{z,m}}{\partial n^L_{i,m}}\right) \tag{4.78}$$

Further, we define Marquardt sensitivity $\tilde{S}^L_{i,u}$ for output layer $l = L$ as

$$\tilde{S}_{i,u}^{L} = \frac{\partial e_{z,m}^{L}}{\partial n_{i,m}^{L}} = -\frac{\partial a_{z,m}^{L}}{\partial n_{i,m}^{L}} \tag{4.79}$$

Thus we have

$$\tilde{S}_{i,u}^{L} = 0 \; if \; z \neq i \tag{4.80}$$

$$\tilde{S}_{i,u}^{L} = -\left(1 - \left(a_{i,m}^{L}\right)^{2}\right) \; if \; z = i \tag{4.81}$$

Step 6. If parameter x_c represents weights $w_{i,j}^{l}$ where i = 1 to N^{l}, j = 1 to N^{l-1}, l = (L-1), then, using equations (4.49), (4.50), (4.51) and (4.79), we have

$$\frac{\partial e_{z,m}^{L}}{\partial w_{i,j}^{L-1}} = \left(\frac{\partial e_{z,m}^{L}}{\partial n_{i,m}^{L}}\right)\left(\frac{\partial n_{i,m}^{L}}{\partial a_{i,m}^{L-1}}\right)\left(\frac{\partial a_{i,m}^{L-1}}{\partial n_{i,m}^{L-1}}\right)\left(\frac{\partial n_{i,m}^{L-1}}{\partial w_{i,j}^{L-1}}\right) = \tilde{S}_{i,u}^{L} w_{i,j}^{L}\left(1 - \left(a_{i,m}^{L-1}\right)^{2}\right) a_{j,m}^{L-2} \tag{4.82}$$

Equation (4.82) can be rewritten in terms of $\tilde{S}_{i,u}^{L-1}$, Marquardt sensitivity of layer l = L-1, as shown in equations (4.83) and (4.84).

$$\frac{\partial e_{z,m}^{L}}{\partial w_{i,j}^{L-1}} = \tilde{S}_{i,u}^{L-1} a_{j,m}^{L-2} \tag{4.83}$$

$$\tilde{S}_{i,u}^{L-1} = \tilde{S}_{i,u}^{L}\left(1 - \left(a_{i,m}^{L-1}\right)^{2}\right) w_{i,j}^{L} \tag{4.84}$$

If parameter x_c represents weights b_{i}^{l} where i = 1 to N^{l}, l = (L-1), then, using equations (4.49), (4.50), (4.52) and (4.79), we have

$$\frac{\partial e_{z,m}^{L}}{\partial b_{i}^{L-1}} = \left(\frac{\partial e_{z,m}^{L}}{\partial n_{i,m}^{L}}\right)\left(\frac{\partial n_{i,m}^{L}}{\partial a_{i,m}^{L-1}}\right)\left(\frac{\partial a_{i,m}^{L-1}}{\partial n_{i,m}^{L-1}}\right)\left(\frac{\partial n_{i,m}^{L-1}}{\partial b_{i}^{L-1}}\right) = \tilde{S}_{i,u}^{L} w_{i,j}^{L}\left(1 - \left(a_{i,m}^{L-1}\right)^{2}\right) \tag{4.85}$$

Equation (4.85) can be rewritten in terms of $\tilde{S}_{i,u}^{L-1}$, as shown in equations (4.86).

$$\frac{\partial e_{z,m}^{L}}{\partial b_{i}^{L-1}} = \tilde{S}_{i,u}^{L-1} \tag{4.86}$$

Step 7. In general, if parameter x_c represents weights $w_{i,j}^{l}$ where i = 1 to N^{l}, j = 1 to N^{l-1}, l = (L-1) to 1, then, using equation (4.79), we have

$$\frac{\partial e_{z,m}^{L}}{\partial w_{i,j}^{l}} = \tilde{S}_{i,u}^{l} a_{j,m}^{l-1} \tag{4.87}$$

$$\tilde{S}_{i,u}^{l} = \tilde{S}_{i,u}^{l+1} \left(1 - \left(a_{i,m}^{l}\right)^{2}\right) w_{i,j}^{l+1} \tag{4.88}$$

Also, in general, if parameter x_c represents weights b_i^l where i = 1 to N^l, l = (L-1) to 1, then, using equation (4.79), we have

$$\frac{\partial e_{z,m}^{L}}{\partial b_{i}^{l}} = \tilde{S}_{i,u}^{l} \tag{4.89}$$

Step 8. Check the stopping criteria such as maximum number of epoch, minimum error performance function and minimum gradient. If stopping criteria is satisfied, end the training process. If stopping criteria is not satisfied, perform step 9.

Step 9. The performance index value, $E_T^L(k+1)$, is recomputed using (k+1)$^{\text{th}}$ epoch weights and biases. If $E_T^L(k+1)$ is less than $E_T^L(k)$, then the weights and biases update in k^{th} epoch is accepted and the LM parameter (μ) is multiplied by LM parameter decrement factor (μ_d). Next, steps 3 to 8 are performed for next epoch. If, however, $E_T^L(k+1)$ is greater than $E_T^L(k)$, then the weights and biases update in k^{th} epoch is rejected and the LM parameter is multiplied by LM parameter increment factor (μ_i). Next, steps 5 to 8 are again performed in order to compute weights and biases for next epoch.

In summary, the LM algorithm begins with μ set to some small value. If it does not yield a smaller value of performance function in current epoch, then the process is repeated with increased μ. Eventually the performance function should decrease, since we would be moving in the direction of steepest-descent. If it does produce a smaller value for performance function, then μ is decreased for the next epoch, so that the LM algorithm approaches Gauss-Newton method which provides faster convergence. Therefore, the LM algorithm provides a good compromise between the speed of Newton's method and the guaranteed convergence of steepest descent. The LM algorithm is also fastest neural network training algorithm for moderate number of parameters which converges in fewer iterations than gradient-descent method and its variants. However, the key drawback of the LM algorithm is the memory requirement. The algorithm

must store the approximate $r \times r$ Hessian matrix $\mathbf{J}^T\mathbf{J}$, where r is the number of parameters in the network given by equation (4.72). In comparison, the gradient-descent and its variants need only their gradients to be stored, which require $r \times 1$ vector. Hence, when the number of parameters is very large, it may be impractical to use the LM algorithm.

4.3.3 Dynamic Modeling of pH Neutralization Process

ANN based dynamic model development of pH neutralization process consist of following three important phases (Beale et al., 2015).

(i) Collection, preprocessing and division of data

Before starting neural network design and implementation process, it is important to first acquire the experimental data from pH neutralization system. Since the neural network can only be as accurate as the data that is used to train the network, care must be taken while planning the experimentation itself. One such important care to be taken is that the collected data span the full range of the input space for which the network will be used. Multilayer networks can be trained to generalize well within the range of inputs for which they have been trained, but they do not have the ability to accurately extrapolate beyond this range. Another important step to be taken is that the acquired data must be normalized before applying it to the neural network. The reason being that tan-sigmoid activation functions used in hidden layers of multilayer networks, become essentially saturated when the net input is greater than three. If this happens at the beginning of the training process, the gradients will be very small, and the network training will be very slow. Normalizing input-target data set during preprocessing stage ensures that the network output always falls into a normalized range. During post-processing stage, the network output is transformed back into the units of the original target data.

With respect to specifications given in Table 4.1, Figures 4.5(a) and 4.5(b) shows the flowchart and LabVIEW block diagram for acquisition of sampled data from pH neutralization system, respectively. Figure 4.6(a) shows the speeds of pumps A and B at every 10^{th} sampling instants, starting from 1^{st} sample, i.e. modified sample numbers 1, 2, 3, 4, 5, and so on corresponds to values at sampling instants 1, 11, 21, 31, 41, and so on, and Figure 4.6(b) shows the corresponding pH response. Equations (4.90) and (4.91) give input vector (X) and target vector (T) for neural network.

$$X = \begin{bmatrix} S_a(11-d) & S_a(12-d) \dots & S_a(n-d) \\ S_a(12-d) & S_a(13-d) \dots & S_a(n-d+1) \\ \vdots & \vdots & \vdots \\ S_a(11) & S_a(12) \dots & S_a(n) \\ S_b(11-d) & S_b(12-d) \dots & S_b(n-d) \\ S_b(12-d) & S_b(13-d) \dots & S_b(n-d+1) \\ \vdots & \vdots & \vdots \\ S_b(11) & S_b(12) \dots & S_b(n) \\ pH(11-d) & pH(12-d) \dots & pH(n-d) \\ pH(12-d) & pH(13-d) \dots & pH(n-d+1) \\ \vdots & \vdots & \vdots \\ pH(10) & pH(11) \dots & pH(n-1) \end{bmatrix} \quad (4.90)$$

$$T = [pH(11) \quad pH(12) \dots \quad pH(n)] \quad (4.91)$$

where total number of samples 'n' is 32750 collected at sampling instants of 1 second, number of delayed samples 'd' varies from 0 to 10, current values of network variables are $\{S_a(i), S_b(i), pH(i)\}$, past values of network variables are $\{S_a(i-d), S_b(i-d), pH(i-d)\}$, and sample number 'i' varies from 11 to n.

For ANN model development, X and T are randomly divided into following three subsets: training set consisting 70% of '(n-10)' data samples; validation set consisting 15% of '(n-10)' data samples; testing set consisting 15% of '(n-10)' data samples.

(ii) Creation, configuration and initialization of network

ANN model development for highly nonlinear pH neutralization process requires dynamic and multilayer neural network architecture. We have used, in this thesis, the widely adopted tapped delay line approach, mainly due to its simplicity of implementation using established feedforward neural network architecture and supervised training algorithms. As shown in Figure 4.7, the dynamic feedforward network uses current value of pumps speed, and past values of pH and pumps speed as the network inputs, to predict current value of pH as the network output. The resulting dynamic feedforward architecture is equivalent to nonlinear autoregressive network with external inputs (NARX) which can be described using equation (4.92).

$$pH(i) = f\big(S_a(i-d), \dots, S_a(i), S_b(i-d), \dots, S_b(i), pH(i-d), \dots, pH(i-1)\big) \quad (4.92)$$

where 'i' is current sample number.

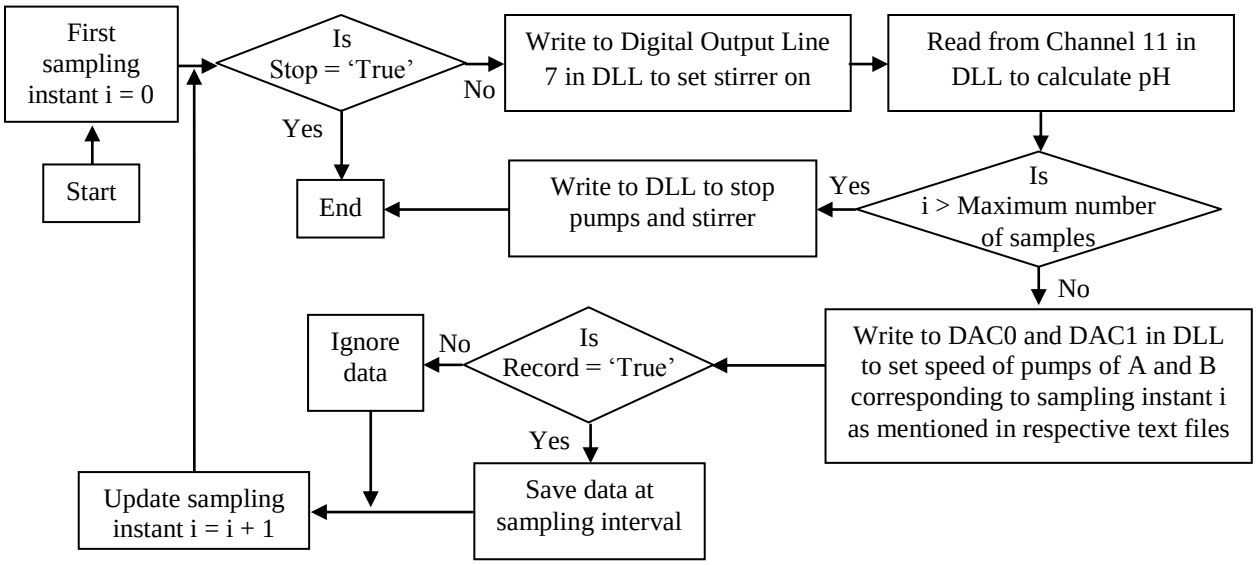


Figure 4.5(a) Flowchart for data acquisition from pH neutralization system

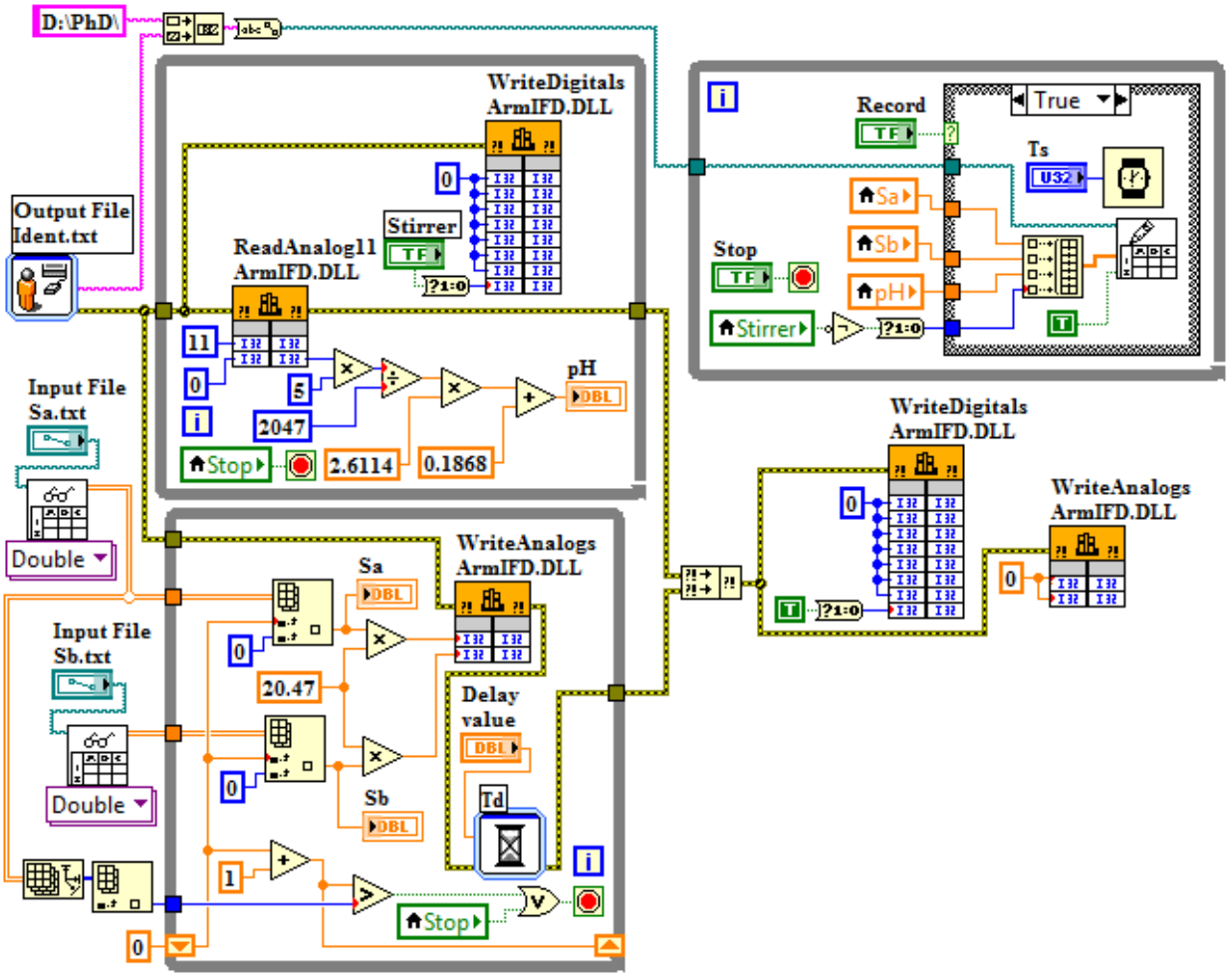


Figure 4.5(b) LabVIEW block diagram for data acquisition from pH neutralization system

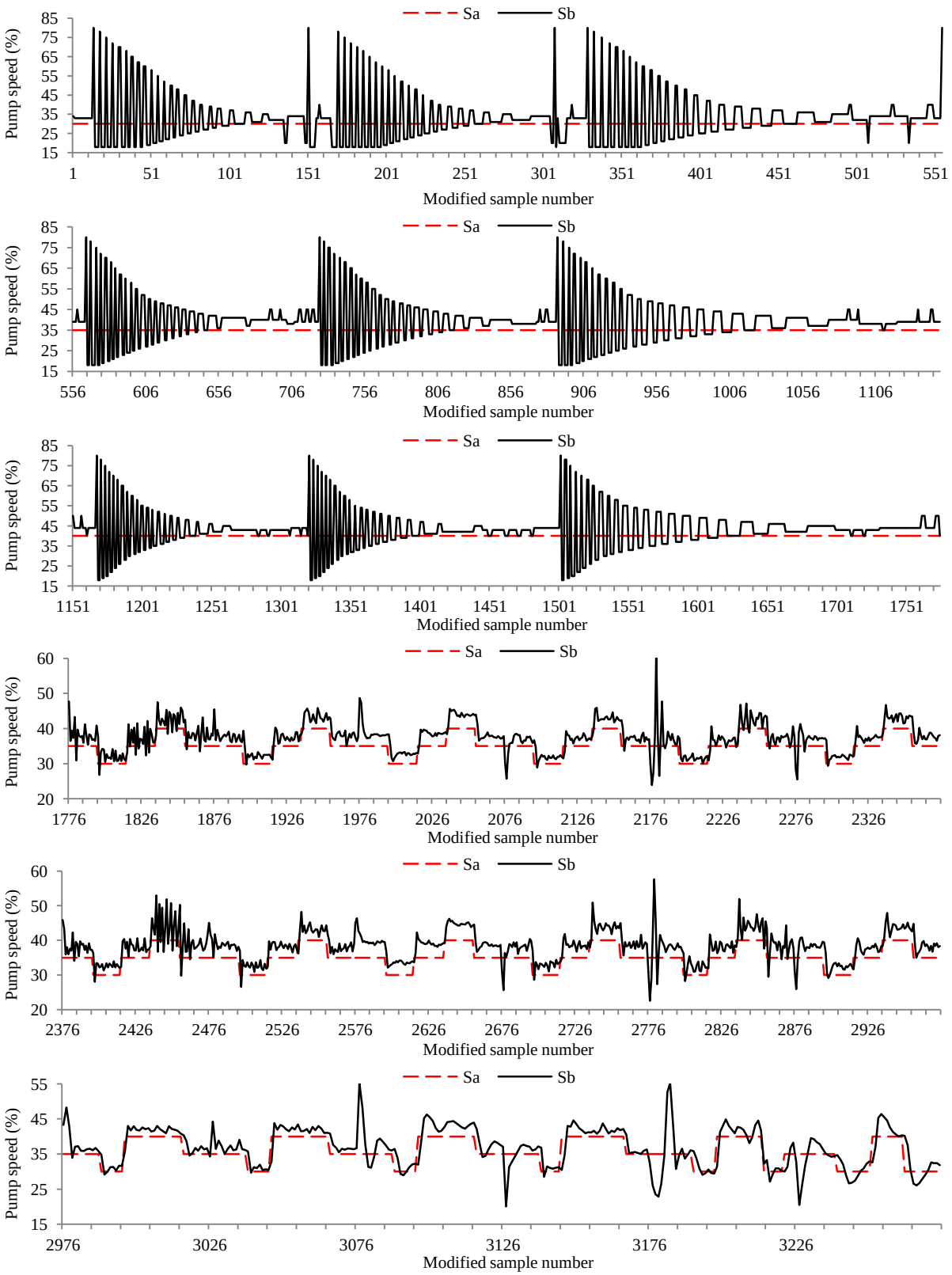


Figure 4.6(a) Pumps speed at every 10th sampling instants, starting from 1st sample, for ANN model development

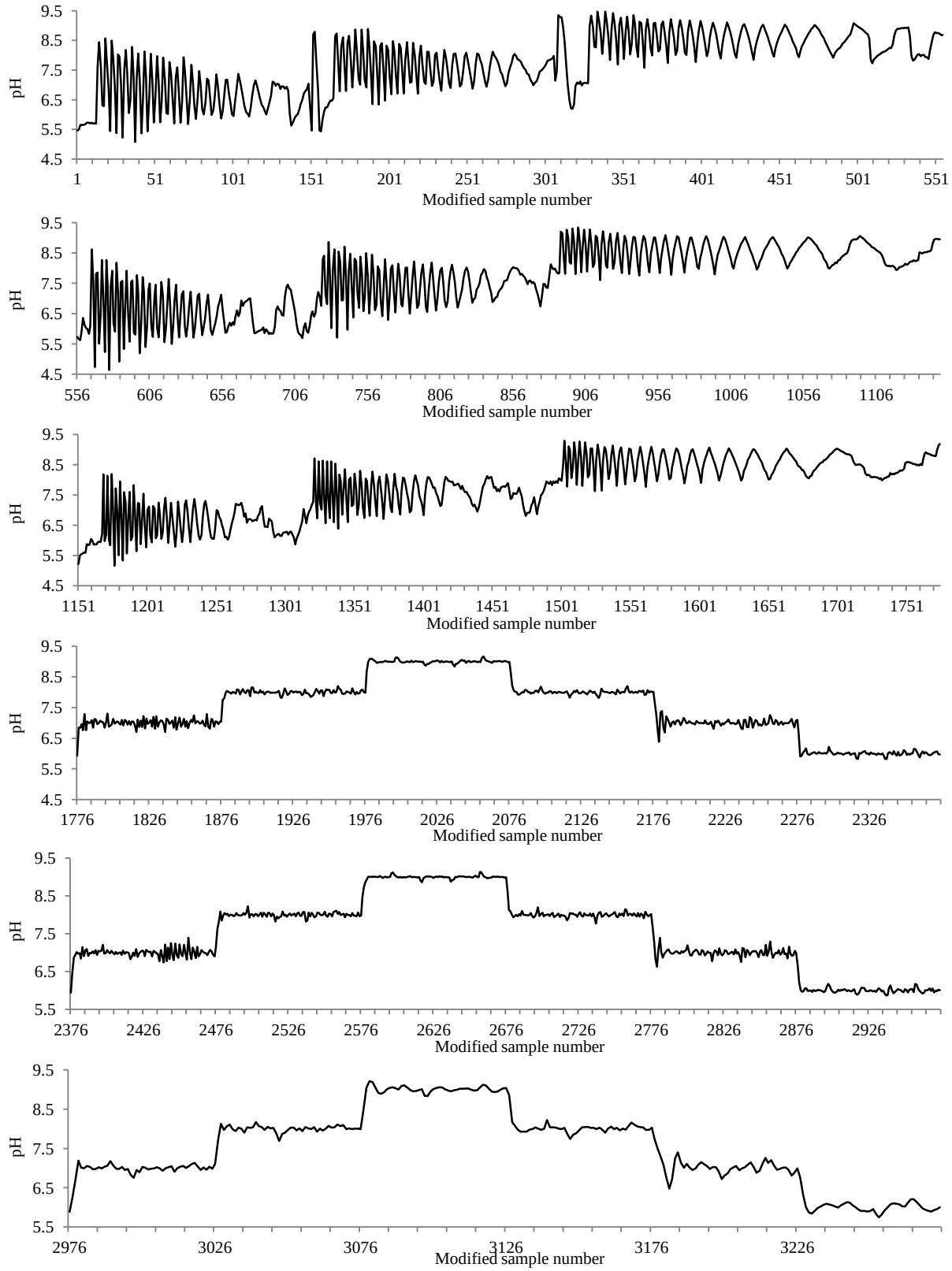


Figure 4.6(b) pH response at every 10th sampling instants, starting from 1st sample, for ANN model development

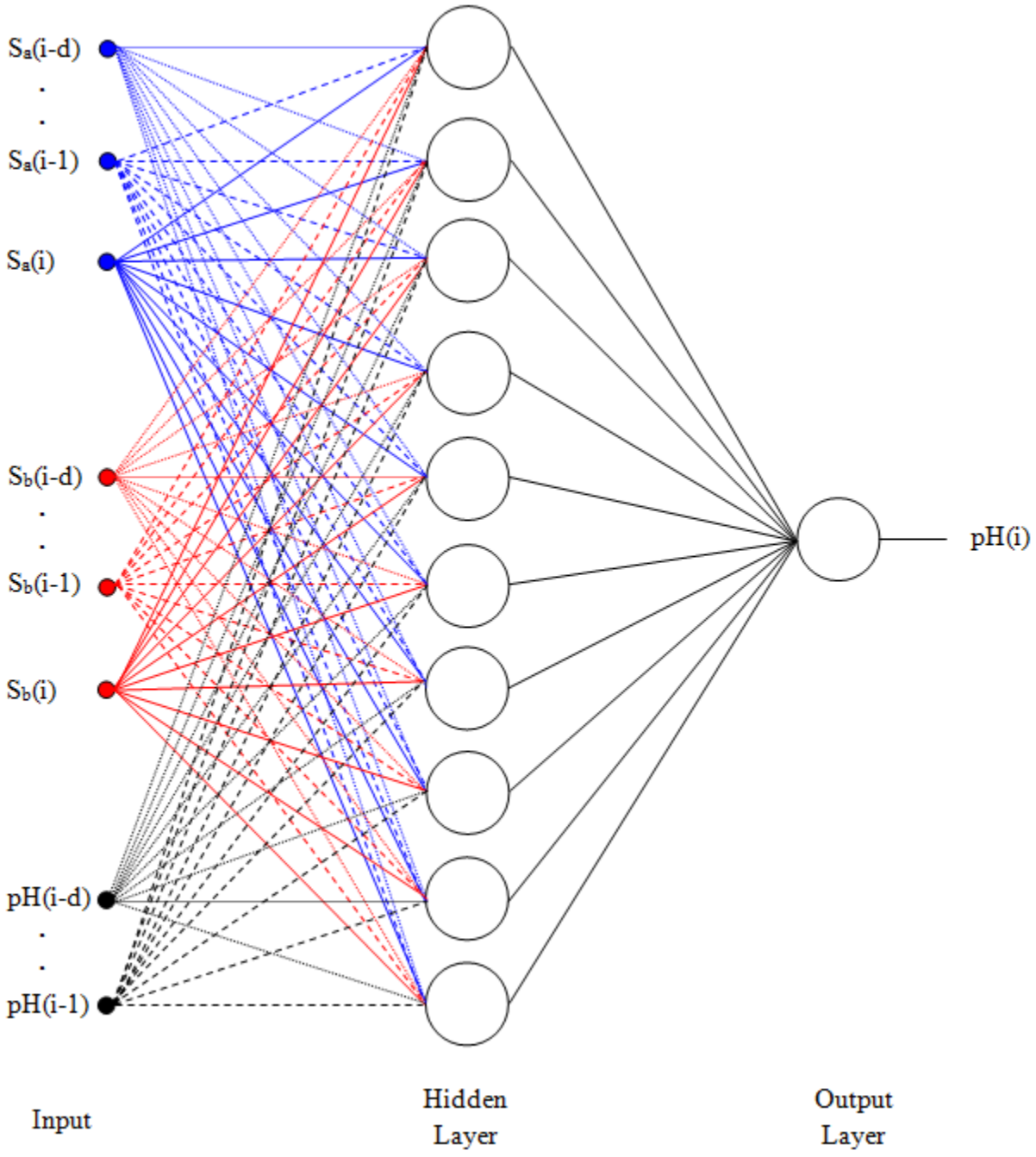


Figure 4.7 Dynamic feedforward neural network architecture

Dynamic neural network shown in Figure 4.7 is created and configured as per neural network properties mentioned in Table 4.3(a). Use of two layered neural network with adequate number of neurons limits computational complexity and chances of overfitting. Table 4.3(b) gives various parameter values of different training functions. Finally, all weights and biases are initialized with random values before start of training process.

Table 4.3(a) Dynamic neural network properties

Parameter	Specification
Network type	feedforward backpropagation
Training functions	GD, GDM, GDA, GDAM, LM
Performance function	MSE
Number of layers	2
Number of neurons in layer 1 (hidden)	10
Number of neurons in layer 2 (output)	1
Transfer function in all layers	tan-sigmoid

Table 4.3(b) Training functions parameters

Parameter	Specification				
	GD	GDM	GDA	GDAM	LM
Maximum number of training epochs	1000	1000	1000	1000	1000
Maximum training time	∞	∞	∞	∞	∞
Minimum training performance value	0	0	0	0	0
Minimum gradient magnitude	10^{-5}	10^{-5}	10^{-5}	10^{-5}	10^{-5}
Maximum number of validation increases	6	6	6	6	6
Learning rate (α)	0.01	0.01	0.01	0.01	
Learning rate increment factor (α_i)			1.05	1.05	
Learning rate decrement factor (α_d)			0.7	0.7	
Maximum performance increase (δ)			1.04	1.04	
Momentum factor (γ)		0.9		0.9	
LM parameter (μ)					0.001
LM parameter decrement factor (μ_d)					0.1
LM parameter increment factor (μ_i)					10
Maximum value of LM parameter (μ_{max})					10^{10}

(iii) Training, validation and testing of network

Training process in a neural network involves tuning the weights and biases of the network in successive epochs to optimize network MSE with the help of training function and input-target data set. We have used in this thesis backpropagation algorithm based training functions in batch mode so that global optimization of neural network could be achieved over entire span of input space.

In successive epochs, training process uses training data set for computing either the gradient of the MSE or the Jacobian of the errors, with respect to the weights and biases, and updating the weights and biases. The validation data set MSE is also monitored at the end of successive epochs. The validation MSE normally decreases during the initial phase of training, as does the training set MSE and the gradient magnitude. However, when the network begins to overfit the training set data, the validation MSE begins to rise although training MSE may still decrease further. If the validation MSE increases for consecutive epochs equals maximum number of validation increases as provided by the user, training process will be terminated, and network weights and biases are set to the value which were at the minimum of the validation MSE. Also, the gradient magnitude becomes very small as the training process reaches minimum MSE value. If the gradient magnitude, in a particular epoch, is less than its minimum limit set by user, training process will be terminated, and network weights and biases are set to the values which were at the end of the particular epoch. Additionally, training process will be terminated if other stopping criteria, such as maximum number of epochs, minimum training MSE value and maximum training time, are satisfied.

Training process also computes test data set MSE in each epoch, but it is not used as stopping criteria. Also, if the test data set MSE reaches a minimum at a significantly different epoch than the validation data set MSE, it indicates poor division of the data set.

4.3.4 Performance Evaluation of Training Functions for Dynamic pH Model Development

In this section, performance of different training functions, namely GD, GDM, GDA, GDAM and LM, used in BP algorithm are evaluated. Out of total 32750 experimental samples, first 10 samples are taken to account for various initial delays and last 32740 samples as input-target data set for ANN based dynamic pH model development. These ANN data samples have been

randomly divided such that number of samples used for training, validation, and testing are 22918, 4911, and 4911 respectively.

Table 4.4 presents performance summary of training functions for d = 3 and 6 when the training process is conducted till stopping criteria, such as maximum number of epochs, maximum training time, minimum training MSE, minimum magnitude of gradient and maximum number of validation increases, is satisfied. Following observations are made about the results.

(i) For d = 3 and 6, GD results in training MSE of 0.0223 and 0.0161, and validation MSE of 0.0201 and 0.0158, respectively after reaching maximum limit of 1000 epochs. Thus neural network training process by GD is very slow. Training performance value can be improved further if we increase the limit for maximum number of epochs. However, it is not guaranteed that MSE will reduce to an order of 10^{-4} or so.

(ii) For d = 3 and 6, GDM results in training MSE of 0.0210 and 0.0148, and validation MSE of 0.0190 and 0.0142, respectively after reaching maximum limit of 1000 epochs, not much improvement over GD both in terms of MSE value as well as number of epochs. Similar to GD, training performance value can be improved further if we increase the limit for maximum number of epochs.

(iii) For d = 3, GDA results in training and validation MSE of 0.0422 and 0.0364 respectively at 49^{th} epoch, and ANN training stops after next 6 epochs due to consecutive validation MSE increase. For d = 6, GDA gives training and validation MSE of 0.0366 and 0.0379 respectively at 61^{th} epoch, and ANN training stops after next 6 epochs. Thus GDA results in MSE performance value comparable with GD in much less number of epochs but algorithm is getting trapped in local minimum point.

(iv) For d = 3, GDAM results in training and validation MSE of 0.0210 and 0.0190 respectively at 87^{th} epoch, and ANN training stops after next 6 epochs. For d = 6, GDAM results in training and validation MSE of 0.0135 and 0.0128 respectively at 131^{th} epoch, and ANN training stops after next 6 epochs. For d = 6, GDAM performs better than GD, GDM and GDA but not for d = 3. Therefore, better performance of GDAM is not guaranteed since the algorithm is getting trapped in local minimum point as evident from d = 3 results.

(v) For d = 3 and 6, LM gives training MSE of 5.175×10^{-4} and 3.479×10^{-4}, and validation MSE of 4.535×10^{-4} and 3.361×10^{-4}, at 201^{th} and 248^{th} epoch respectively, and training stops after next 6 epochs. LM takes moderate number of epochs to arrive at MSE performance value of the order of 10^{-4}, which can be considered as global optimum performance and far superior than MSE performance values of GD, GDM, GDA and GDAM.

Table 4.5 shows the performance of LM when d varies from 0 to 10. Observations derived from results are as follows.

(i) For d = 0, 1, LM perform poorly. For d = 0, neural network is being trained on the basis of current values of pumps speed only and it does not contain any information about network initial conditions. For d = 1, the LM algorithm is trapped in local optimum point.

(ii) For d = 2 to 10, LM results in performance function values of neural network of the order of 10^{-4}. Since testing data set is not used for network parameter update, testing MSE value can be used for comparing performance of various ANN models. Based on testing MSE values, the two best trained ANN model are obtained for d = 10 and 6 with LM resulting in testing MSE values as 3.648×10^{-4} and 3.776×10^{-4} respectively. For d = 3, 5, 7, 8, and 9, LM results in ANN models with comparable testing MSE performance values as 4.671×10^{-4}, 4.437×10^{-4}, 4.447×10^{-4}, 4.489×10^{-4}, and 4.040×10^{-4} respectively. For d = 2 and 4, LM results in testing MSE values as 5.966×10^{-4} and 8.576×10^{-4} respectively.

Since number of inputs to neural network becomes very large for large values of d, we will consider ANN model with d = 3 and 6 only for further comparisons. For d = 3 and 6, Figures 4.8(a) and 4.8(b) respectively shows the error at every 10^{th} sampling instants, starting from 1^{st} sample, in the trained neural network response when subjected to ANN training, validation, and testing data set. It is found that magnitude of error never exceeds 0.4 pH unit for entire data set of 32740 samples. Also more than 99% of those errors lie within magnitude range of 0.1 pH unit. For d = 3, Figures 4.9(a), 4.9(b) and 4.9(c) show performance, training state and regression plots respectively. Figures 4.10(a), 4.10(b) and 4.10(c) show corresponding plots for d = 6. The performance plot shows the values of MSE for training, validation and testing data sets in successive epochs. The training state plot shows the value of LM parameter Mu (μ), its gradient magnitude, and validation fail in successive epochs. The regression plot shows regression

between network output and network target for individual training, validation testing data sets as well as total data set. The dashed line in regression plot represents the perfect result i.e. network output equals network target, and the solid line represents the best fit linear regression line between network output and target. The R-value of correlation coefficient indicates of the relationship between the network output and target. R = 1 indicates that there is an exact linear relationship between network output and target, and R close to zero indicates that there is no linear relationship between network output and target.

4.4 Concluding Remarks

In this chapter, we have developed dynamic pH model based on: (i) first principles technique proposed by McAvoy et al. (1972) and (ii) feedforward dynamic ANN. McAvoy et al. (1972) based dynamic pH model gives poor performance function values for various step tests conducted at pH = 7. For initial acid and base pump speeds as 35% and 38.5% respectively, when the base pump speed is given step changes of 41.5%, 31.5%, 21.5%, 11.5%, 1.5%, -3.5%, -8.5% and -18.5%, keeping the acid pump speed at 35%, the McAvoy et al. (1972) based dynamic pH model response does not obey the experimental results, especially in the dynamic pH range of 4 to 10. Feedforward ANN based dynamic pH model using 32750 experimental data samples covering dynamic region pH range from 4 to 10 has been developed. Since pH process is extremely nonlinear, TDL method is used to represent delayed input-output samples. For d = 3 and 6, training, validation and testing MSE values of ANN models are evaluated for BP algorithm with different training functions, namely GD, GDM, GDA, GDAM, and LM, in batch mode. From the results it is found that MSE value for LM is much less than that for GD and its variants. Also, in order to obtain best delay setting for TDL method based dynamic ANN configuration, d is varied from 0 to 10 and MSE values are evaluated using LM function. It is found that for d = 6 we get trained network with minimum training and validation MSE values as 3.479×10^{-4} and 3.361×10^{-4} respectively at 248^{th} epoch. However, with d = 6 the number of inputs to neural network is very large. For rest of the thesis, we have used ANN model with d = 3 which gives acceptable trained network performance with training, validation, and testing MSE values at 201^{th} epoch as 5.175×10^{-4}, 4.535×10^{-4}, and 4.671×10^{-4} respectively.

Table 4.4 Performance comparison of different training functions for dynamic pH modeling

Training function	No. of delayed samples (d)	Neural network training, validation and testing					
		Best epoch number for minimum validation	Training MSE at best epoch number	Validation MSE at best epoch number	Testing MSE at best epoch number	Epoch number where training terminated	Overall regression
GD	3	1000	0.0223	0.0201	0.0219	1000	0.98860
GDM	3	1000	0.0210	0.0190	0.0208	1000	0.98923
GDA	3	49	0.0422	0.0364	0.0415	55	0.97937
GDAM	3	87	0.0210	0.0190	0.0210	93	0.98932
LM	3	201	5.175×10^{-4}	4.535×10^{-4}	4.671×10^{-4}	207	0.99974
GD	6	1000	0.0161	0.0158	0.0168	1000	0.99160
GDM	6	1000	0.0148	0.0142	0.0153	1000	0.99230
GDA	6	61	0.0366	0.0379	0.0384	67	0.98137
GDAM	6	131	0.0135	0.0128	0.0139	137	0.99301
LM	6	248	3.479×10^{-4}	3.361×10^{-4}	3.776×10^{-4}	254	0.99982

Table 4.5 Performance comparison of LM for various amount of delayed input-output samples

No. of delayed samples (d)	Neural network training, validation and testing					
	Best epoch number for minimum validation	Training MSE at best epoch number	Validation MSE at best epoch number	Testing MSE at best epoch number	Epoch number where training terminated	Overall regression
0	51	0.9358	0.9345	0.8962	57	0.18968
1	63	1.959×10^{-3}	1.714×10^{-3}	1.857×10^{-3}	69	0.99901
2	216	6.065×10^{-4}	5.875×10^{-4}	5.966×10^{-4}	222	0.99969
3	201	5.175×10^{-4}	4.535×10^{-4}	4.671×10^{-4}	207	0.99974
4	325	9.135×10^{-4}	8.544×10^{-4}	8.576×10^{-4}	331	0.99954
5	36	4.489×10^{-4}	4.230×10^{-4}	4.437×10^{-4}	42	0.99977
6	248	3.479×10^{-4}	3.361×10^{-4}	3.776×10^{-4}	254	0.99982
7	102	4.120×10^{-4}	4.272×10^{-4}	4.447×10^{-4}	108	0.99978
8	53	4.304×10^{-4}	4.129×10^{-4}	4.489×10^{-4}	59	0.99978
9	191	4.267×10^{-4}	4.116×10^{-4}	4.040×10^{-4}	197	0.99978
10	239	3.586×10^{-4}	3.803×10^{-4}	3.648×10^{-4}	245	0.99981

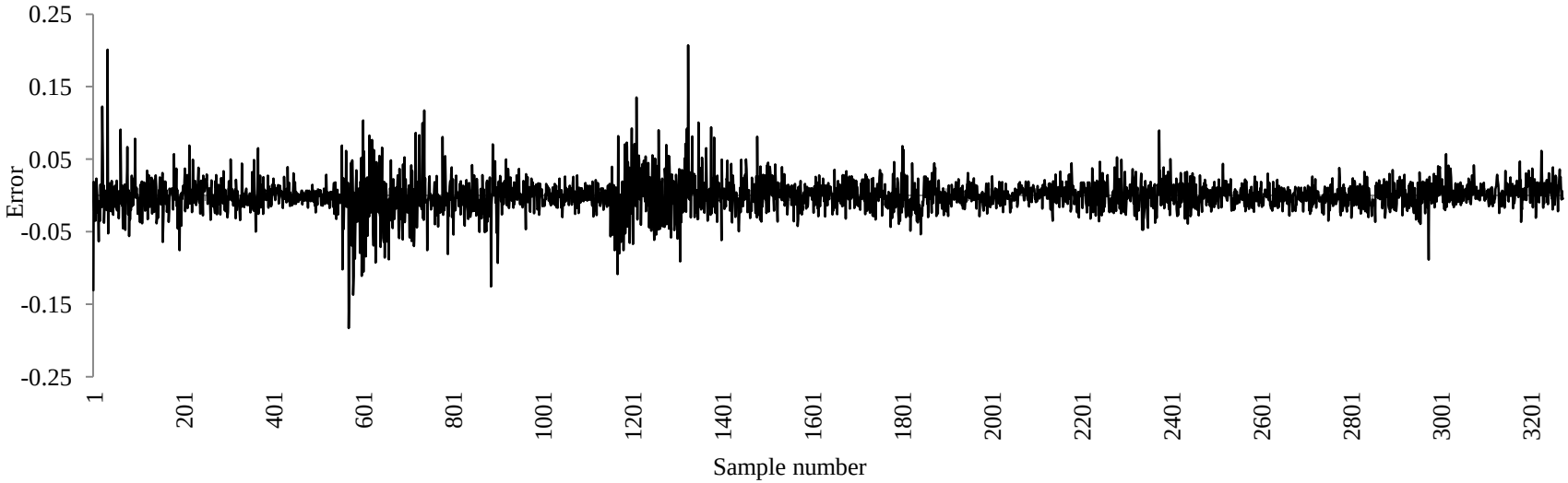


Figure 4.8(a) LM algorithm based ANN pH model error at every 10th sampling instants, starting from 1st sample, for three delayed input-output samples

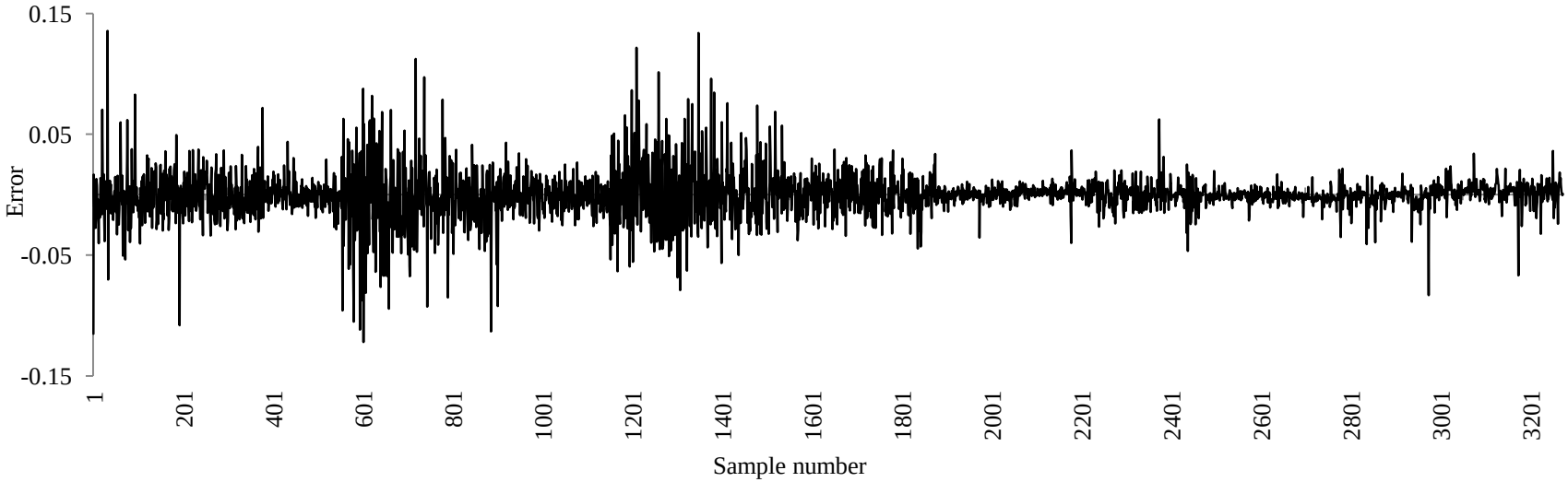


Figure 4.8(b) LM algorithm based ANN pH model error at every 10th sampling instants, starting from 1st sample, for six delayed input-output samples

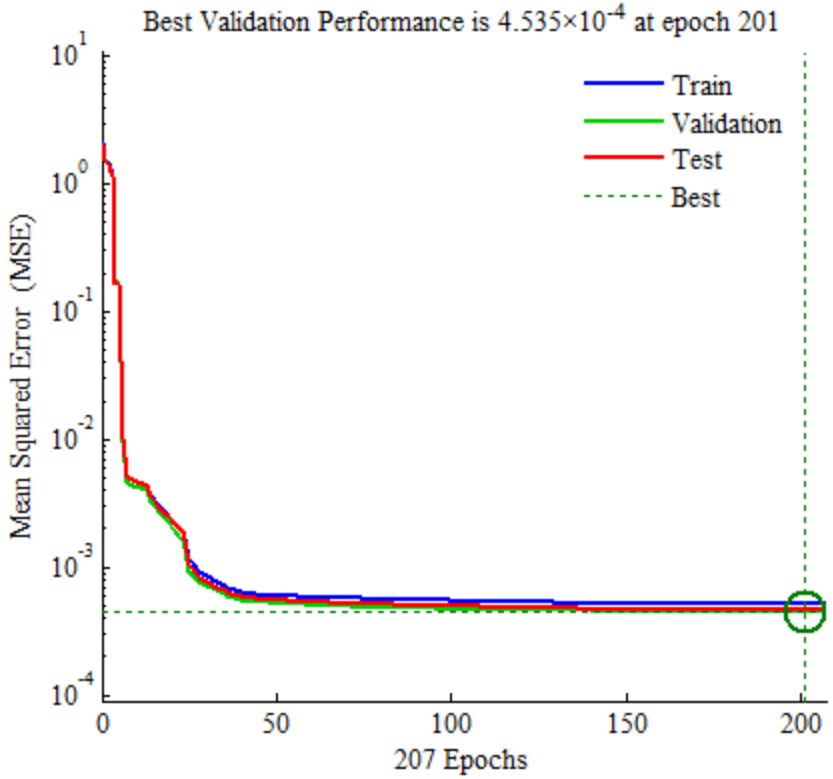


Figure 4.9(a) LM algorithm based ANN pH model performance plot for three delayed input-output samples

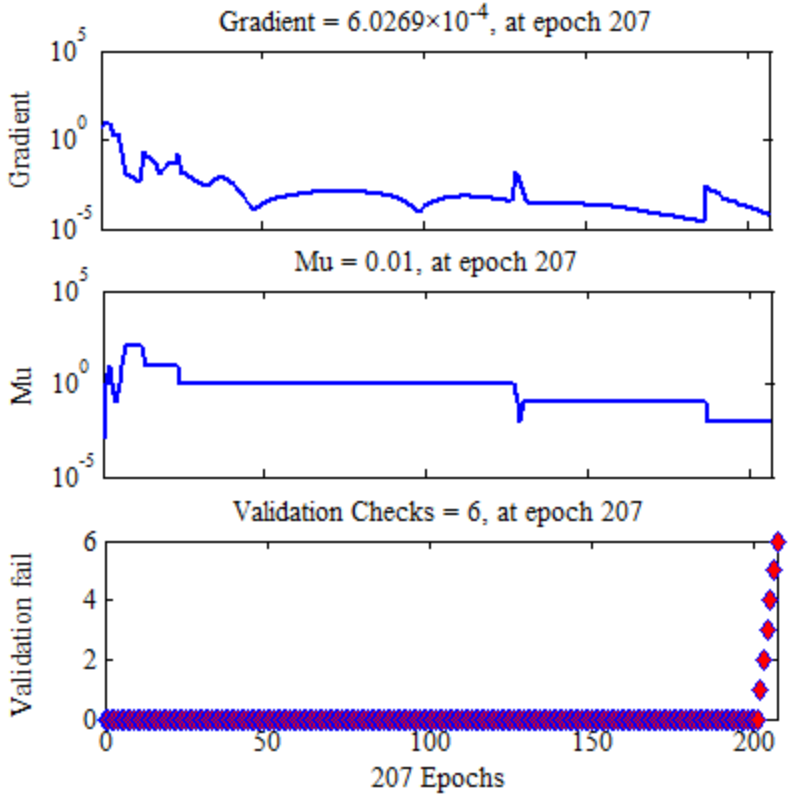


Figure 4.9(b) LM algorithm based ANN pH model training state for three delayed input-output samples

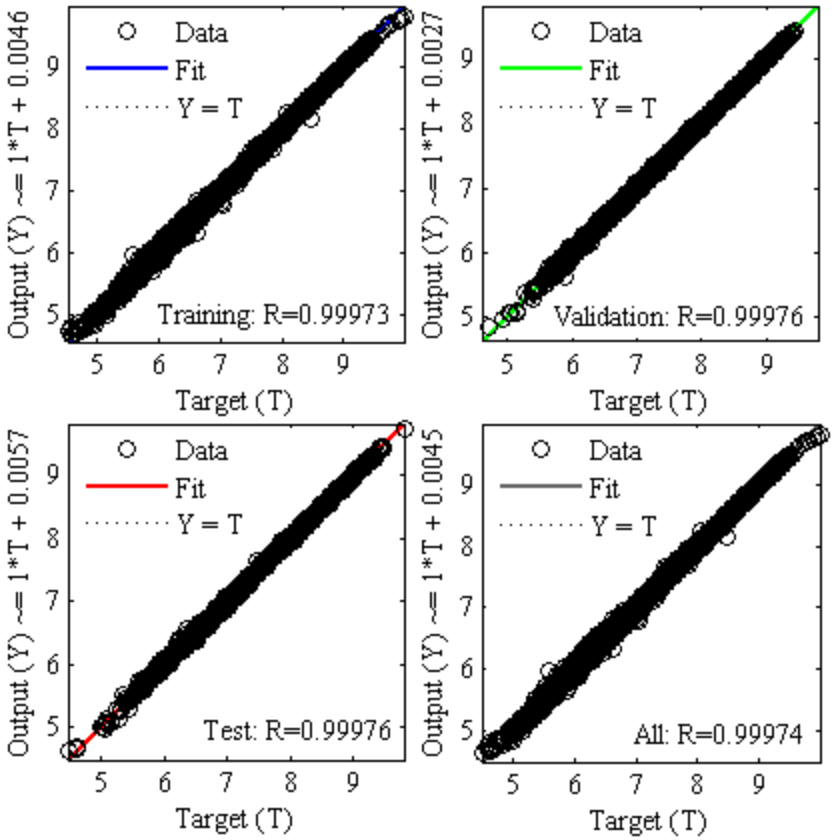


Figure 4.9(c) LM algorithm based ANN pH model regression plot for three delayed input-output samples

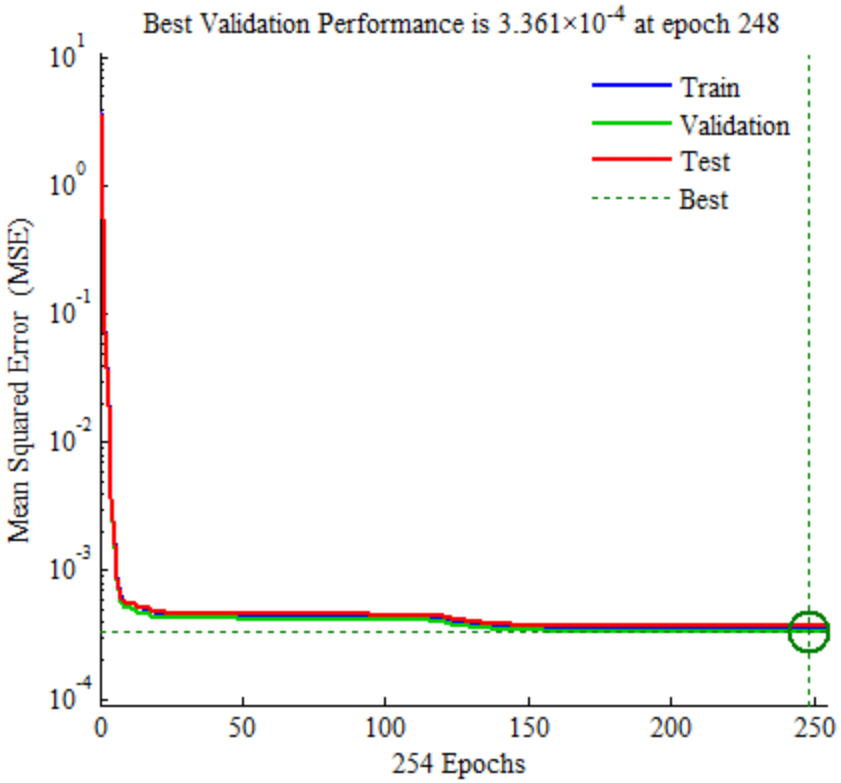


Figure 4.10(a) LM algorithm based ANN pH model performance plot for six delayed input-output samples

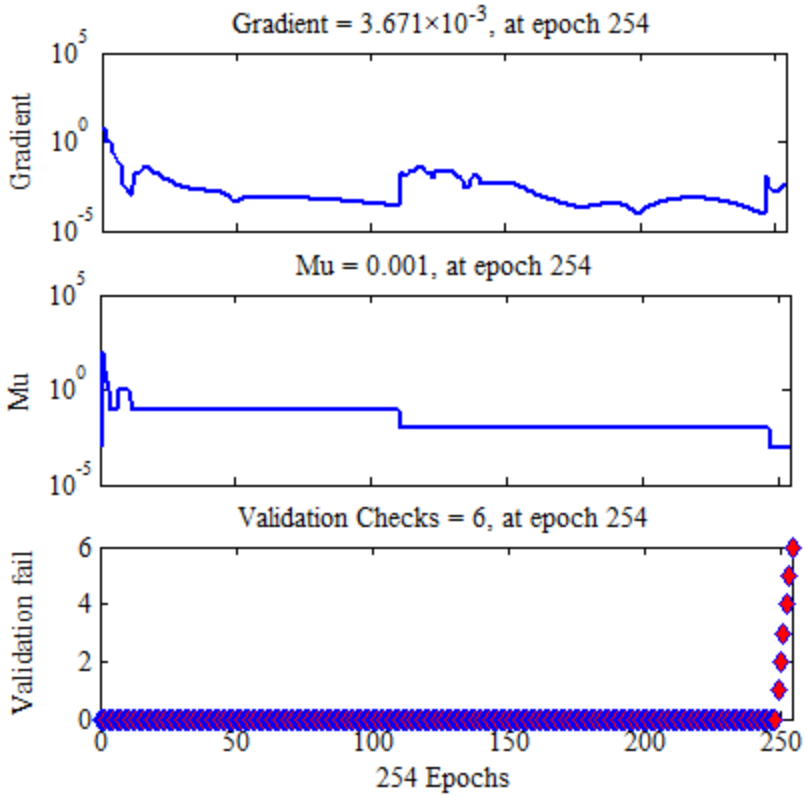


Figure 4.10(b) LM algorithm based ANN pH model training state for six delayed input-output samples

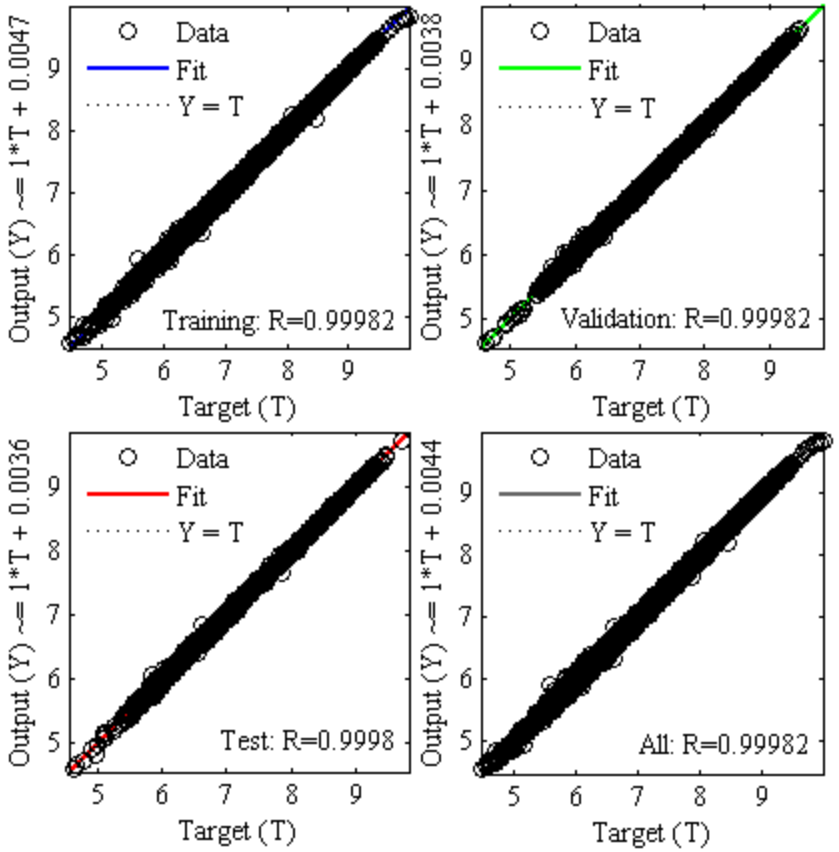


Figure 4.10(c) LM algorithm based ANN pH model regression plot for six delayed input-output samples

CHAPTER 5

OPTIMIZED FUZZY LOGIC BASED pH CONTROL SCHEMES

5.1 Introduction

Control system and engineering in past has been an art practiced and perfected by operators/engineers based on their experience. Towards end of first half in the last century, industrial growth led to design, analysis and implementation of classical control techniques, such as PID. However, towards end of second half in last century, modernization of process industries and production methods resulted in highly nonlinear, complex and partly unknown system behavior. Researchers addressed this problem with development of advanced control methods based on nonlinear process model, but model accuracy remained a vital impediment. Intelligent control methods, especially FLC, without using process model provide an alternate solution to aforesaid problem. FLC essentially relies on extensive and successful experience of the system operator gained through observation, study and understanding the behavior of system. In this chapter, basic design of PID control and direct FLC has been presented. The parameters of PID and fuzzy logic controllers need to be tuned for optimal or best performance. Global optimization techniques namely GA, DE and PSO have been used to optimize PID and fuzzy logic controllers. The performance of optimization techniques has been evaluated using MATLAB® simulations on ANN based pH model. The performance of optimization techniques for FLC has also been experimentally validated on Armfield® pH neutralization process using LabVIEW® implementations.

5.2 Conventional Proportional-Integral-Derivative (PID) Control

The dynamic performance of conventional PID controller on the pH neutralization system is often used as a benchmark against which advanced intelligent control schemes such as FLC is compared. Let the pH neutralization system encounters deviation as shown below.

$$e(t) = pH_{SP}(t) - pH(t) \tag{5.1}$$

where $e(t)$, $pH_{SP}(t)$ and $pH(t)$ are instantaneous values of error, set-point and output of the pH neutralization system respectively.

The instantaneous response of a basic PID controller in parallel form due to error $e(t)$ is given by following equation.

$$o_{PID}(t) = K_P e(t) + K_I \int_0^t e(t)\,dt + K_D \frac{de(t)}{dt} + o_{PID}(0) \tag{5.2}$$

where K_P is the proportional gain in $\frac{\%}{pH}$, K_I is the integral gain in $\frac{\%/s}{pH}$, K_D is the derivative gain in $\frac{\%}{pH/s}$, $o_{PID}(0)$ is the initial PID controller output in % and $o_{PID}(t)$ is the instantaneous PID controller output in %.

The first three terms on right-hand side of equation (5.2) show that PID controller acts instantly using proportional action, takes the past into account using integral action and anticipates the future using derivative action.

To implement PID controller in software, equations (5.1) and (5.2) are discretized. Using equation (5.1) in discrete domain, the error $e(k)$ at k^{th} sampling instant is given as

$$e(k) = pH_{SP}(k) - pH(k) \tag{5.3}$$

where $pH_{SP}(k)$ and $pH(k)$ are values of set-point and output of the pH neutralization system at k^{th} sampling instant respectively.

Discretizaton of equation (5.2) by using backward rectangular rule for integral term yields discrete PID controller equation as

$$o_{PID}(k) = K_P e(k) + K_I T_s \sum_{i=0}^{k} e(i) + K_D \frac{e(k) - e(k-1)}{T_s} + o_{PID}(0) \tag{5.4}$$

where T_s is sampling interval in second, $e(i)$ is error at i^{th} sampling instant and $o_{PID}(k)$ is PID controller output at k^{th} sampling instant.

Equation (5.4), also known as 'position algorithm' of PID, is not convenient for implementation because it requires large memory to remember all previous error samples as well as initial controller output. A better alternative is to express discrete PID controller expression by taking difference between two consecutive discrete output values as expresses in equation (5.5).

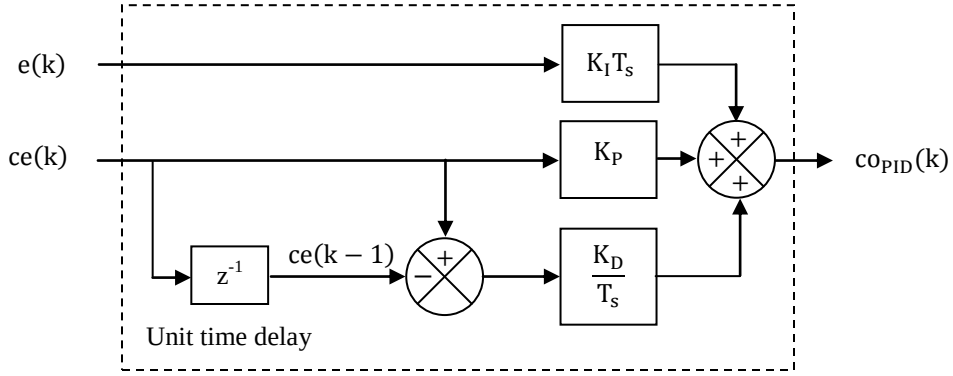


Figure 5.1 Velocity algorithm based PID controller

$$o_{PID}(k) - o_{PID}(k-1) = K_P\big(e(k) - e(k-1)\big) + K_I T_s e(k) + K_D \frac{e(k) - 2e(k-1) + e(k-2)}{T_s} \tag{5.5}$$

The change in error $ce(k)$ and change in PID controller output $co_{PID}(k)$ are defined as follows:

$$ce(k) = e(k) - e(k-1) \tag{5.6}$$

$$co_{PID}(k) = o_{PID}(k) - o_{PID}(k-1) \tag{5.7}$$

Using equations (5.6) and (5.7), equation (5.5) can be rewritten as

$$co_{PID}(k) = K_P ce(k) + K_I T_s e(k) + K_D \frac{ce(k) - ce(k-1)}{T_s} \tag{5.8}$$

Equation (5.8), also known as 'velocity algorithm' of PID, shows that in order to calculate discrete PID output we need to store present error, present change in error, previous change in error and previous output. Figure 5.1 shows schematic diagram of velocity algorithm based PID controller.

5.3 Fuzzy Logic and Fuzzy Inference System (FIS)

In everyday life there are a few situations where human beings convey definite and complete information with exact terms like *yes*, *no*, *true*, *false*, *all*, or *none*. On the other hand, there are many situations where human beings convey uncertain and incomplete information with fuzzy terms like *may be*, *possibly*, *many*, or *some*. Fuzzy logic is used to analytically express such

human thinking and decision making in fuzzy terms. Fuzzy logic based FIS forms the basis for design of FLC structure. This section presents a brief introduction to the various components of FIS, such as fuzzy sets, linguistic variables, fuzzification, fuzzy rule base, fuzzy rule based inference, and defuzzification, to be used in design of fuzzy logic controller

5.3.1 Fuzzy Sets

According to classical set theory, a *crisp set* can be defined by its *characteristic function*. Let *C* be a crisp set from *domain of discourse X* such that $x \in X$ where *x* is any generic element in the domain *X*. The *characteristic function* $\mu_C(x)$ of the crisp set *C* attains value $\mu_C(x) = 1$ if $x \in C$, and $\mu_C(x) = 0$ if $x \notin C$. Alternatively the *characteristic function* can also be represented as $\mu_C: X \rightarrow \{0, 1\}$.

If *A* be a set of all integers *greater* than 10, and *B* be a set of all integers *much greater* than 10, and we need to determine whether the numbers 11, 12, 15, 1150 and 14^{10} belong to sets *A* and/or *B*. It is clear that 11, 12, 15, 1150 and 14^{10} belong to set *A* since the characteristic function of set *A*, μ_A, attains value equal to 1 for all the numbers greater than 10. We could give straightforward answer to the above problem with regard to set *A* because set *A* has been completely defined. The same is not true for set *B* because set *B* has not been sufficiently defined due to presence of a vague term, *much greater*, in its definition. Generally any one will agree that 1150 and 14^{10} are elements of set *B*, but it is doubtful whether 11, 12 and 15 are elements of set *B*. The characteristic function of set *B*, μ_B, cannot be used to describe set *B* because of its inability to address the vagueness in determining the lowest integer which would belong to set *B*. This problem can be solved if an alternate way of describing a set *B* is used instead of traditional set theory where all elements are supposed to belong to the set but with varying *membership degree* values. Zadeh (1965) introduced the concept of *fuzzy set* to describe such classes of object that are vague in nature.

Membership function of a fuzzy set: According to fuzzy set theory, a *fuzzy set* can be defined by its *membership function*. Let *F* be a fuzzy set from *universe of discourse* (*UOD*) *X* such that $x \in X$ where *x* is any generic element in the domain *X*. The *membership function* $\mu_F(x)$ of the fuzzy set *F* assigns value, or *membership degree*, from the *unit interval* [0, 1] to every $x \in F$. Alternatively the *membership function* can also be represented as $\mu_F: X \rightarrow [0, 1]$.

For set *B*, suppose we assign constant membership degree of 0 and 1 for numbers less than 10 and greater than 110 respectively, and a linearly membership degree with slope 0.01 in the range [0, 1] for all numbers lying between 10 and 110. It is verified that 1150 and 14^{10} completely belong to set *B* with membership degree as 1. In addition, 11, 12 and 15 are also part of set *B* with membership degrees as 0.01, 0.02 and 0.05 respectively.

The membership function may take different shapes, such as triangular, trapezoidal, Gaussian and singleton, to arrive at the degree of fuzziness of the crisp set members into the normalized interval [0, 1]. In this thesis, we have used only trapezoidal and triangular membership functions as defined in equations (5.9) and (5.10), and shown in Figures 5.2(a) and 5.2(b) respectively.

$$\mu_{trapezoid}(x) = \begin{cases} 0, & \text{for } x < a \\ \dfrac{x - \mathrm{a}}{\mathrm{b} - \mathrm{a}}, & \text{for a} \le x < b \\ 1, & \text{for b} \le x < c \\ \dfrac{\mathrm{d} - x}{\mathrm{d} - \mathrm{c}}, & \text{for c} \le x \le \mathrm{d} \\ 0, & \text{for } x > d \end{cases} \tag{5.9}$$

$$\mu_{triangle}(x) = \begin{cases} 0, & \text{for } x < a \\ \dfrac{x - \mathrm{a}}{\mathrm{b} - \mathrm{a}}, & \text{for a} \le x < b \\ \dfrac{\mathrm{c} - x}{\mathrm{c} - \mathrm{b}}, & \text{for b} \le x \le \mathrm{c} \\ 0, & \text{for } x > c \end{cases} \tag{5.10}$$

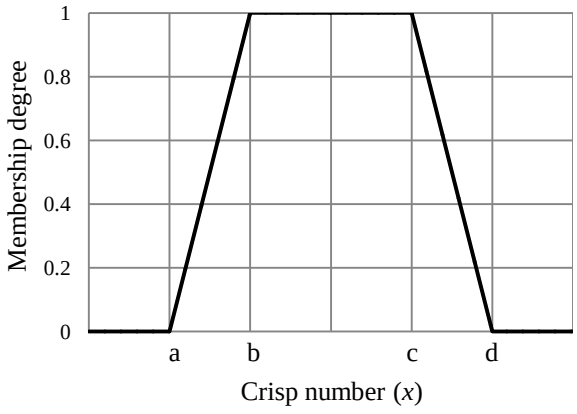


Figure 5.2(a) Trapezoidal membership function

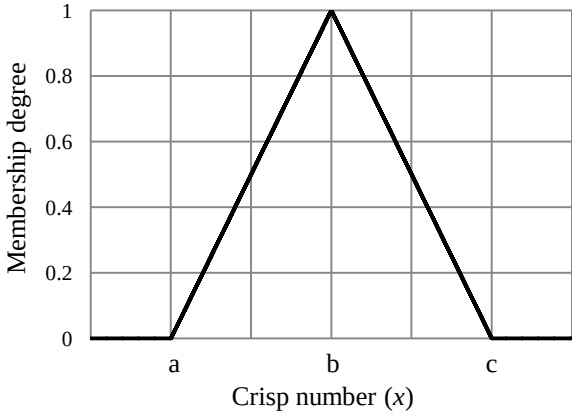


Figure 5.2(b) Triangular membership function

Table 5.1 Fundamental fuzzy set operations

Fuzzy operation	Fuzzy operator ($\wp$)	Definition
Union	OR	$\mu_{A\cup B}(x) = max[\mu_A(x), \mu_B(x)]$
Intersection	AND	$\mu_{A\cap B}(x) = min[\mu_A(x), \mu_B(x)]$
Complement	NOT	$\mu_{\bar{A}}(x) = 1 - \mu_A(x)$

In traditional set theory, *union*, *intersection*, and *complement* operations are defined on crisp sets using classical *or*, *and*, and *not* operators, respectively. Similarly, in fuzzy set theory, the membership function of a fuzzy set as a result of *union*, *intersection*, and *complement* operations on other fuzzy sets can be determined using fuzzy operators ($\wp$), namely OR, AND, and NOT, respectively. These operations are illustrated in Table 5.1, based on definitions proposed by Zadeh (1965), using fuzzy sets A and B having membership functions $\mu_A(x)$ and $\mu_B(x)$ respectively in UOD $X = \{x\}$ where x is a generic element in domain X.

5.3.2 Linguistic Variables

Among various distinguished features of fuzzy logic, use of *linguistic variables* is an important one. Zadeh (1975) states that: "By a *linguistic variable* we mean a variable whose values are words or sentences in a natural or artificial language. For example, *Age* is a linguistic variable if its values are linguistic rather than numerical, i.e., *young*, *not young*, *very young*, *quite young*, *old*, *not very old* and *not very young*, etc., rather than 20, 21, 22, 23, ..."

The linguistic variable may assume different linguistic values over specified UOD. *Fuzzy proposition* assigns linguistic value to the linguistic variable, and is interpreted by a process known as *fuzzification*.

Fuzzy proposition and fuzzification: Let x be a linguistic variable in UOD X such that $x \in X$, and $L_i(x)$ is a fuzzy set having membership function $\mu_{L_i}(x)$ associated with a linguistic value L_i which x may attain. The *fuzzy proposition* P_i can be represented through following structure.

P_i: x is L_i (5.11)

Fuzzification is the process of associating membership degree $\mu_{L_i}(x)$ to crisp numerical value of *x*. Mathematically, *fuzzification* process can be represented as follows: $x \rightarrow \mu_{L_i}(x)$.

Fuzzy propositions are the building blocks of a FIS. In case of multiple input-multiple output system configurations, two or more fuzzy propositions are put in relation using fuzzy operators to describe the complex system. It is important to note that the selection of fuzzy operator directly influences the structure of designed FIS.

Fuzzy relation: Let *x* and *y* be linguistic variables in UODs *X* and *Y* such that $x \in X$ and $y \in Y$, and $L_i(x)$ and $M_j(y)$ be fuzzy sets having membership functions $\mu_{L_i}(x)$ and $\mu_{M_j}(y)$ associated with a linguistic values L_i and M_j which *x* and *y* may attain, respectively. The two-dimensional *fuzzy relation* R_{ij} can be represented using fuzzy operator $\wp$ through following structure.

R_{ij}: *x* is L_i $\wp$ *y* is M_j (5.12)

The fuzzy relation membership function $\mu_{R_{ij}}(x, y)$ can be represented using following expression.

$$\mu_{R_{ij}}(x, y) = \wp\left\{\mu_{L_i}(x), \mu_{M_j}(y)\right\} \quad (5.13)$$

5.3.3 Fuzzy Rule Base

Fuzzy logic based system mimic human intelligence and experience to devise individual fuzzy actions under various conditions. The individual and conditional fuzzy actions are known as *fuzzy rules*. A *fuzzy rule* FR is an IF-THEN statement whose generic structure can be represented as follows.

FR: IF *premise* (*antecedent*), THEN *conclusion* (*consequent*) (5.14)

In this thesis we have used two input-single output fuzzy system so that *premise* (*antecedent*) is a *fuzzy relation* and *conclusion* (*consequent*) is a *fuzzy proposition*.

The entire set of *fuzzy rules* constitutes a *fuzzy rule base*. The size of fuzzy rule base depends upon the number of fuzzy rules, while the number of fuzzy rules depends on the number of input and output variables, and the number of linguistic values associated with each variable. In

general, the formation of fuzzy rules should be such that it tries to preserve basic fuzzy rule characteristics such as consistency, continuity, and completeness.

5.3.4 Fuzzy Rule based Inference

Fuzzy rule based inference mechanism computes the contribution of all activated rules using *fuzzy implication* and *aggregation* procedures.

Fuzzy implication: The procedure for assessing the influence produced by the *premise (antecedent)* part of the activated fuzzy rule on the *conclusion (consequent)* part of it is known as *fuzzy implication*. The fuzzy implication procedure yields a new membership function by modifying output membership function of the activated rule with the input fuzzy relation membership function. In this thesis, we have used *Mamdani based fuzzy implication* procedure.

Fuzzy aggregation: The procedure for concocting the membership functions resulting from fuzzy implication process applied to all activated fuzzy rules is known as fuzzy aggregation. The fuzzy aggregation process yields an equivalent membership function which can be used to determine the crisp output from the fuzzy system as a result of crisp input. In this thesis, we have used *max-min fuzzy aggregation* procedure.

Let *x*, *y*, and *z* are linguistic variables in UODs *X*, *Y* and *Z* such that $x \in X$, $y \in Y$ and $z \in Z$, and $L_i(x)$, $M_j(y)$ and $N_k(z)$ are fuzzy sets having membership functions $\mu_{L_i}(x)$, $\mu_{M_j}(y)$ and $\mu_{N_k}(z)$ associated with a linguistic values L_i, M_j and N_k which *x*, *y* and *z* may attain, respectively. Suppose *x* and *y* are input to the fuzzy system, *z* is output from the fuzzy system, and $\wp$ is the fuzzy operator. The generalized fuzzy rules FR_l, l = 1 to *r*, can be expressed using following expression.

FR_l: IF *x* is L_i $\wp$ *y* is M_j, THEN *z* is N_k (5.15)

Application of *Mamdani based fuzzy implication* to individual activated rules results in output fuzzy sets whose membership functions are given below.

$$\mu_{FR_l}(x, y, z) = min\left\{\wp\left\{\mu_{L_i}(x), \mu_{M_j}(y)\right\}, \mu_{N_k}(z)\right\} \quad (5.16)$$

Using max-min fuzzy aggregation to the membership functions $\mu_{FR_l}(x, y, z)$ results in an equivalent fuzzy set whose membership function is given below.

$$\mu_{FR}(x, y, z) = max\left[\mu_{FR_1}, \mu_{FR_2}, \mu_{FR_3}, \ldots \mu_{FR_{r-1}}, \mu_{FR_r}\right] \tag{5.17}$$

5.3.5 Defuzzification

The result of fuzzy rule based inference process is a fuzzy output set. However, FIS used for FLC structure must give a crisp output value. The procedure for extracting the crisp value from output fuzzy set resulting from fuzzy rule based inference process is known as *defuzzification*. In this thesis, we have used *Centre of Gravity* (COG) method for defuzzification.

For fuzzy system mentioned in section 5.3.4, the defuzzified output value z_{COG} using COG method is given below.

$$z_{\mathrm{COG}} = \frac{\int_{z \in Z}\{(\mu_{FR})z\}dz}{\int_{z \in Z}(\mu_{FR})dz} \tag{5.18}$$

5.4 Design of Fuzzy Logic Control (FLC) for pH Neutralization Process

Basic structure of direct FLC is based on either *Mamdani* FIS or *Takagi-Sugeno* FIS. In this thesis, we have used *Mamdani* FIS based direct FLC structure, as shown in Figure 5.3. Table 5.2 shows summary of Mamdani FIS specifications pertaining to pH neutralization system.

The input variables of fuzzy logic controller for pH neutralization process are error $\mathrm{e(k)}$ and change in error $\mathrm{ce(k)}$ at $\mathrm{k^{th}}$ sampling instant. After dividing the input variables $\mathrm{e(k)}$ and $\mathrm{ce(k)}$ with scaling factors $\mathrm{K_1}$ and $\mathrm{K_2}$, we obtain normalized error and change in error, $\mathrm{e^*(k)}$ and $\mathrm{ce^*(k)}$, as shown in equations (5.19) and (5.20), respectively. The signal multiplexer combines $\mathrm{e^*(k)}$ and $\mathrm{ce^*(k)}$ to give vector $[\mathrm{e^*(k), ce^*(k)}]$ as input to Mamdani FIS. The UOD of input linguistic variables $\mathrm{e^*(k)}$ and $\mathrm{ce^*(k)}$ are [-1, 1], in pH.

$$\mathrm{e^*(k) = e(k)/K_1} \tag{5.19}$$

$$\mathrm{ce^*(k) = ce(k)/K_2} \tag{5.20}$$

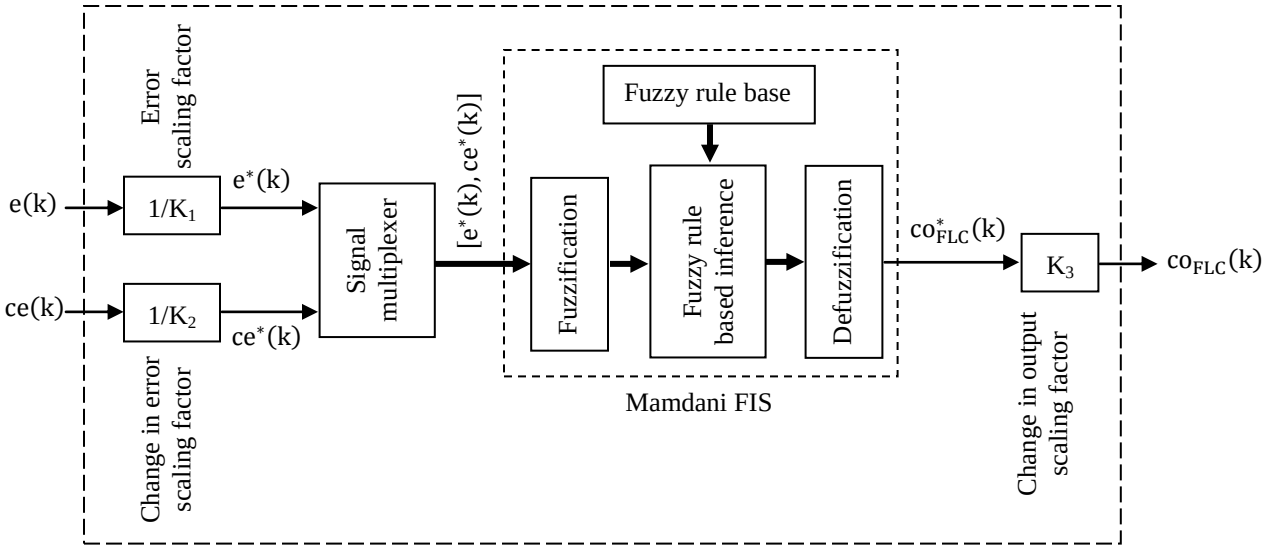


Figure 5.3 Mamdani FIS based fuzzy logic controller structure

Table 5.2 Summary of Mamdani FIS specifications

Quantity	Specification
UOD of input linguistic variables (e^*, ce^*)	[-1, 1] in pH
UOD of output linguistic variable (co^*_{FLC})	[-1, 1] in %
Linguistic values of linguistic variables	Negative Large (NL), Negative Medium (NM), Negative Small (NS), Zero (ZE), Positive Small (PS), Positive Medium (PM), Positive Large (PL)
Shape of membership functions for linguistic values	Trapezoidal for NL, PL Triangular for NM, NS, ZE, PS, PM
Fuzzy operator	AND
Fuzzy implication	Mamdani
Fuzzy aggregation	Max-min
Defuzzification	Centre of Gravity (COG)
Number of rules in fuzzy rule base	49

After multiplying the normalized change in output $co^*_{FLC}(k)$, which is defuzzified output of Mamdani FIS, by the scaling factor K_3, we obtain output variable $co_{FLC}(k)$ of fuzzy logic controller as shown in equation (5.21). The UOD of output linguistic variable $co^*_{FLC}(k)$ is [-1, 1], in %.

$$co_{FLC}(k) = co^*_{FLC}(k) \times K_3 \tag{5.21}$$

The input and output linguistic variables of Mamdani FIS has seven linguistic values each, namely NL, NM, NS, ZE, PS, PM, and PL. The membership functions associated with linguistic values of e^*, ce^*, and co^*_{FLC} are shown in Figures 5.4(a), 5.4(b), and 5.4(c) respectively. The membership functions of NL and PL have trapezoidal shape with vertices [*a*, *b*, *c*, *d*] in reference to Figure 5.2(a) are given as [-100, -99, -1, -0.67] and [0.67, 1, 99, 100] respectively, and equation (5.9) gives their mathematical definitions. The membership functions of NM, NS, ZE, PS, and PM have triangular shape with vertices [*a*, *b*, *c*] in reference to Figure 5.2(b) are given as [-1, -0.67, -0.33], [-0.67, -0.33, 0], [-0.33, 0, 0.33], [0, 0.33, 0.67], and [0.33, 0.67, 1] respectively, and equation (5.10) gives their mathematical definitions.

The fuzzy rule base is the actual repository of experience and knowledge of human operator, and is widely regarded as the heart of a fuzzy logic controller. A fuzzy rule table is a very convenient form of displaying fuzzy rules. To frame fuzzy rules for pH control at a given set-point value pH_{SP} with base flow rate as manipulating variable, consider following two illustrations.

First suppose that $e^*(k)$ = NL and $ce^*(k)$ = NL. Here $e^*(k)$ = NL implies that $pH(k) \gg pH_{SP}$ i.e. the present pH is much away from the set-point, and $ce^*(k)$ = NL implies that $pH(k) \gg pH(k-1)$ i.e. the pH response has tendency to move away from the set-point. Therefore to bring pH back to the set-point we must decrease co^*_{FLC} by large amount so that base flow rate also decreases by large amount.

Next, suppose that $e^*(k)$ = NL and $ce^*(k)$ = PL. Here $e^*(k)$ = NL implies that $pH(k) \gg pH_{SP}$ i.e. the present pH is much away from the set-point, and $ce^*(k)$ = PL implies that $pH(k) \ll pH(k-1)$ i.e. the pH response has tendency to move towards the set-point. Therefore to bring pH back to the set-point we must keep co^*_{FLC} as zero so that base flow rate remains unchanged.

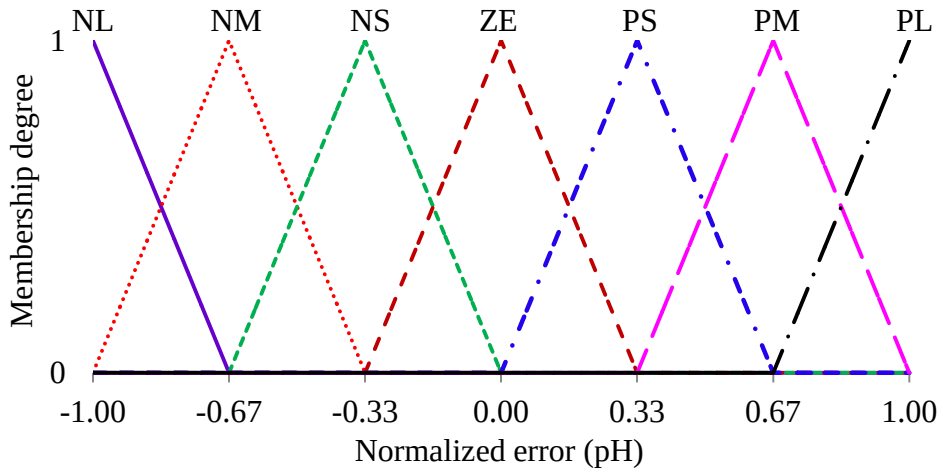


Figure 5.4(a) Fuzzy membership functions for normalized error (e^*)

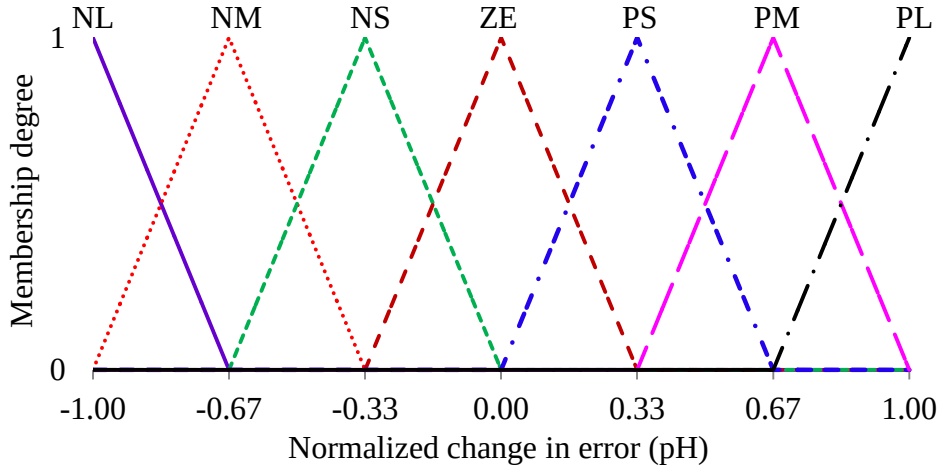


Figure 5.4(b) Fuzzy membership functions for normalized change in error (ce^*)

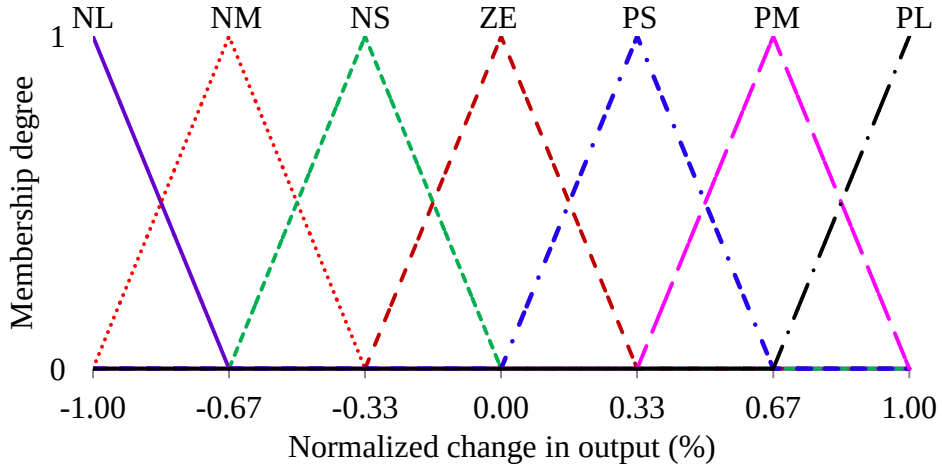


Figure 5.4(c) Fuzzy membership functions for normalized change in output (co^*_{FLC})

There are seven linguistic values for both inputs to Mamdani FIS of fuzzy pH controller. Thus total 7 × 7 = 49 fuzzy rules are needed to completely represent fuzzy rule base. Table 5.3 shows the proposed fuzzy rule table with 49 fuzzy rules for pH control of neutralization process. Since fuzzy rules are culmination of experience and knowledge of an operator, the proposed fuzzy rules ensure the stability of fuzzy controller. The individual fuzzy rule can be represented using following structure.

$$FR_l: \text{IF } e^* \text{ is } m \text{ AND } ce^* \text{ is } n, \text{ THEN } co^*_{FLC} \text{ is } MF_{mn} \tag{5.22}$$

where $l = 7m + n - 7$; $m, n = 1, 2, 3, 4, 5, 6, 7$ represents NL, NM, NS, ZE, PS, PM, PM respectively; MF_{11}, MF_{12}, MF_{13}, MF_{14}, MF_{21}, MF_{22}, MF_{23}, MF_{31}, MF_{32}, MF_{41} represents NL; MF_{15}, MF_{24}, MF_{33}, MF_{42}, MF_{51} represents NM; MF_{16}, MF_{25}, MF_{34}, MF_{43}, MF_{52}, MF_{61} represents NS; MF_{17}, MF_{26}, MF_{35}, MF_{44}, MF_{53}, MF_{62}, MF_{71} represents ZE; MF_{27}, MF_{36}, MF_{45}, MF_{54}, MF_{63}, MF_{72} represents PS; MF_{37}, MF_{46}, MF_{55}, MF_{64}, MF_{73} represents PM; MF_{47}, MF_{56}, MF_{57}, MF_{65}, MF_{66}, MF_{67}, MF_{74}, MF_{75}, MF_{76}, MF_{77} represents PL.

Use of different linguistic values of input variables in fuzzy rule table ensures consistency of fuzzy rules since only one entry needs to be filled for output linguistic value against each rule. Also, the proposed fuzzy rule table shows continuity of fuzzy rule base since output linguistic values of any two successive fuzzy rules are either same or adjacent to each other.

Table 5.3 Fuzzy rule table

		ce^*						
		NL	NM	NS	ZE	PS	PM	PL
e^*	NL	MF_{11}	MF_{12}	MF_{13}	MF_{14}	MF_{15}	MF_{16}	MF_{17}
	NM	MF_{21}	MF_{22}	MF_{23}	MF_{24}	MF_{25}	MF_{26}	MF_{27}
	NS	MF_{31}	MF_{32}	MF_{33}	MF_{34}	MF_{35}	MF_{36}	MF_{37}
	ZE	MF_{41}	MF_{42}	MF_{43}	MF_{44}	MF_{45}	MF_{46}	MF_{47}
	PS	MF_{51}	MF_{52}	MF_{53}	MF_{54}	MF_{55}	MF_{56}	MF_{57}
	PM	MF_{61}	MF_{62}	MF_{63}	MF_{64}	MF_{65}	MF_{66}	MF_{67}
	PL	MF_{71}	MF_{72}	MF_{73}	MF_{74}	MF_{75}	MF_{76}	MF_{77}

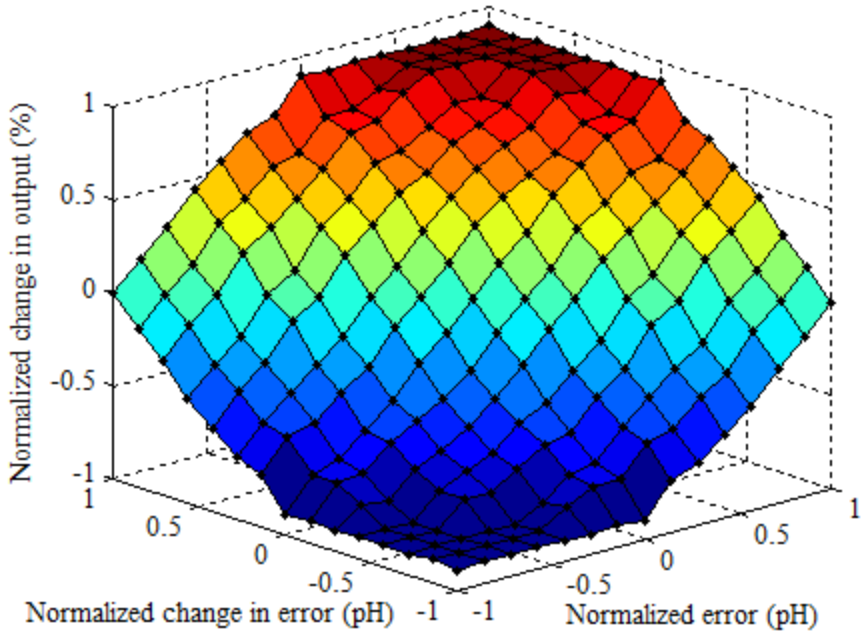


Figure 5.5 Normalized excitation-response plot for Mamdani FIS

Figure 5.5 shows the plot of defuzzified output against normalized inputs variations over their UODs for Mamdani FIS. From the plot, it is quite evident that the proposed Mamdani FIS is nonlinear in nature.

5.5 Feedback Control of Armfield pH Neutralization Process

In this thesis, we have used feedback control of Armfield pH neutralization process in which pH is Controlled Variable (CV), speed of acid pump A (S_a) is Disturbance Variable (DV), and speed of base pump B (S_b) is Manipulated Variable (MV). Under nominal operating conditions, CV is maintained at a set-point value (pH_{SP}) with zero error as input to the pH controller, and manipulated and disturbance variables have values MV0 and DV0 respectively.

As per objectives of thesis discussed in section 1.1, we need to compare performances of PID and fuzzy logic based pH controllers on ANN based dynamic pH model of neutralization process using MATLAB simulations. Figure 5.6 shows block diagram of feedback control of pH neutralization process for simulation using either PID controller of Figure 5.1 or fuzzy logic controller of Figure 5.3. The inputs to pH controller are $e(k)$ and $ce(k)$, and generalized change

of output from pH controller is $co(k)$ which represents $co_{PID}(k)$ for PID and $co_{FLC}(k)$ for fuzzy logic, at k^{th} sampling instant. Manipulating variable is subjected to a saturation limiter in order to maintain $S_b(k)$ within bound [MV_{LB}, MV_{UB}]. The simulated output $pH(k)$ of ANN based dynamic pH model depends upon present inputs $[S_a(k),\ S_b(k)]$, and past three values of inputs-output $[S_a(k-1), S_b(k-1), pH(k-1)]$ to $[S_a(k-3),\ S_b(k-3), pH(k-3)]$. To compare performances of PID and fuzzy logic based pH controllers, fitness function ISE(k) is evaluated for both type of controller under similar operating conditions. The controller which results in minimum ISE is considered to be superior in performance.

Another proposed objective is real-time experimental validation of fuzzy logic based pH controller on neutralization process using LabVIEW for interface and graphical display. Figure 5.7 shows block diagram of feedback control of pH neutralization process for experimental validation on Armfield pH neutralization system using fuzzy logic controller of Figure 5.3. Since we are considering a real, physical, and constantly stirred pH neutralization process, the initial pH range must be maintained within bound [pH_{LB}, pH_{UB}] to ensure approximately same initial conditions.

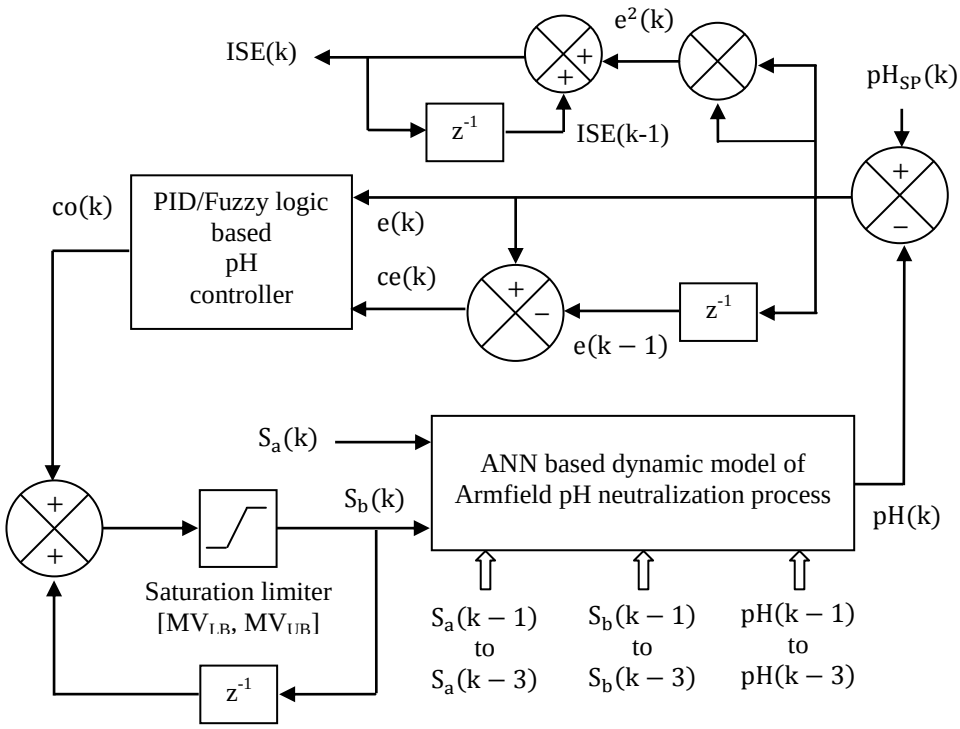


Figure 5.6 Block diagram of feedback control of pH neutralization process for simulation on ANN based dynamic model of Armfield pH neutralization process

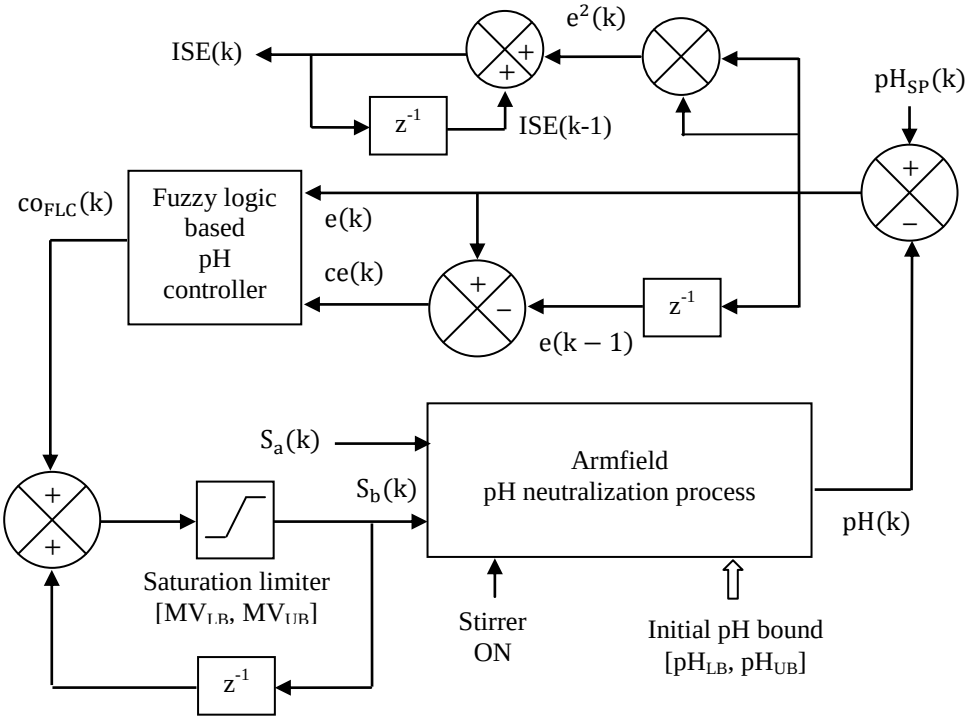


Figure 5.7 Block diagram of feedback control of pH neutralization process for experimental validation on Armfield pH neutralization process

5.6 Tuning of pH Controller Parameters by Global Optimization Techniques

For satisfactory performance, pH controller parameters, namely [K_P, K_I, K_D] of PID controller and [K_1, K_2, K_3] of fuzzy logic controller, must be tuned for given operating conditions. Due to nonlinear nature of pH neutralization process, parameters of linear PID and nonlinear fuzzy logic controllers are tuned using global optimization techniques. Evolutionary and swarm algorithms are popular heuristic based approaches for global optimization techniques. In this thesis we have used Genetic Algorithm (GA) and Differential Evolution (DE) belonging to evolutionary algorithm, and Particle Swarm Optimization (PSO) of swarm algorithm, for offline tuning of PID and fuzzy logic controllers, and also for online tuning of fuzzy logic controller. Next in this section, we have briefly described the flowchart and presented a detailed procedure for single-objective optimization of pH controller parameters using real number coded GA, DE, and PSO techniques independently. By default, the given procedures are valid for offline MATLAB simulations and online LabVIEW implementation, unless otherwise mentioned specifically.

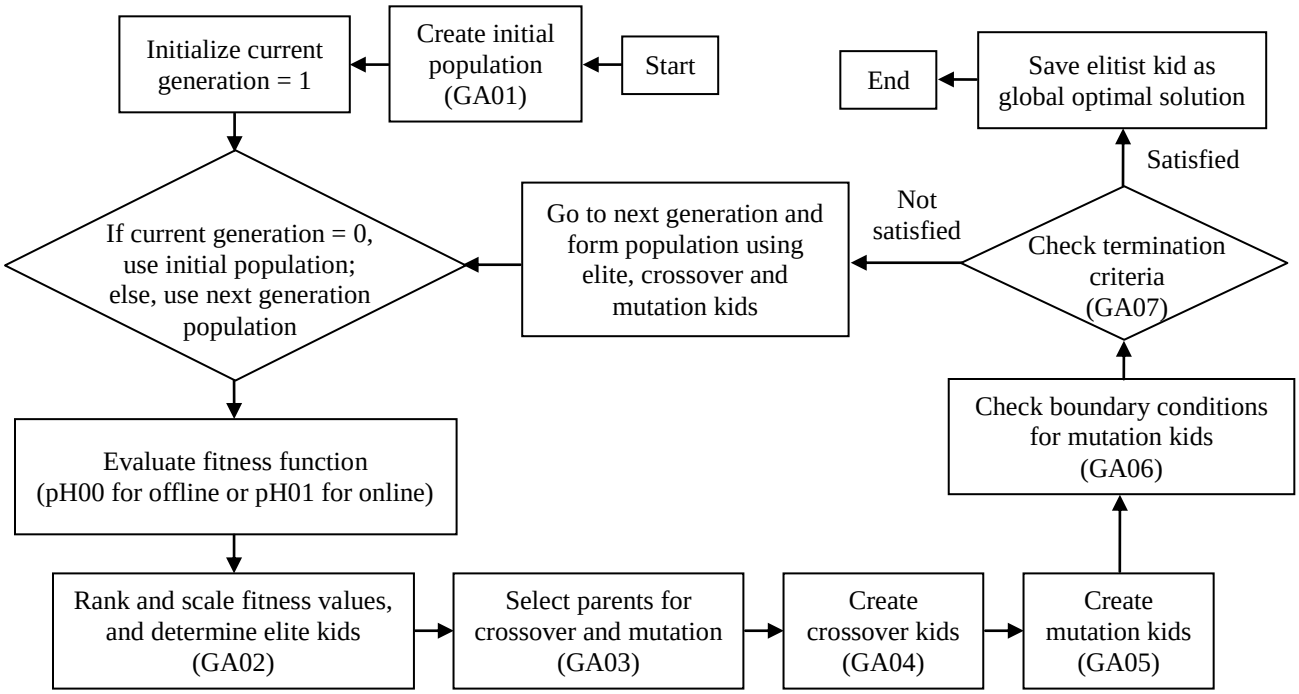


Figure 5.8(a) Flowchart for GA based pH controller parameters optimization

5.6.1 Design of Genetic Algorithm (GA) based Optimized pH Controller

The flowchart of real number coded GA technique for pH controller parameters optimization is shown in Figure 5.8(a). First step in GA optimization is to create initial population (GA01) of type ‘double’ and matrix size L × n where 'L' is the no. of individual population members and 'n' is the no. of variables in each population member. The randomly generated individuals are uniformly distributed over entire initial population range, [K_{PL}, K_{IL}, K_{DL}; K_{PU}, K_{IU}, K_{DU}] for PID controller and [K_{1L}, K_{2L}, K_{3L}; K_{1U}, K_{2U}, K_{3U}] for fuzzy logic controller where the phrases 'L' and 'U' in subscripts represents the lower and upper respectively. Each individual member in the population represents a potential solution to the optimization problem under consideration. The individual population members evolve through successive iterations called generations. In order to evaluate fitness function (pH00 for offline and pH01 for online operations) during each generation, overall ISE is calculated for each individual member of the population. To rank and scale evaluated fitness values, and determine elite kids (GA02), the fitness values of the individuals are ranked between 1 and L such that the elitist individual member having minimum fitness value has the rank as 1, the next elite individual member with next lowest fitness value

has the rank as 2, and similarly, the individual member with highest fitness values has the rank as L. The ranked individual members are assigned scaled values inversely proportional to square root of their rank. The assigned scaled values are used to select parents for crossover and mutation (GA03) operations so that offspring kids can be produced for next generation. In GA03, the stochastic uniform selection operator is represented by a roulette-wheel in which each parent corresponds to a portion of the wheel proportional to its scaled value. The GA moves along the wheel in steps of equal size and, at each step GA allocates a parent to the portion of roulette-wheel it occupies. To create crossover kids (GA04), GA uses scattered crossover operator to combine a pair of parents from allocated parents for crossover operation. To create mutation kids (GA05), GA uses Gaussian mutation operator to apply random changes to a single parent from allocated parents for mutation operation using parameters namely mutation scale, mutation shrink, current generation, and total generation. Since there is a possibility that mutation kids may go out of initial population range, it is required to check boundary conditions for mutation kids (GA06). In case any mutation kid variable is out of range, the concerned variable is regenerated using process similar to GA01. For continuation of GA, it is required to check termination criteria (GA07). If any criteria are satisfied, then elitist kid with least ISE is saved as global optimal solution and process is stopped. Otherwise, elite kids, crossover kids, and mutation kids are combined to create the next generation population, and the complete procedure of fitness function evaluation to next generation population creation is again repeated.

The pseudocodes for GA01, pH00, pH01, GA02, GA03, GA04, GA05, GA06, and GA07 are given in Appendix A9.

Figure 5.8(b) depicts LabVIEW block diagram implementation of GA optimization for fuzzy logic based pH controller on Armfield pH neutralization process. LabVIEW block diagram for various blocks namely pH01, FL01, GA01, GA02, GA03, GA04, GA05, GA06, and GA07 are shown in Figures 5.8(c) to 5.8(k) respectively. The various variables used in Figures 5.8(b) to 5.8(k) are mentioned in Appendix A9.

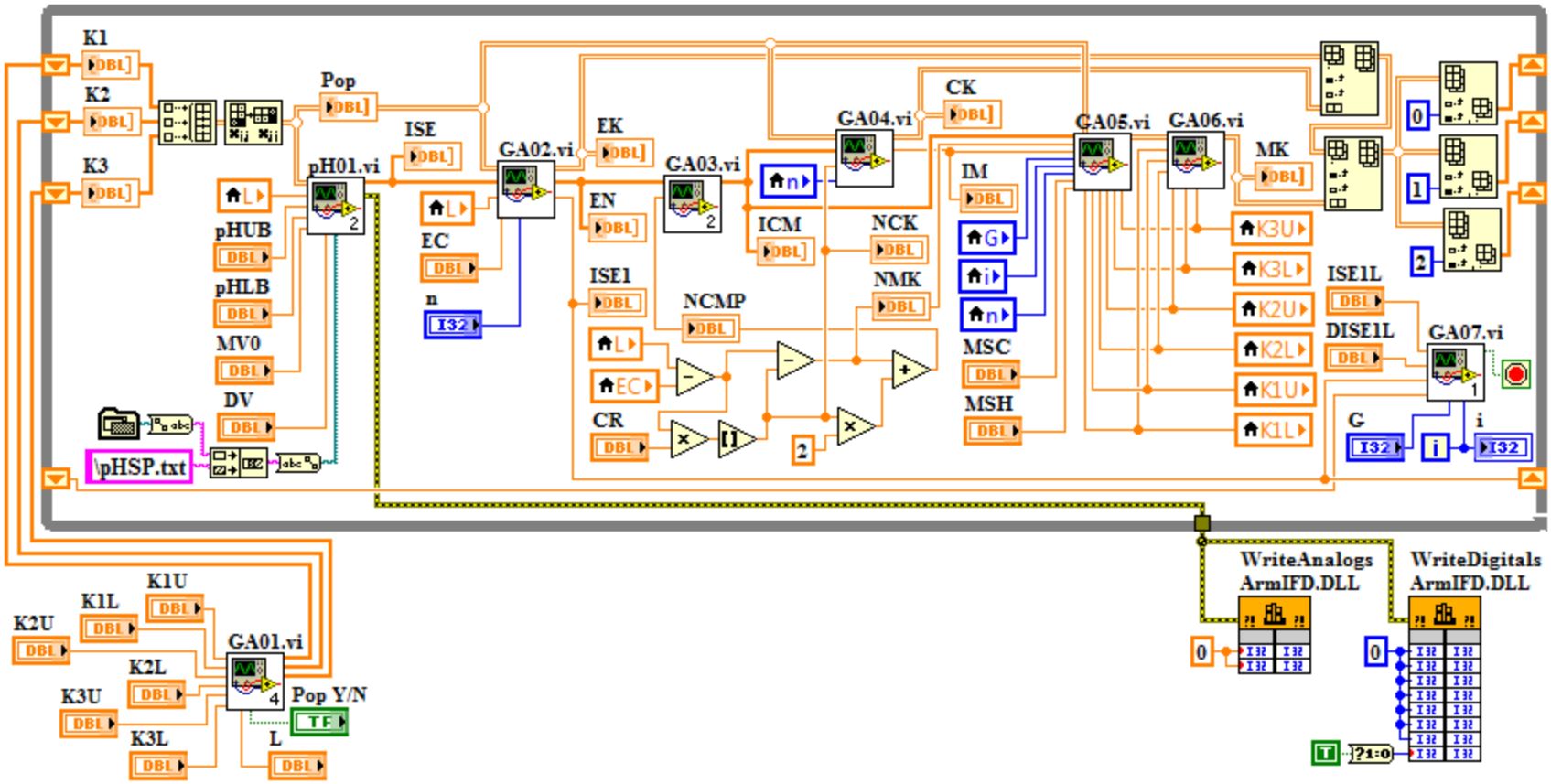


Figure 5.8(b) LabVIEW block diagram implementation of GA optimization for fuzzy logic based pH controller

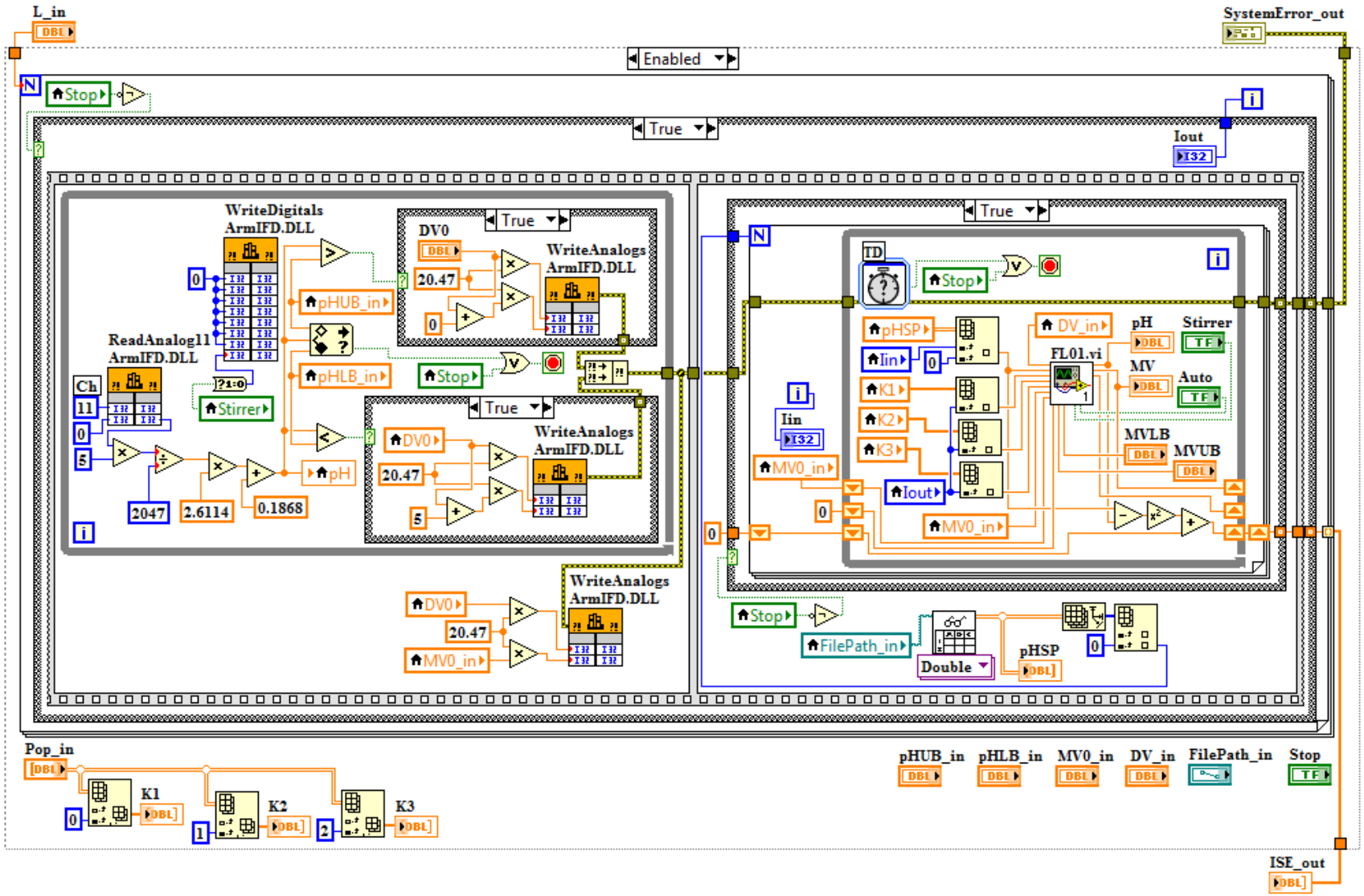


Figure 5.8(c) LabVIEW block diagram to evaluate fitness function (pH01 for online)

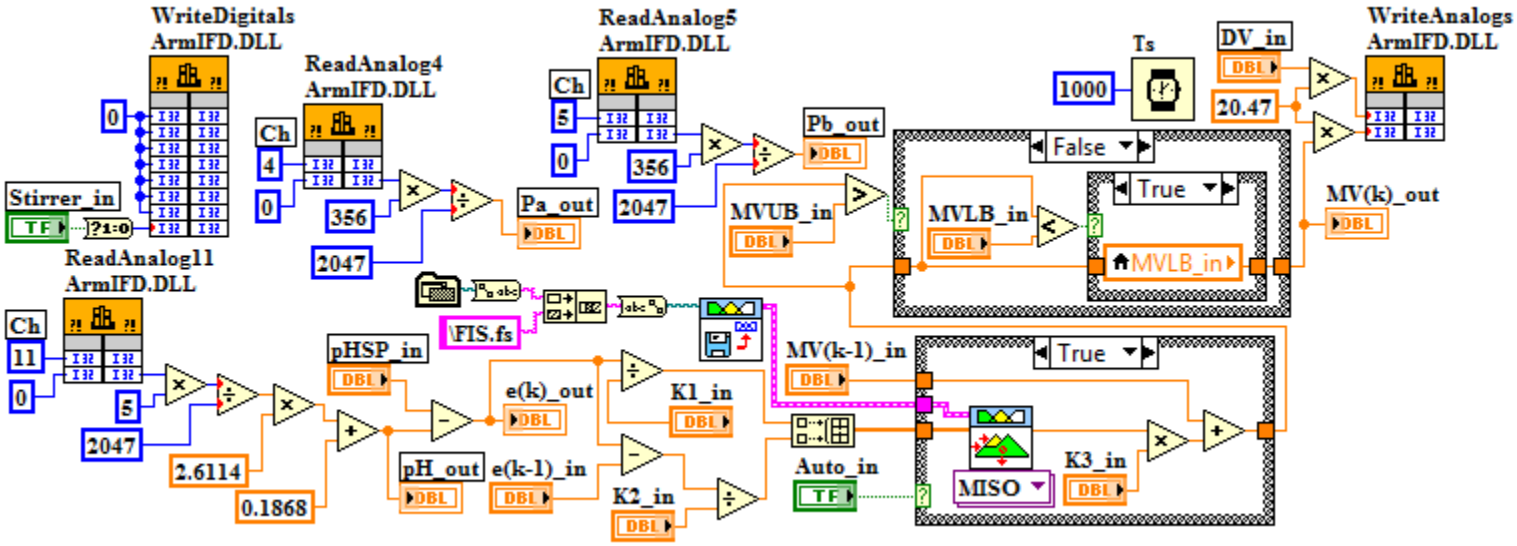


Figure 5.8(d) LabVIEW block diagram for Mamdani FIS based fuzzy logic controller (FL01)

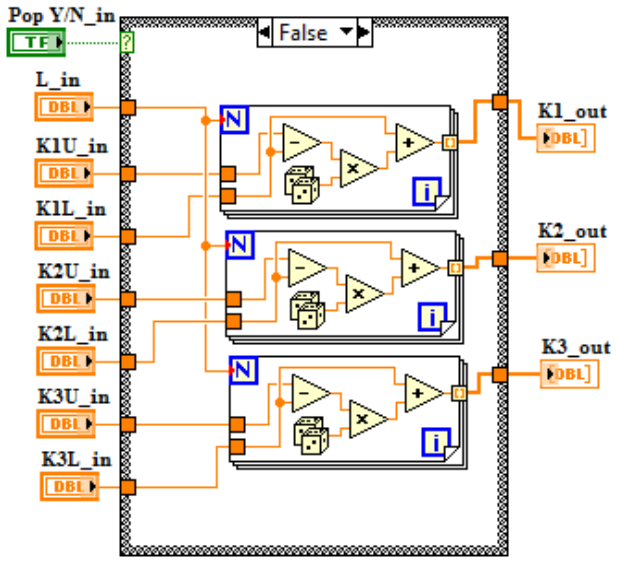


Figure 5.8(e) LabVIEW block diagram to create initial population (GA01)

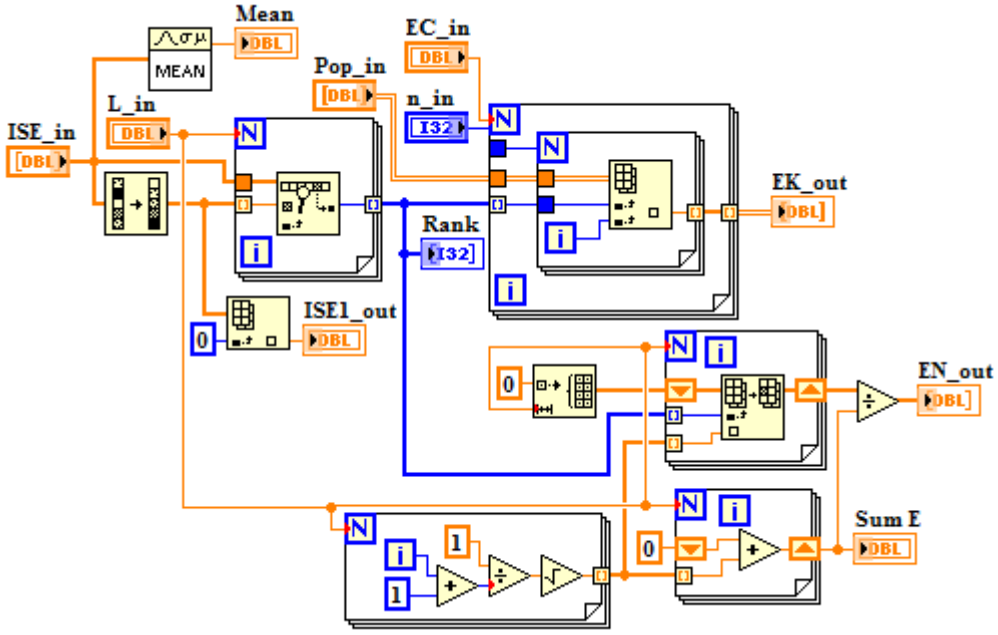


Figure 5.8(f) LabVIEW block diagram to rank and scale fitness values, and determine elite kids (GA02)

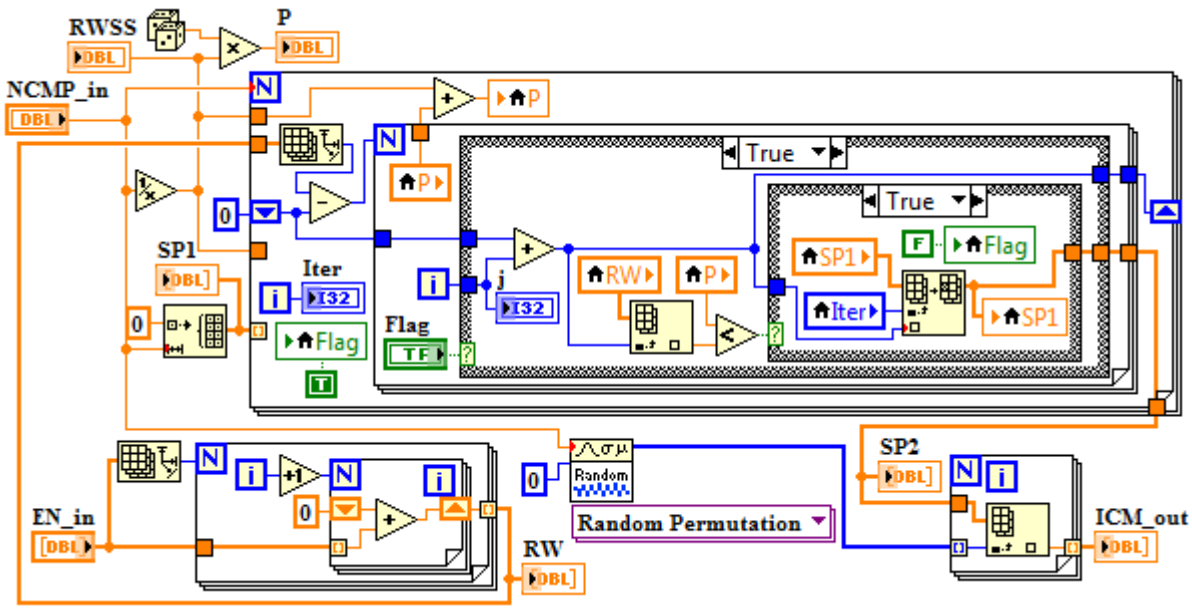


Figure 5.8(g) LabVIEW block diagram to select parents for crossover and mutation (GA03)

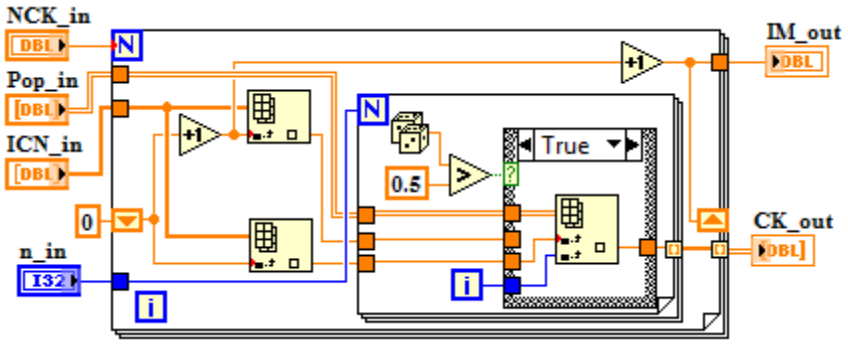


Figure 5.8(h) LabVIEW block diagram to create crossover kids (GA04)

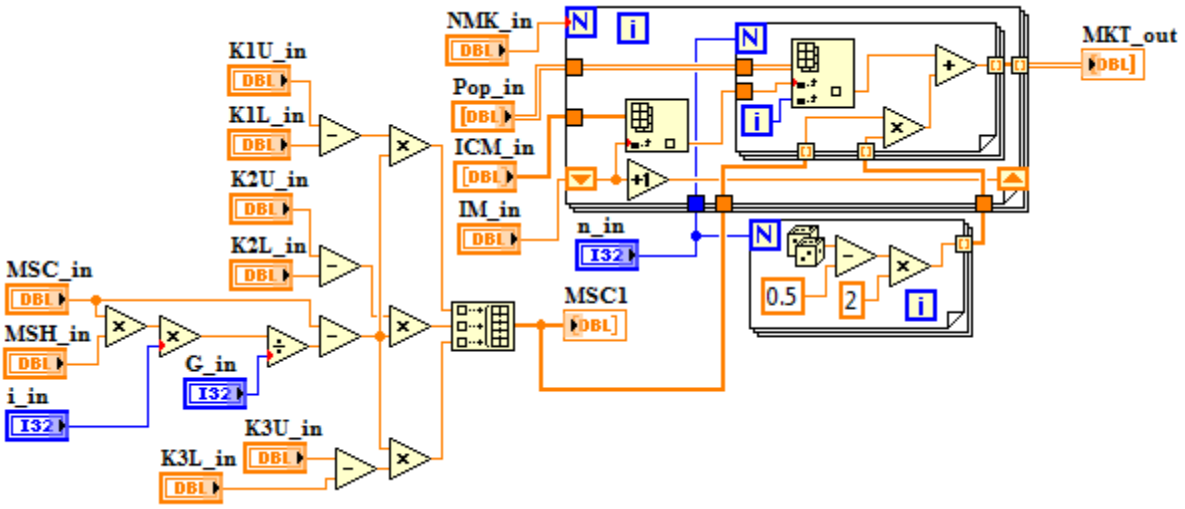


Figure 5.8(i) LabVIEW block diagram to create mutation kids (GA05)

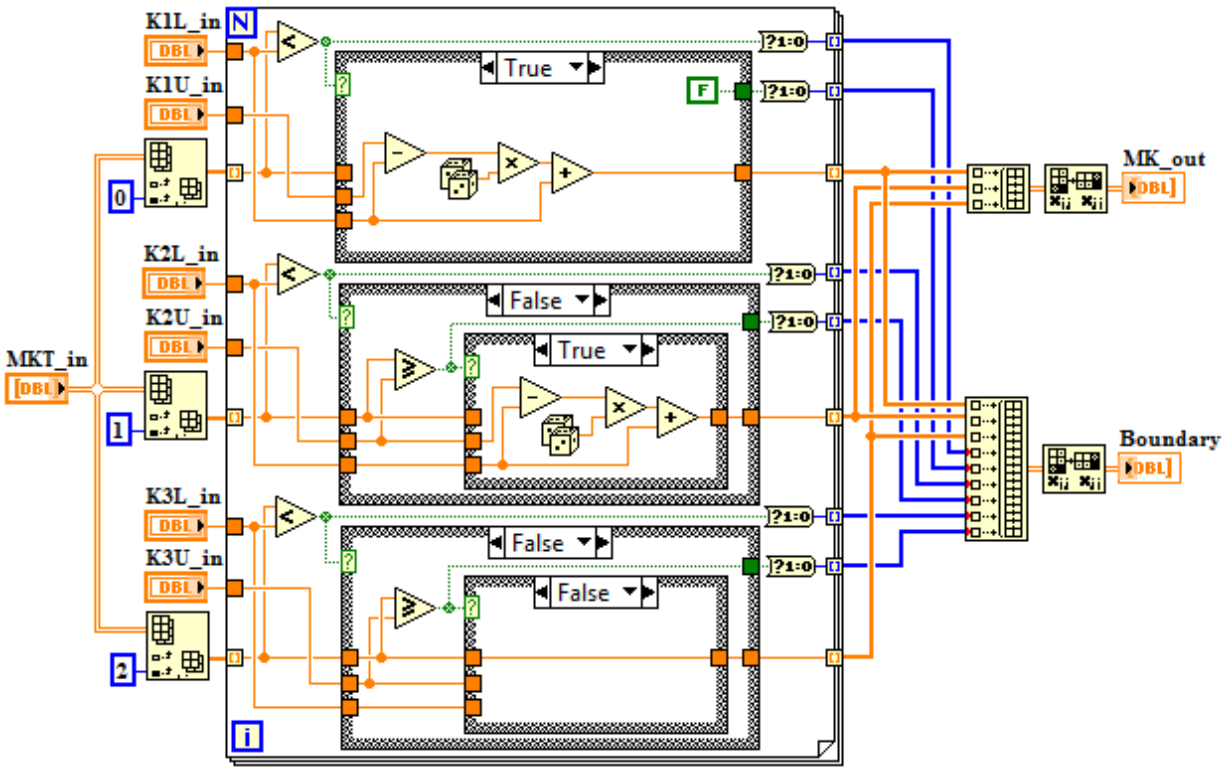


Figure 5.8(j) LabVIEW block diagram to check boundary conditions for mutation kids (GA06)

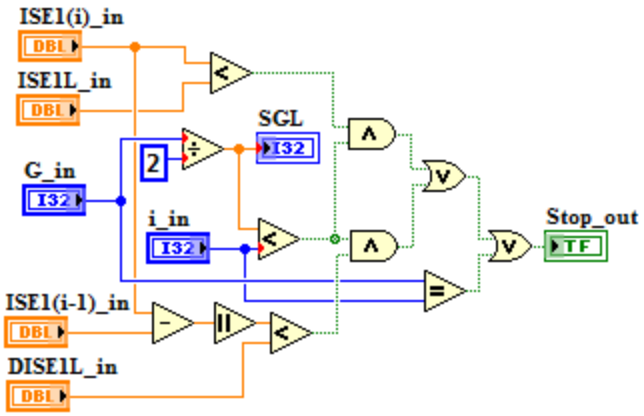


Figure 5.8(k) LabVIEW block diagram to check termination criteria (GA07)

5.6.2 Design of Differential Evolution (DE) based Optimized pH Controller

The flowchart of real number coded DE technique for pH controller parameters optimization is shown in Figure 5.9(a). Similar to GA01, first step in DE optimization is to create initial population (DE01) of type 'double' and matrix size L × n, and the individual population members evolve through successive generations. Also, during each generation, in order to evaluate fitness function (pH00 for offline and pH01 for online operations), overall ISE is calculated for each individual member of the population. To select competitive population members for current generation (DE02), ISE of individual members in present generation are compared with corresponding ISE in the last generation, and the evolved individual member is accepted only in case its fitness value is improved. The most important step in DE is differential mutation in which weighted difference of two population members are added to third one. In order to keep the three population members distinct, it is necessary to subject current population members with random shuffling (DE03). To create trial population with differential mutation and crossover (DE04), the resulting differential mutation quantities and last population members are subjected to crossover. The crossover operation in DE increases the diversity of differential mutation operation. It is required to check boundary conditions for trial population (DE05), and in case any variable is out of range, then new variable value is regenerated using DE01. Similar to GA07, we need to check termination criteria (DE06). On termination, the best member with minimum ISE is saved as global optimal solution.

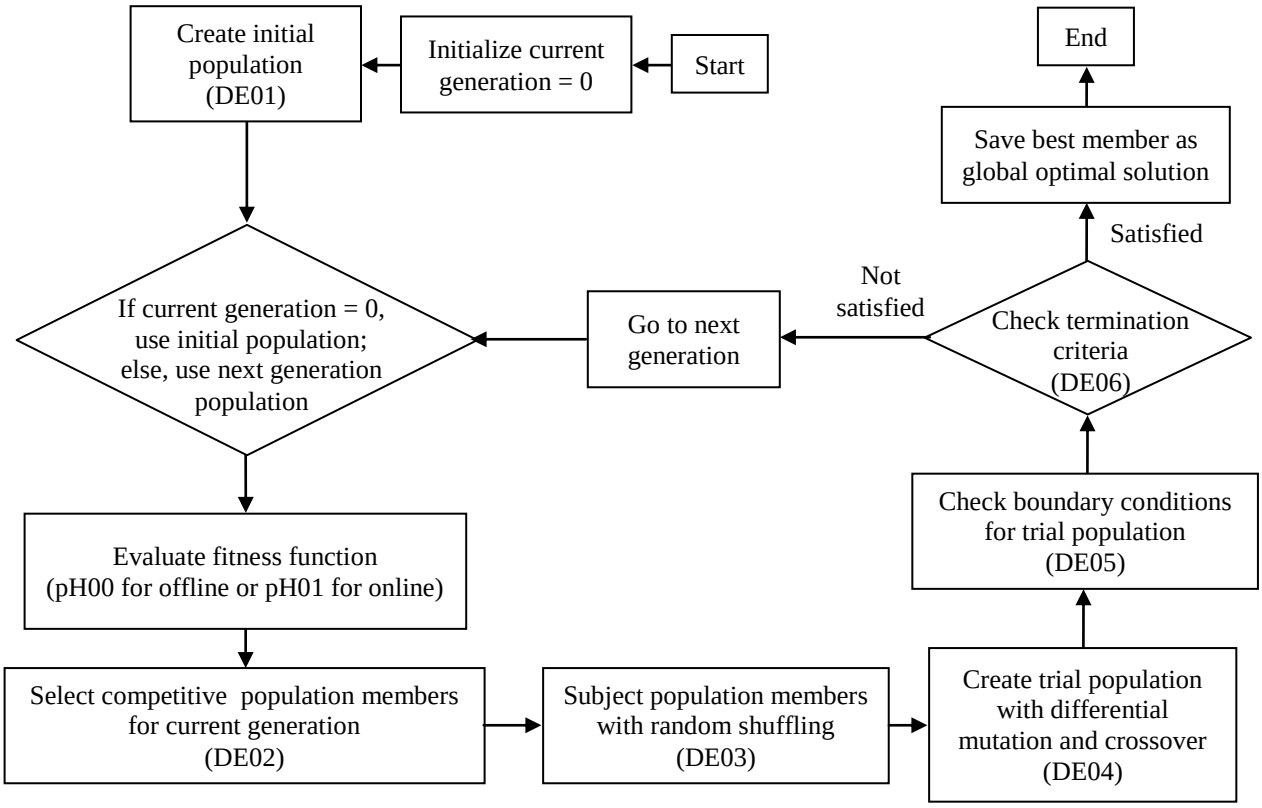


Figure 5.9(a) Flowchart for DE based pH controller parameters optimization

The pseudocodes for DE01 and DE06 are similar to pseudocodes for GA01 and GA07 respectively as given in Appendix A9. The pseudocodes for DE02, DE03, DE04, and DE05 are given in Appendix A10.

Figure 5.9(b) depicts LabVIEW block diagram implementation of DE algorithm for fuzzy logic based pH controller on Armfield pH neutralization process. LabVIEW block diagram for blocks DE01 and DE06 are similar to those shown in Figures 5.8(e) and 5.8(k) respectively. LabVIEW block diagram for remaining blocks namely DE02, DE03, DE04, and DE05 are shown in Figures 5.9(c), 5.9(d), 5.9(e), and 5.9(f) respectively. The various variables used in Figures 5.9(b) to 5.9(f) are mentioned in Appendix A10.

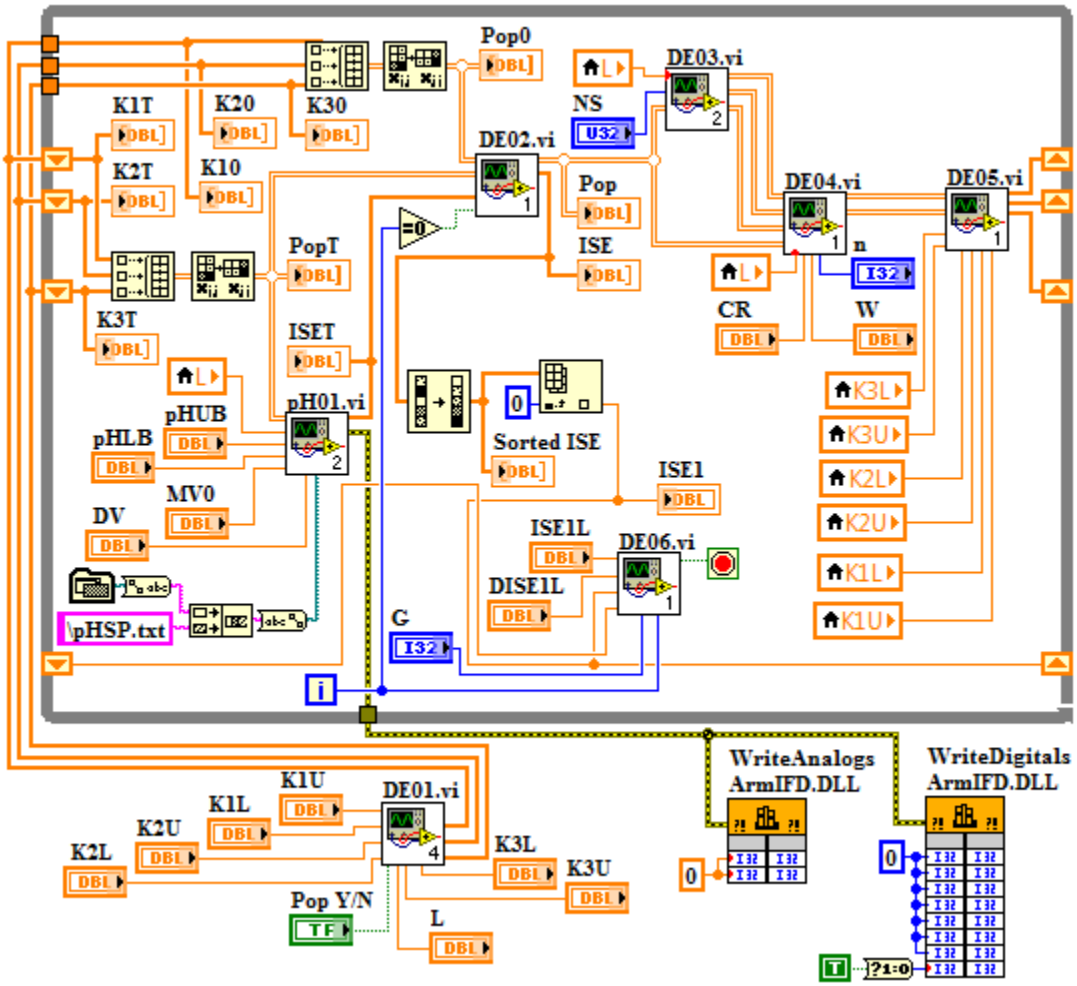


Figure 5.9(b) LabVIEW block diagram implementation of DE algorithm for fuzzy logic based pH controller

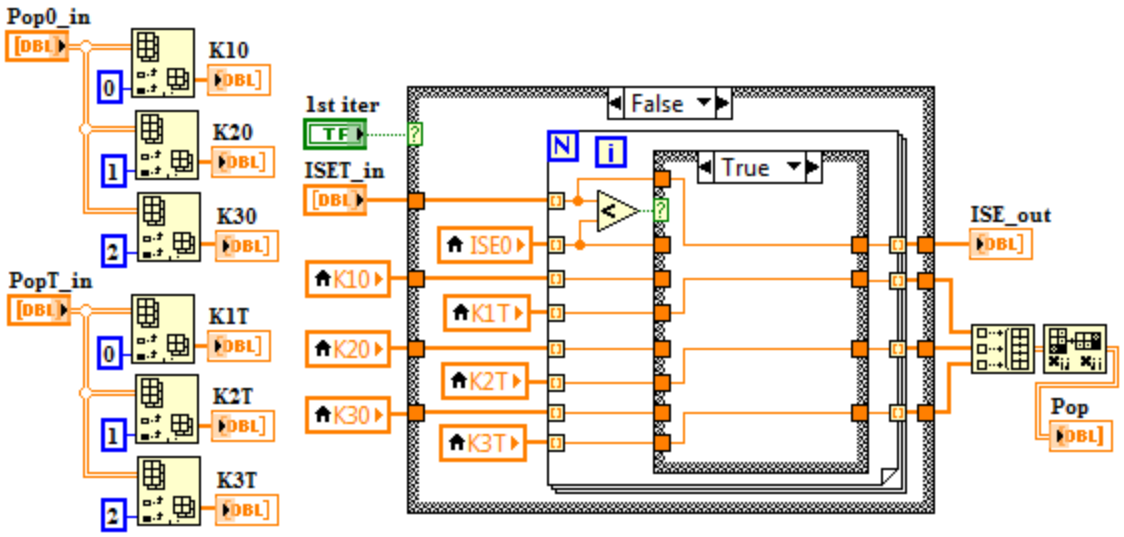


Figure 5.9(c) LabVIEW block diagram to select competitive population members for current generation (DE02)

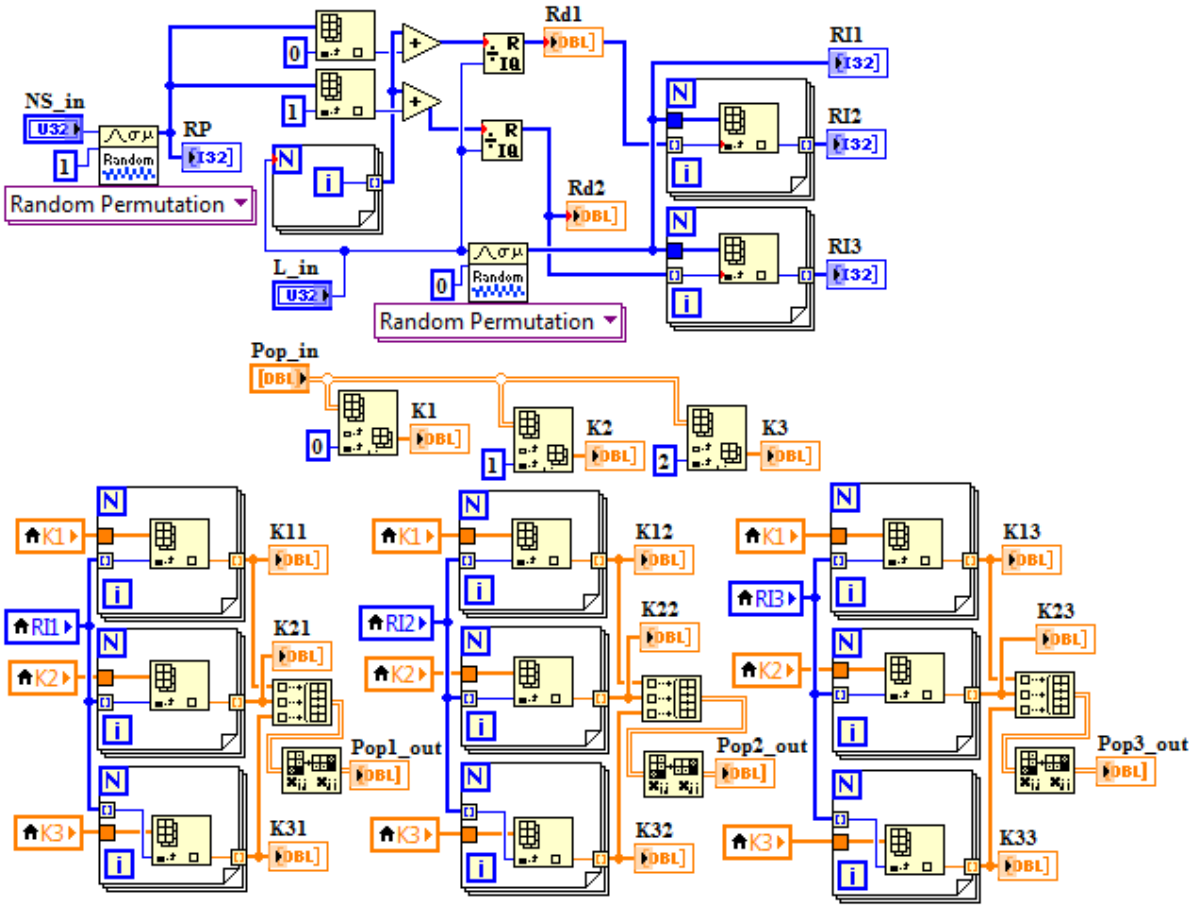


Figure 5.9(d)LabVIEW block diagram to subject population members with random shuffling (DE03)

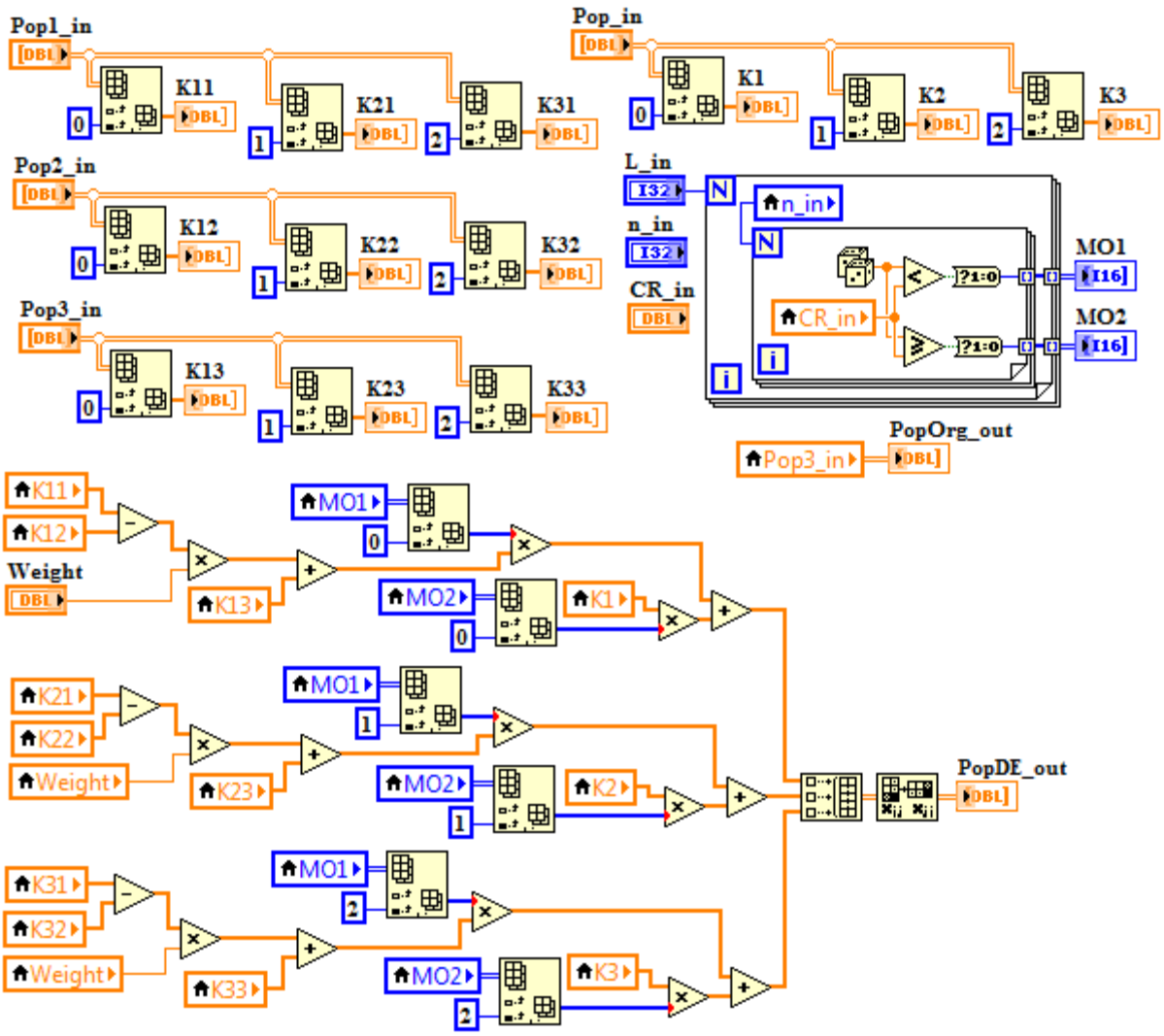


Figure 5.9(e) LabVIEW block diagram to create trial population with differential mutation and crossover (DE04)

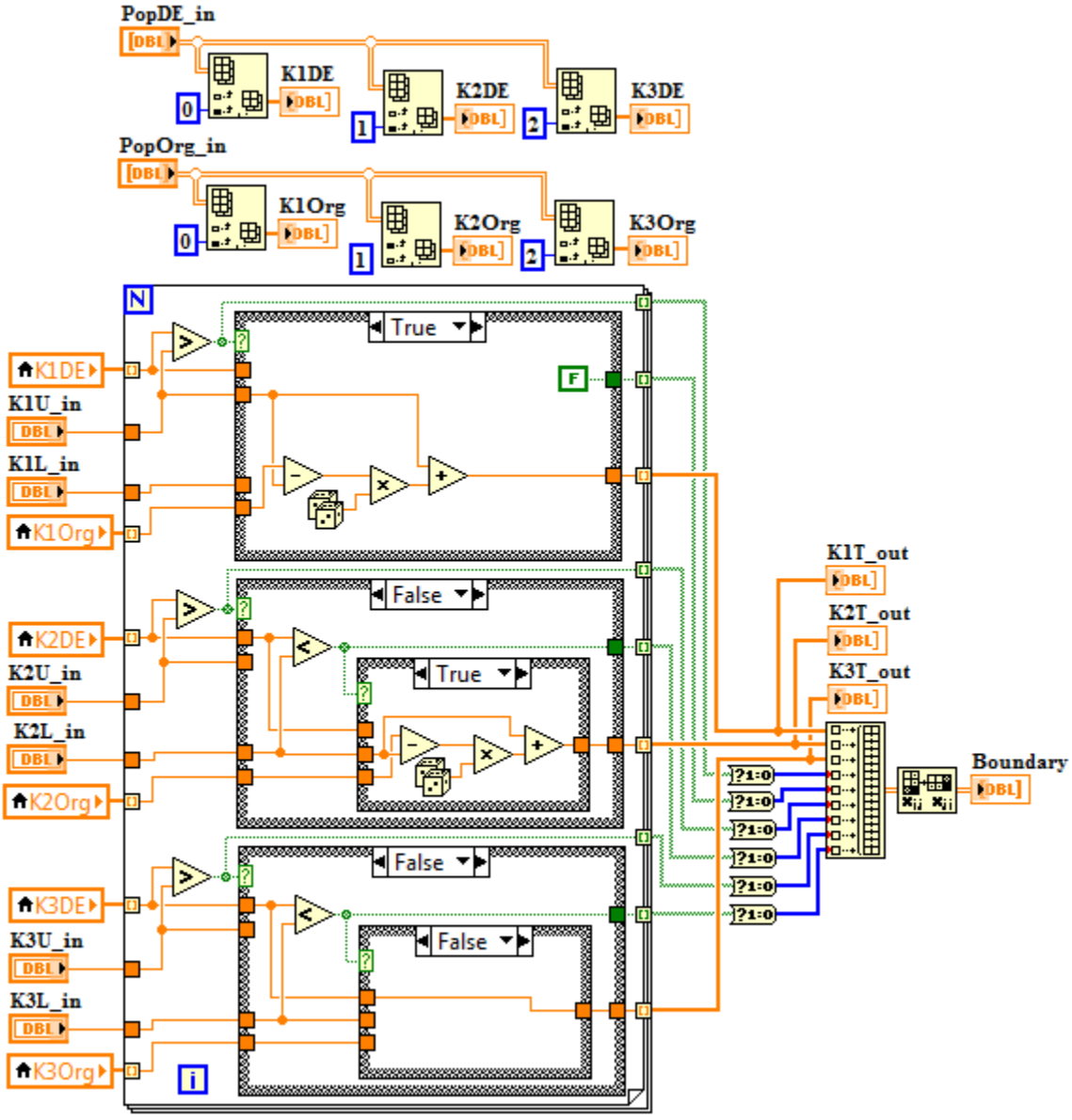


Figure 5.9(f) LabVIEW block diagram to check boundary conditions for trial population (DE05)

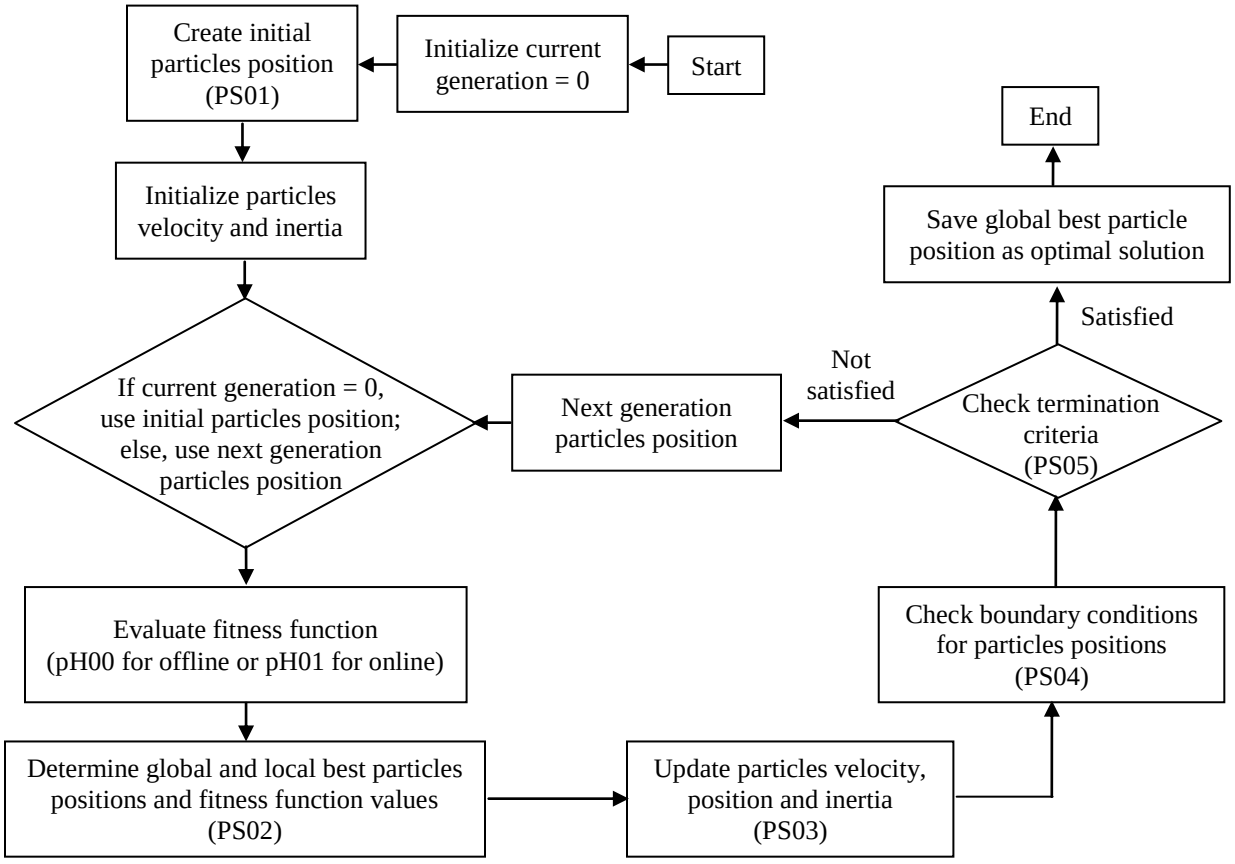


Figure 5.10(a) Flowchart for PSO based pH controller parameters optimization

5.6.3 Design of Particle Swarm Optimization (PSO) based Optimized pH Controller

The flowchart of real number coded PSO technique for pH controller parameters optimization is shown in Figure 5.10(a). First step in PSO is to create initial particles position (PS01) of type 'double' and matrix size L × n, similar to GA01. The particles are assigned an initial velocity with magnitude same as corresponding particle position, and an initial inertia whose magnitude is same for all particles. Over successive generations, the particles update their velocity to reach the global optimal position based on their global and local best positions which are decided on the basis of fitness function values. In order to evaluate fitness function (pH00 for offline and pH01 for online operations) during each generation, overall ISE is calculated for each individual particle of the population. To determine global and local best particles positions and fitness function values (PS02), it is required for algorithm to compare present fitness function values

with past values. In a particular generation, a particle is regarded as global best if it has lowest ever ISE, and local best if it has ISE less than that of corresponding particle in immediate preceding generation. For next generation, it is required to update particles velocity, position and inertia (PS03). To update individual particle velocity following three terms are added: First - cureent inertia multiplied with current velocity; Second - local best position minus current position is multiplied with a random number and a cognitive attraction constant; Third - global best position minus current position is multiplied with a random number and a social attraction constant. The current position is added with updated velocity in order to obtain updated individual particle position. The particle inertia is reduced with successive generations till it reaches lowest bound value. It is required to check boundary conditions for particles position (PS04), and in case any variable is out of range, then new variable value is regenerated using PS01. Similar to GA07, we need to check termination criteria (PS05). On termination, the global best particle position with minimum ISE is saved as global optimal solution.

The pseudocodes for PS01, PS04, and PS05 are similar to pseudocodes for GA01, GA06, and GA07 as given in Appendix A9. The pseudocodes for PS02 and PS03 are given in Appendix A11.

Figure 5.10(b) depicts LabVIEW block diagram implementation of PSO algorithm for fuzzy logic based pH controller on Armfield pH neutralization process. LabVIEW block diagram for blocks PS01, PS04, and PS05 are similar to those shown in Figures 5.8(e), 5.8(j), and 5.8(k) respectively. LabVIEW block diagram for remaining blocks namely PS02 and PS03 are shown in Figures 5.10(c) and 5.10(d) respectively. The various variables used in Figures 5.10(b) to 5.10(d) are mentioned in Appendix A11.

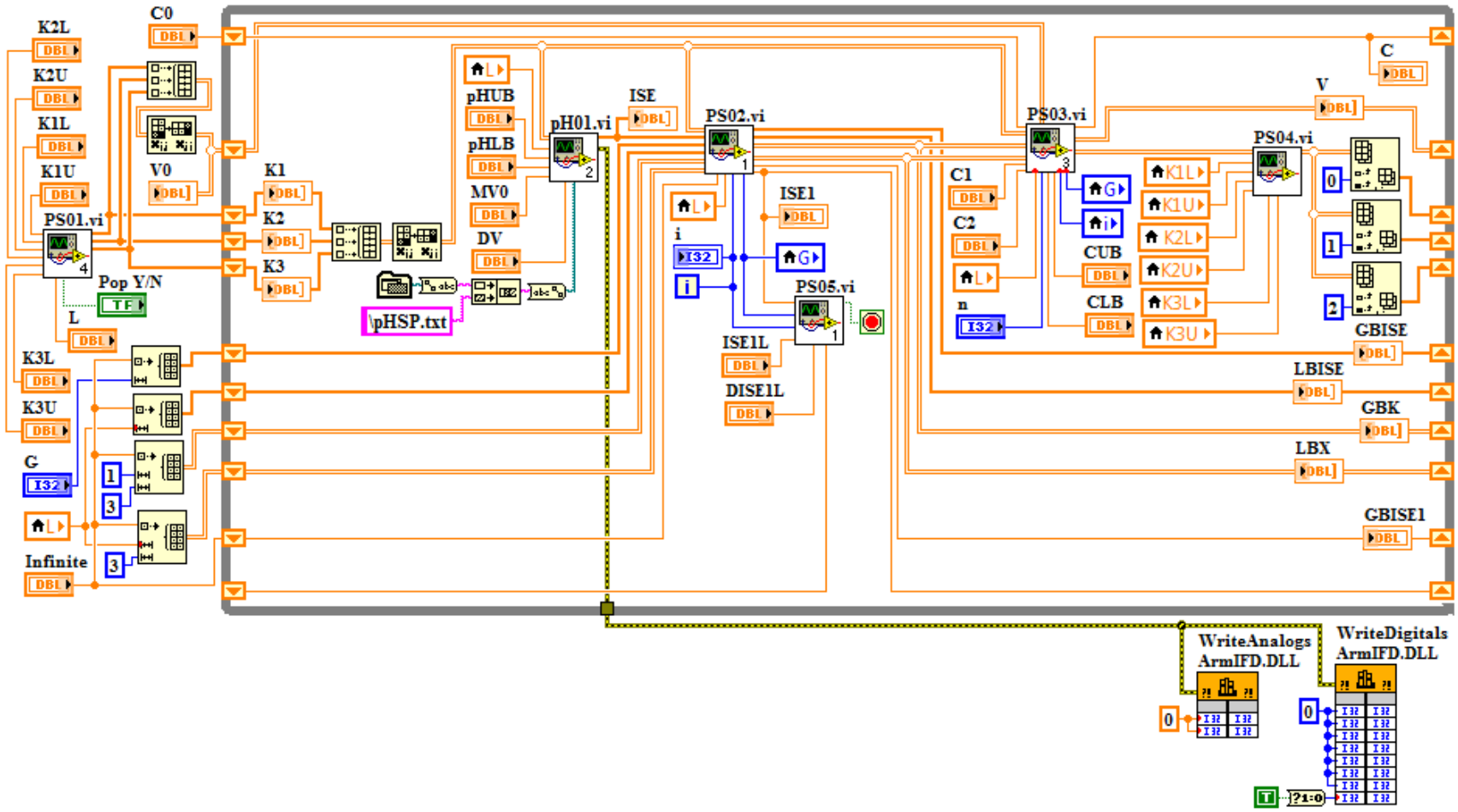


Figure 5.10(b) LabVIEW block diagram implementation of PSO algorithm for fuzzy logic based pH controller

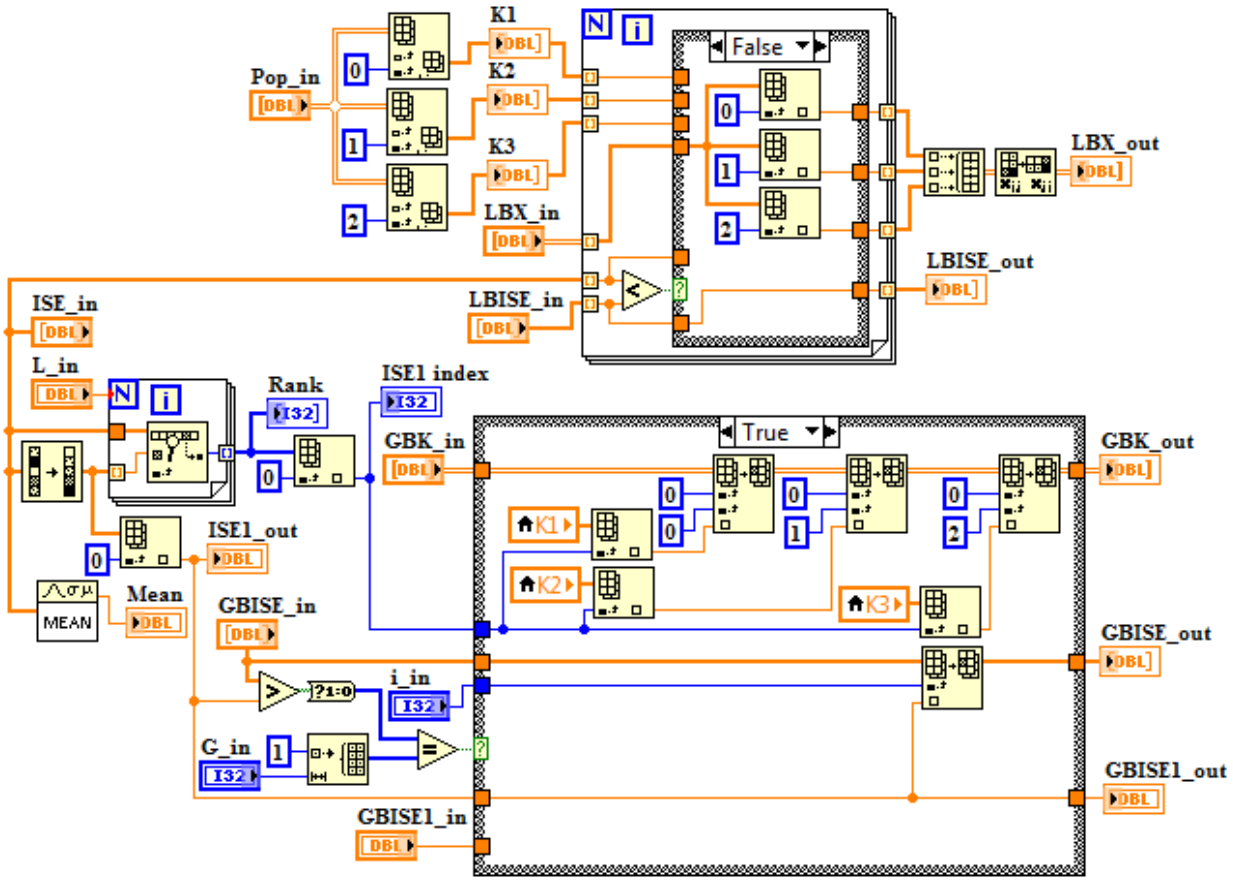


Figure 5.10(c) LabVIEW block diagram to determine global and local best particles positions and fitness function values (PS02)

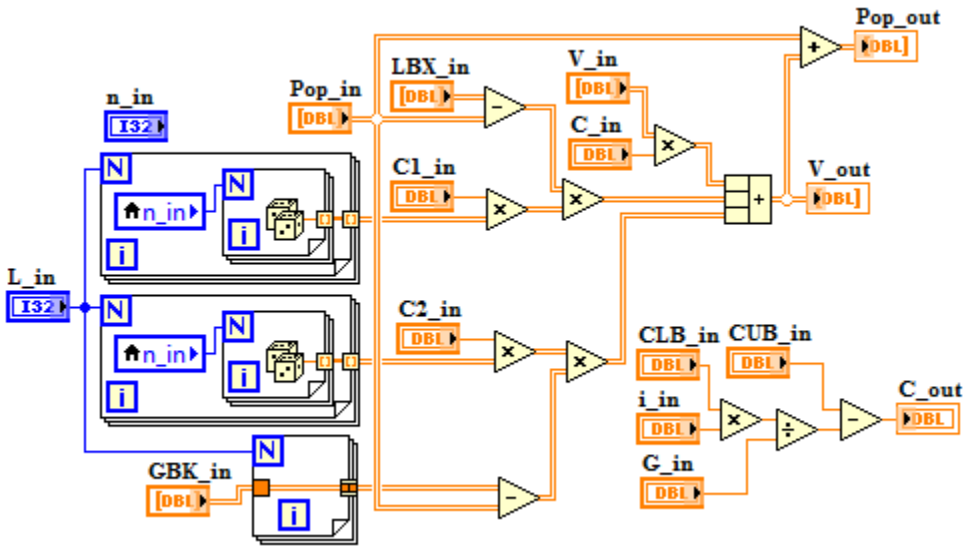


Figure 5.10(d) LabVIEW block diagram to update particles velocity, position and inertia (PS03)

Table 5.4(a) Common parameters for GA, DE, and PSO techniques based pH control system

Parameters	Values
Number of variables (n)	3
Population size (L)	20
Range of population members (for GA and DE)/range of particles positions (for PSO): $[K_{PL}, K_{IL}, K_{DL}; K_{PU}, K_{IU}, K_{DU}]$ for PID $[K_{1L}, K_{2L}, K_{3L}; K_{1U}, K_{2U}, K_{3U}]$ for FLC	[6 0.2 0.02;30 1 0.1] for PID [6 0.2 6;30 1 30] for FLC
Number of generations (G)	50 for offline simulation 5 for online validation
Minimum ISE desired (ISE1L)	0
Absolute difference between minimum ISE for two successive generations (DISE1L)	0
Random numbers	[0, 1]
Steady-state values at setpoint $(pH_{SP})_{initial}$ used as initial conditions for offline simulations: $[S_a(1), S_b(1), pH(1); S_a(2), S_b(2), pH(2);$ $S_a(3), S_b(3), pH(3)]$	[35, 39.59, 5.95;35, 39.34, 5.97; 35, 39.56, 5.96] at $(pH_{SP})_{initial} = 6$
	[35, 38.29, 6.99;35, 37.99, 7.01; 35, 37.99, 7.01] at $(pH_{SP})_{initial} = 7$
	[35, 39.97, 7.96;35, 39.72, 7.97; 35, 39.60, 7.98] at $(pH_{SP})_{initial} = 8$
	[35, 39.42, 9.01;35, 39.24, 9.02; 35, 39.22, 9.02] at $(pH_{SP})_{initial} = 9$
pH values range used as initial conditions for online validation, $[pH_{LB}, pH_{UB}]$	$pH_{UB} = (pH_{SP})_{initial} + 0.1$ $pH_{LB} = (pH_{SP})_{initial} - 0.1$
Nominal setting of manipulating variable (MV) for online validation, MV0	38.5
Nominal setting of disturbance variable (DV) for online validation, DV0	35
Saturation limiter for MV, $[MV_{LB}, MV_{UB}]$	[18, 80]

Table 5.4(a) continued

Step changes in setpoint, from $(pH_{SP})_{initial}$ to $(pH_{SP})_{final}$, for servo operation with DV = DV0	6 to 7, 7 to 8, 8 to 9, 9 to 8, 8 to 7, 7 to 6
Step changes in disturbance variable, from $(DV)_{initial}$ to $(DV)_{final}$, for regulatory operation at each setpoint $(pH_{SP})_{final}$ = 6, 7, 8, 9	35 to 30, 30 to 35, 35 to 40, 40 to 35
Time duration for servo operation i.e. for each step change from $(pH_{SP})_{initial}$ to $(pH_{SP})_{final}$	200 seconds each for offline simulation and online validation
Time duration for regulatory operation i.e. for each step changes from $(DV)_{initial}$ to $(DV)_{final}$	100 seconds each for offline simulation and online validation

Table 5.4(b) Additional parameters for GA, DE, and PSO techniques based pH control system

Technique	Parameters	Values
GA	Elite count (EC)	2
	Crossover rate (CR)	0.8
	Mutation scale (MSC)	0.1
	Mutation shrink (MSH)	0.1
DE	Weight factor (Weight)	1
	Crossover rate (CR)	0.8
	Number of random shuffling (N_S)	5
PSO	Initial particle inertia (C_0)	0.9
	Lower and upper bounds for particle inertia, [C_{LB}, C_{UB}]	[0.4, 0.9]
	Cognitive attraction (C_1)	0.5
	Social attraction (C_2)	2

Table 5.5 Cases of servo-regulatory (SR) operations in pH neutralization process

Particulars	Definition
Case 1: Servo-regulatory operation SR1 $(pH_{SP})_{initial}$ = 6, $(DV)_{initial}$ = 35%	$pH_{SP}(i) = \{7 \quad \text{for} \quad 1 \le i \le 600$ $S_a(i) = \begin{cases} 35 & \text{for} \quad 1 \le i \le 200 \\ 30 & \text{for} \quad 201 \le i \le 300 \\ 35 & \text{for} \quad 301 \le i \le 400 \\ 40 & \text{for} \quad 401 \le i \le 500 \\ 35 & \text{for} \quad 501 \le i \le 600 \end{cases}$
Case 2: Servo-regulatory operation SR2 $(pH_{SP})_{initial}$ = 7, $(DV)_{initial}$ = 35%	$pH_{SP}(i) = \{8 \quad \text{for } 601 \le i \le 1200$ $S_a(i) = \begin{cases} 35 & \text{for} \quad 601 \le i \le 800 \\ 30 & \text{for} \quad 801 \le i \le 900 \\ 35 & \text{for} \quad 901 \le i \le 1000 \\ 40 & \text{for} \quad 1001 \le i \le 1100 \\ 35 & \text{for} \quad 1101 \le i \le 1200 \end{cases}$
Case 3: Servo-regulatory operation SR3 $(pH_{SP})_{initial}$ = 8, $(DV)_{initial}$ = 35%	$pH_{SP}(i) = \{9 \quad \text{for } 1201 \le i \le 1800$ $S_a(i) = \begin{cases} 35 & \text{for} \quad 1201 \le i \le 1400 \\ 30 & \text{for} \quad 1401 \le i \le 1500 \\ 35 & \text{for} \quad 1501 \le i \le 1600 \\ 40 & \text{for} \quad 1601 \le i \le 1700 \\ 35 & \text{for} \quad 1701 \le i \le 1800 \end{cases}$
Case 4: Servo-regulatory operation SR4 $(pH_{SP})_{initial}$ = 9, $(DV)_{initial}$ = 35%	$pH_{SP}(i) = \{8 \quad \text{for } 1801 \le i \le 2400$ $S_a(i) = \begin{cases} 35 & \text{for} \quad 1801 \le i \le 2000 \\ 30 & \text{for} \quad 2001 \le i \le 2100 \\ 35 & \text{for} \quad 2101 \le i \le 2200 \\ 40 & \text{for} \quad 2201 \le i \le 2300 \\ 35 & \text{for} \quad 2301 \le i \le 2400 \end{cases}$
Case 5: Servo-regulatory operation SR5 $(pH_{SP})_{initial}$ = 8, $(DV)_{initial}$ = 35%	$pH_{SP}(i) = \{7 \quad \text{for } 2401 \le i \le 3000$ $S_a(i) = \begin{cases} 35 & \text{for} \quad 2401 \le i \le 2600 \\ 30 & \text{for} \quad 2601 \le i \le 2700 \\ 35 & \text{for} \quad 2701 \le i \le 2800 \\ 40 & \text{for} \quad 2801 \le i \le 2900 \\ 35 & \text{for} \quad 2901 \le i \le 3000 \end{cases}$
Case 6: Servo-regulatory operation SR6 $(pH_{SP})_{initial}$ = 7, $(DV)_{initial}$ = 35%	$pH_{SP}(i) = \{6 \quad \text{for } 3001 \le i \le 3600$ $S_a(i) = \begin{cases} 35 & \text{for} \quad 3001 \le i \le 3200 \\ 30 & \text{for} \quad 3201 \le i \le 3300 \\ 35 & \text{for} \quad 3301 \le i \le 3400 \\ 40 & \text{for} \quad 3401 \le i \le 3500 \\ 35 & \text{for} \quad 3501 \le i \le 3600 \end{cases}$

5.7 Discussion on Simulation and Experimental Results

In this thesis we have used GA, DE, and PSO techniques independently to optimize parameters of PID and fuzzy logic controllers for servo-regulatory (SR) operations in pH neutralization process. The common parameters for GA, DE, and PSO techniques based pH control system are given in Table 5.4(a). Table 5.4(b) gives additional parameters for GA, DE, and PSO techniques based pH control system, respectively.

5.7.1 Offline Optimized PID and FLC Schemes for Servo and Regulatory Operations

Comparison of optimized PID and FLC schemes is based on SR operations as illustrated in Table 5.5. SR operations has been divided in six cases, namely SR1, SR2, SR3, SR4, SR5, and SR6 as mentioned in Table 5.5, to cover dynamic pH range from 6 to 9. For servo operations, step changes in setpoint, from $(pH_{SP})_{initial}$ to $(pH_{SP})_{final}$ i.e. 6 to 7, 7 to 8, 8 to 9, 9 to 8, 8 to 7, and 7 to 6, are introduced for 200 seconds with nominal acid flow rate as S_a = DV0 i.e. 35%. For regulatory operations, step changes in disturbance variable, from $(DV)_{initial}$ to $(DV)_{final}$ i.e. 35% to 30%, 30% to 35%, 35% to 40%, and 40% to 35%, are introduced consecutively for 100 seconds at each setpoint $(pH_{SP})_{final}$ i.e. 7, 8, 9, 8, 7, and 6. Thus, SRi, where i = 1, 2, 3, 4, 5, and 6, involves servo operation of 200 seconds followed by regulatory operations of 400 seconds. Therefore, entire duration for SR operations is 3600 seconds.

Optimization of PID controller is carried out offline using MATLAB software in order to obtain optimal values of K_P, K_I, and K_D. The performance of optimized PID controller is evaluated on Armfiled pH neutralization process using LabVIEW software. Following observations are made for offline optimized PID controller.

(i) Offline GA optimization gives best simulated ISE as 80.6441 and optimized parameters as $[K_P, K_I, K_D]$ = [27.4506, 0.7494, 0.0970]. Figure 5.11(a) shows the best and mean values, and Figure 5.11(b) shows initial and final population, for offline GA optimization. Since mutation operator brings random changes in population members, the final population members are still diversified, although less than initial population members. Figures 5.12(a) and 5.12(b) shows simulated as well as experimental pH response and pump speed variations, using offline GA optimized PID controller. Experimental validation gives total ISE as 411.7163 of which nearly 88% is accounted together for SR1, SR5, and SR6 operations. Tables 5.6(a) and 5.6(b) gives

performance summary of simulated and experimental responses of offline GA optimized PID controller for SR operations on the basis of ISE and maximum overshoot or undershoot.

(ii) Offline DE optimization gives best simulated ISE as 80.6644 and optimized parameters as $[K_P, K_I, K_D]$ = [27.1793, 0.7598, 0.1000]. Figure 5.13(a) shows the best and mean values, and Figure 5.13(b) shows initial and final population, for offline DE optimization. Since next generation population members proposed using DE algorithm must perform better than the existing population members, the final population members are fully converged near best solution. Figures 5.14(a) and 5.14(b) shows simulated as well as experimental pH response and pump speed variations, using offline DE optimized PID controller. Experimental validation gives total ISE as 577.2561 of which nearly 92% is accounted together for SR1, SR5, and SR6 operations. Tables 5.6(a) and 5.6(b) gives performance summary of simulated and experimental responses of offline DE optimized PID controller for SR operations on the basis of ISE and maximum overshoot or undershoot.

(iii) Offline PSO gives best simulated ISE as 80.7496 and optimized parameters as $[K_P, K_I, K_D]$ = [27.1825, 0.7770, 0.0897]. Figure 5.15(a) shows the best and mean values, and Figure 5.15(b) shows initial and final particles positions, for offline PSO. Since random number multipliers in cognitive and social attractions brings random changes in particles velocities, the final particles position are still diversified, although less than initial particles position. Figures 5.16(a) and 5.16(b) shows simulated as well as experimental pH response and pump speed variations, using offline PSO based PID controller. Experimental validation gives total ISE as 126.1982 of which nearly 72% is accounted together for SR1, SR5, and SR6 operations. Tables 5.6(a) and 5.6(b) gives performance summary of simulated and experimental responses of offline PSO based PID controller for SR operations on the basis of ISE and maximum overshoot or undershoot.

Optimization of fuzzy logic controller is also carried out offline using MATLAB software in order to obtain optimal values of K_1, K_2, and K_3. The performance of optimized fuzzy logic controller is evaluated on Armfiled pH neutralization process using LabVIEW software. Following observations are made for offline optimized fuzzy logic controller.

(i) Offline GA optimization gives best simulated ISE as 73.8843 and optimized parameters as $[K_1, K_2, K_3]$ = [25.2150, 0.6495, 14.1478]. Figure 5.17(a) shows the best and mean values, and

Figure 5.17(b) shows initial and final population, for offline GA optimization. Figures 5.18(a) and 5.18(b) shows simulated as well as experimental pH response and pump speed variations, using offline GA optimized fuzzy logic controller. Experimental validation gives total ISE as 80.3776 of which nearly 55% is accounted together for SR1, SR5, and SR6 operations. Tables 5.7(a) and 5.7(b) gives performance summary of simulated and experimental responses of offline GA optimized fuzzy logic controller for SR operations on the basis of ISE and maximum overshoot or undershoot.

(ii) Offline DE optimization gives best simulated ISE as 71.9779 and optimized parameters as $[K_1, K_2, K_3] = [29.9802, 0.7527, 16.1632]$. Figure 5.19(a) shows the best and mean values, and Figure 5.19(b) shows initial and final population, for offline DE optimization. Figures 5.20(a) and 5.20(b) shows simulated as well as experimental pH response and pump speed variations, using offline DE optimized fuzzy logic controller. Experimental validation gives total ISE as 67.9637 of which nearly 54% is accounted together for SR1, SR5, and SR6 operations. Tables 5.7(a) and 5.7(b) gives performance summary of simulated and experimental responses of offline DE optimized fuzzy logic controller for SR operations on the basis of ISE and maximum overshoot or undershoot.

(iii) Offline PSO gives best simulated ISE as 72.2608 and optimized parameters as $[K_1, K_2, K_3] = [29.3358, 0.7326, 15.7505]$. Figure 5.21(a) shows the best and mean values, and Figure 5.21(b) shows initial and final population, for offline PSO. Figures 5.22(a) and 5.22(b) shows simulated as well as experimental pH response and pump speed variations, using offline PSO based fuzzy logic controller. Experimental validation gives total ISE as 67.9266 of which nearly 50% is accounted together for SR1, SR5, and SR6 operations. Tables 5.7(a) and 5.7(b) gives performance summary of simulated and experimental responses of offline PSO based fuzzy logic controller for SR operations on the basis of ISE and maximum overshoot or undershoot.

From the above discussions it is clear that offline optimized PID controller failed the experimental validation test whereas offline optimized fuzzy logic controller qualified the same test. Here offline optimization of PID and fuzzy logic controllers uses ANN based dynamic model which has its own limitation in representing actual real-time dynamics of the pH neutralization process. The linear PID controller with constant parameters makes it unsuitable for control of pH process because of its inability to adapt with highly nonlinear and unknown

process dynamics. A better approach would be to use gain-scheduled, adaptive PID controller. However requirement of fine control needs large number of PID controller gain adjustment variables which will make the control system complex. On the other hand, the nonlinear fuzzy logic controller uses membership functions whose degree varies with error and change in error. The variation in membership degree and choice of appropriate rules based on error and change in error allows variation in fuzzy logic controller output and makes fuzzy logic controller as intelligent.

5.7.2 Offline Optimized Piecewise FLC Schemes for Servo and Regulatory Operations

Offline optimization of piecewise FLC for pH neutralization process is carried out using MATLAB software in order to obtain optimal values of K_1, K_2, and K_3 for SR1, SR2, SR3, SR4, SR5, and SR6 operations. The performance of optimized piecewise fuzzy logic controller is evaluated on Armfiled pH neutralization process using LabVIEW software. Following observations are made for offline optimized piecewise fuzzy logic controller.

(i) Offline GA optimization gives best simulated ISE as 64.7311 and optimized parameters as [K_1, K_2, K_3] = [28.3724, 0.8441, 15.3606] for SR1, [26.7230, 0.7950, 19.0777] for SR2, [8.1809, 0.4979, 29.4345] for SR3, [29.7914, 0.9453, 23.2001] for SR4, [25.2150, 0.5883, 12.2996] for SR5, [28.3210, 0.9251, 22.4813] for SR6. Figure 5.23(a) shows the best and mean values, and Figure 5.23(b) shows initial and final population, for offline GA optimization. Figures 5.24(a) and 5.24(b) shows simulated as well as experimental pH response and pump speed variations, using offline GA optimized piecewise fuzzy logic controller. Experimental validation gives total ISE as 66.1221. Tables 5.8(a) and 5.8(b) gives performance summary of simulated and experimental responses of offline GA optimized piecewise fuzzy logic controller for SR operations on the basis of ISE and maximum overshoot or undershoot.

(ii) Offline DE optimization gives best simulated ISE as 64.3618 and optimized parameters as [K_1, K_2, K_3] = [29.9970, 0.6999, 15.0027] for SR1, [29.9977, 0.7949, 20.3831] for SR2, [9.0666, 0.5237, 29.9997] for SR3, [30.0000, 0.9713, 23.7756] for SR4, [29.9599, 0.6773, 13.9229] for SR5, [28.7984, 0.9236, 22.3913] for SR6. Figure 5.25(a) shows the best and mean values, and Figure 5.25(b) shows initial and final population, for offline DE optimization. Figures 5.26(a) and 5.26(b) shows simulated as well as experimental pH response and pump speed variations,

using offline DE optimized piecewise fuzzy logic controller. Experimental validation gives total ISE as 64.3561. Tables 5.8(a) and 5.8(b) gives performance summary of simulated and experimental responses of offline DE optimized piecewise fuzzy logic controller for SR operations on the basis of ISE and maximum overshoot or undershoot.

(iii) Offline PSO gives best simulated ISE as 64.4981 and optimized parameters as $[K_1, K_2, K_3]$ = [29.8878, 0.6907, 14.7302] for SR1, [27.4920, 0.7853, 19.5650] for SR2, [9.2593, 0.5044, 29.8706] for SR3, [29.8229, 0.9458, 23.0210] for SR4, [27.3910, 0.6286, 13.0664] for SR5, [28.3064, 0.9400, 22.4326] for SR6. Figure 5.27(a) shows the best and mean values, and Figure 5.27(b) shows initial and final population, for offline PSO. Figures 5.28(a) and 5.28(b) shows simulated as well as experimental pH response and pump speed variations, using offline PSO based piecewise fuzzy logic controller. Experimental validation gives total ISE as 64.8058. Tables 5.8(a) and 5.8(b) gives performance summary of simulated and experimental responses of offline PSO based piecewise fuzzy logic controller for SR operations on the basis of ISE and maximum overshoot or undershoot.

From the above discussions it is clear that offline optimized piecewise fuzzy logic controller qualifies the experimental validation test. In comparison with offline optimized FLC for SR operations in section 5.7.1, use of offline optimized piecewise FLC for SR operations brings ISE values down by amount 14.2555 for GA, 3.6076 for DE, and 3.1208 for PSO. Further it is evident from Tables 5.8(a) and 5.8(b) that pH control for SR1 and SR5 cases are most challenging task.

5.7.3 Online Optimized piecewise FLC Schemes for Servo and Regulatory Operations

Online optimization of piecewise FLC for pH neutralization process is carried out using LabVIEW software in order to obtain optimal values of K_1, K_2, and K_3 for SR1 and SR5 operations. Following observations are made for online optimized piecewise fuzzy logic controller.

(i) Online GA optimization gives best experimental ISE as 8.7858 for SR1, and 10.8342 for SR5, and optimized parameters as $[K_1, K_2, K_3]$ = [22.6760, 0.6350, 27.9280] for SR1, and [28.5220, 0.7570, 23.1800] for SR5. Figures 5.29(a) and 5.31(a) shows the best and mean values, and Figures 5.29(b) and 5.31(b) shows initial and final population, for online GA optimization of

SR1 and SR5 operations respectively. We know that GA assumes the population member with least ISE in a particular generation as elitist member. For online experimentation, it is possible that a population member has different ISE values over successive generations, as shown in Figures 5.29(a) and 5.31(a). Figures 5.30(a) and 5.32(a) shows pH response, and Figures 5.30(b) and 5.32(b) shows pump speed variations obtained experimentally, using online GA optimized piecewise fuzzy logic controller for SR1 and SR5 operations respectively. Table 5.9 gives performance summary of experimental responses of online GA optimized piecewise fuzzy logic controller for SR1 and SR5 operations on the basis of ISE and maximum overshoot or undershoot.

(ii) Online DE optimization gives best experimental ISE as 9.7916 for SR1, and 11.3888 for SR5, and optimized parameters as $[K_1, K_2, K_3]$ = [26.9410, 0.8540, 28.0570] for SR1, and [28.6350, 0.7690, 29.7860] for SR5. Figures 5.33(a) and 5.35(a) shows the best and mean values, and Figures 5.33(b) and 5.35(b) shows initial and final population, for online DE optimization of SR1 and SR5 operations respectively. Figures 5.34(a) and 5.36(a) shows pH response, and Figures 5.34(b) and 5.36(b) shows pump speed variations obtained experimentally, using online DE optimized piecewise fuzzy logic controller for SR1 and SR5 operations respectively. Table 5.9 gives performance summary of experimental responses of online DE optimized piecewise fuzzy logic controller for SR1 and SR5 operations on the basis of ISE and maximum overshoot or undershoot.

(iii) Online PSO gives best experimental ISE as 9.3424 for SR1, and 9.4614 for SR5, and optimized parameters as $[K_1, K_2, K_3]$ = [25.5040, 0.6420, 27.9730] for SR1, and [28.8680, 0.7840, 23.8840] for SR5. Figures 5.37(a) and 5.39(a) shows the best and mean values, and Figures 5.37(b) and 5.39(b) shows initial and final population, for online PSO of SR1 and SR5 operations respectively. Figures 5.38(a) and 5.40(a) shows pH response, and Figures 5.38(b) and 5.40(b) shows pump speed variations obtained experimentally, using online PSO based piecewise fuzzy logic controller for SR1 and SR5 operations respectively. Table 5.9 gives performance summary of experimental responses of online PSO based piecewise fuzzy logic controller for SR1 and SR5 operations on the basis of ISE and maximum overshoot or undershoot.

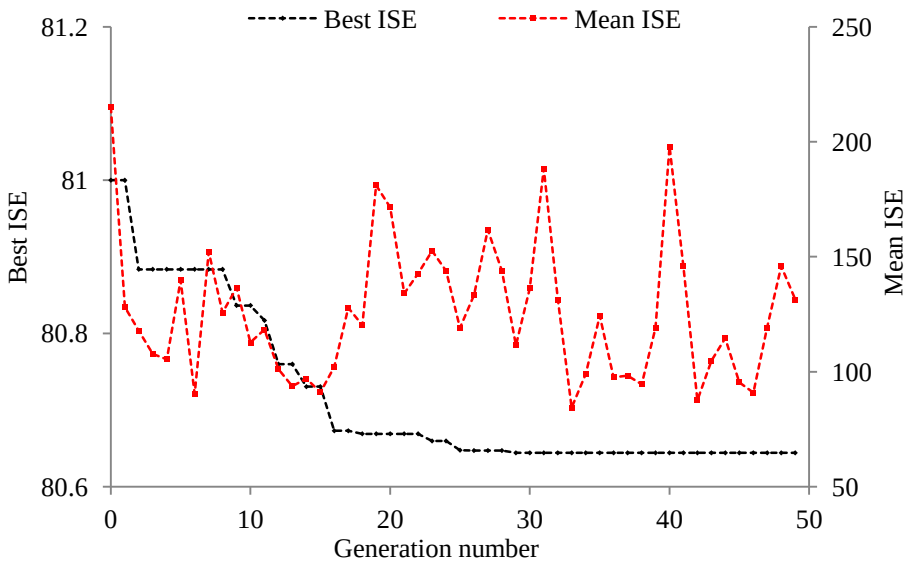


Figure 5.11(a) Best and mean ISE values of offline GA optimization based PID control for SR operations

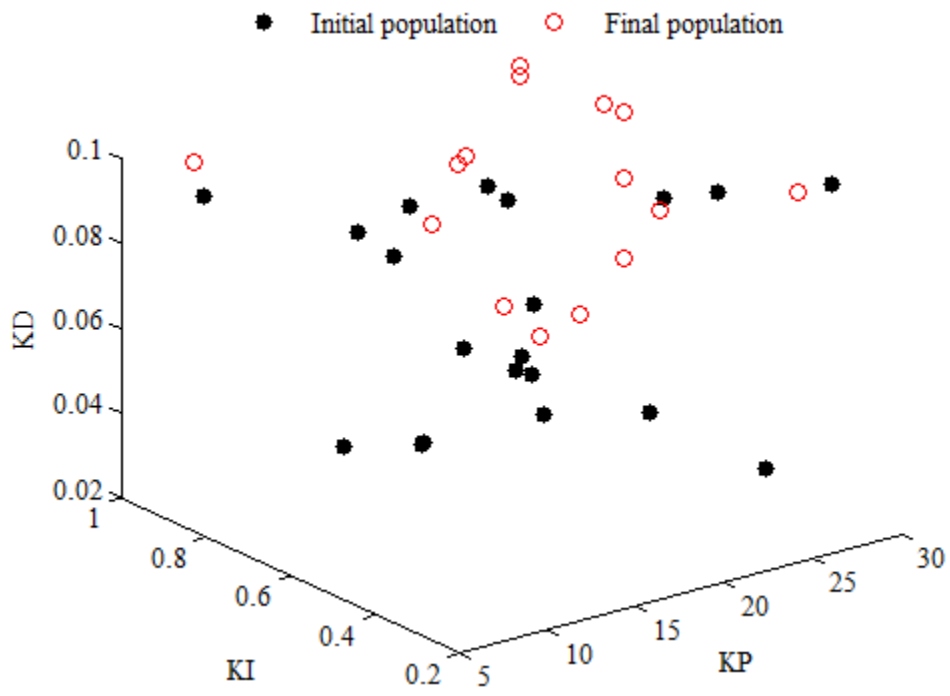


Figure 5.11(b) Initial and final population members of offline GA optimization based PID control for SR operations

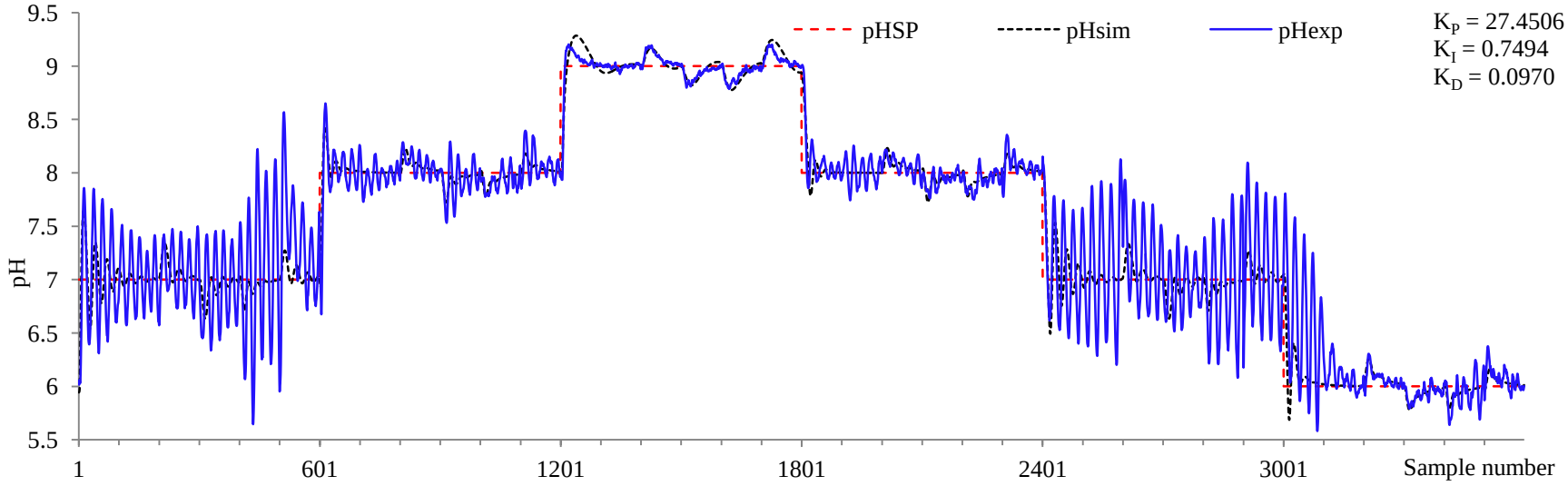


Figure 5.12(a) Simulated and experimental pH responses of offline GA optimization based PID control for SR operations

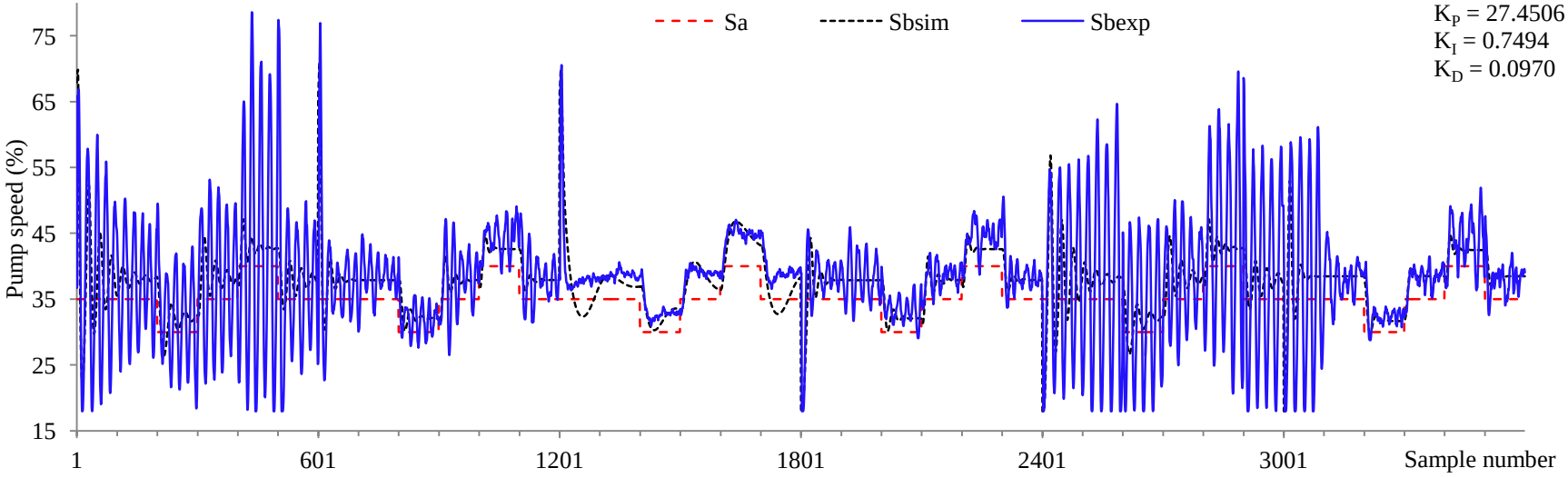


Figure 5.12(b) Simulated and experimental pumps speed variations of offline GA optimization based PID control for SR operations

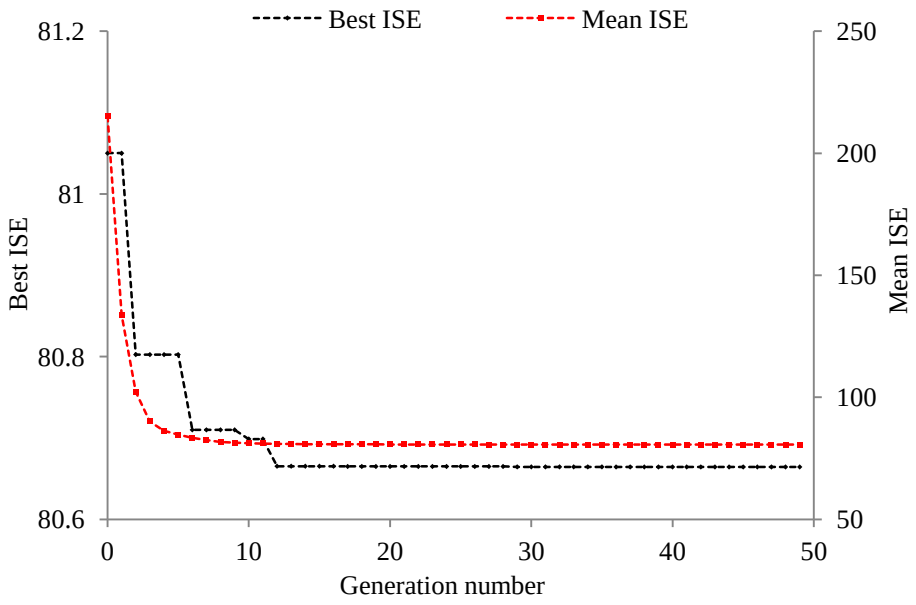


Figure 5.13(a) Best and mean ISE values of offline DE algorithm based PID control for SR operations

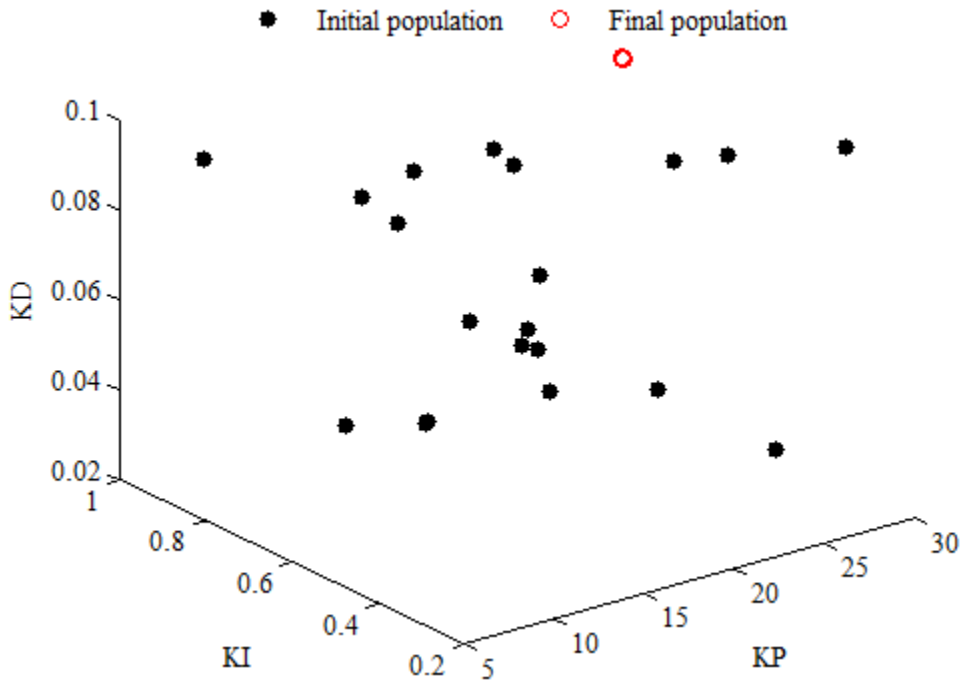


Figure 5.13(b) Initial and final population members of offline DE algorithm based PID control for SR operations

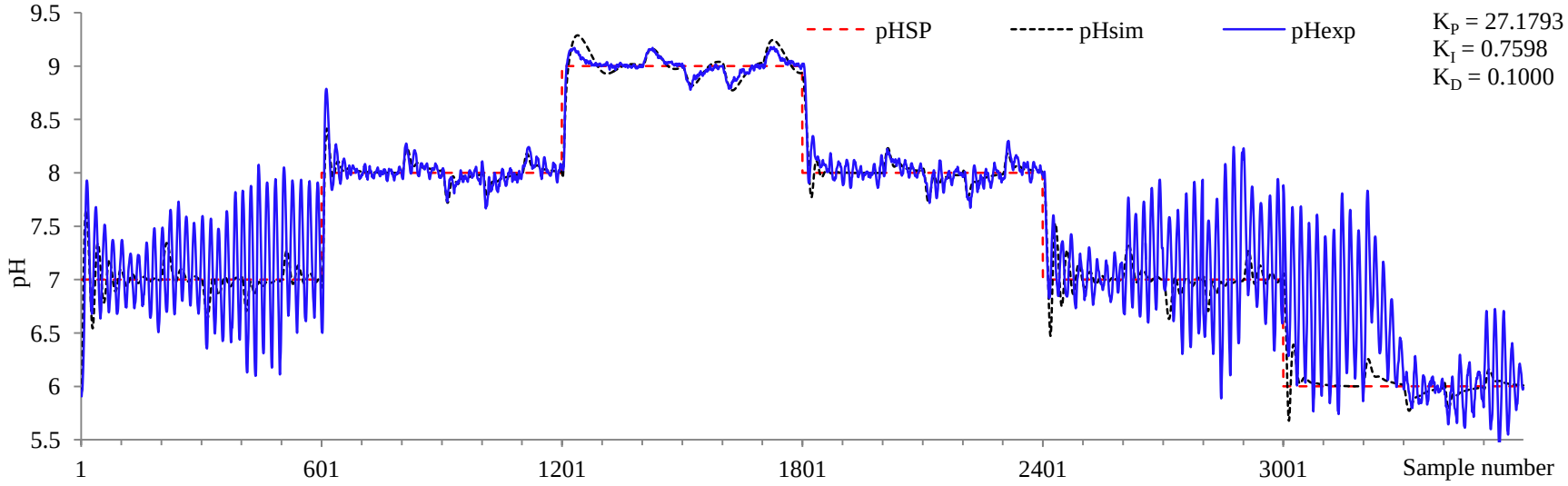


Figure 5.14(a) Simulated and experimental pH responses of offline DE algorithm based PID control for SR operations

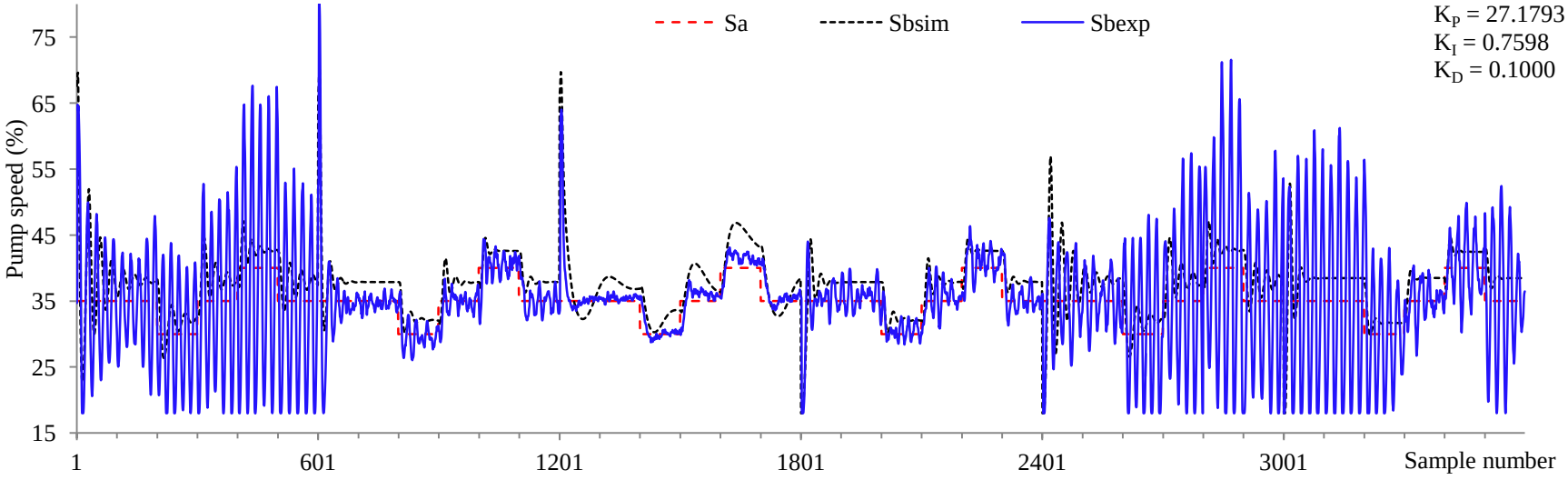


Figure 5.14(b) Simulated and experimental pumps speed variations of offline DE algorithm based PID control for SR operations

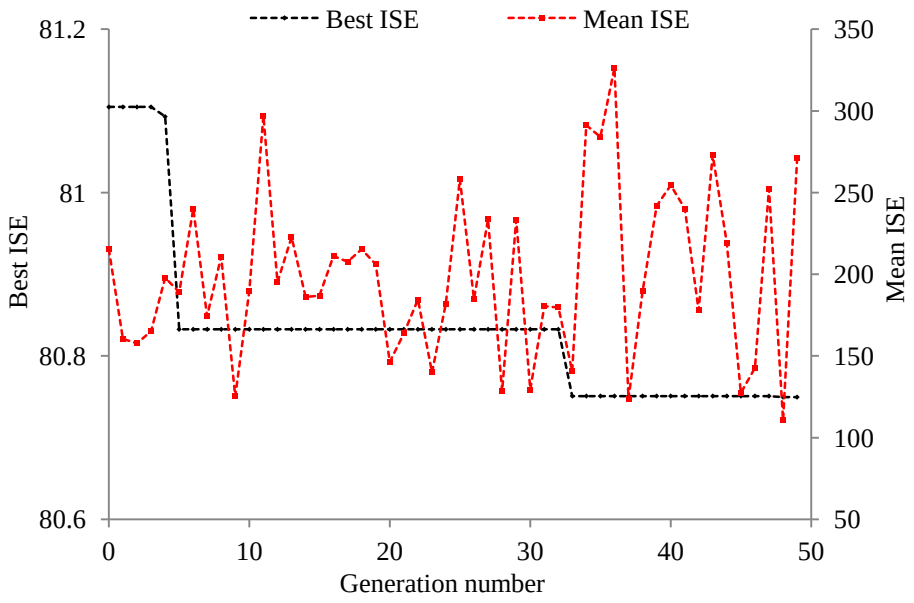


Figure 5.15(a) Best and mean ISE values of offline PSO algorithm based PID control for SR operations

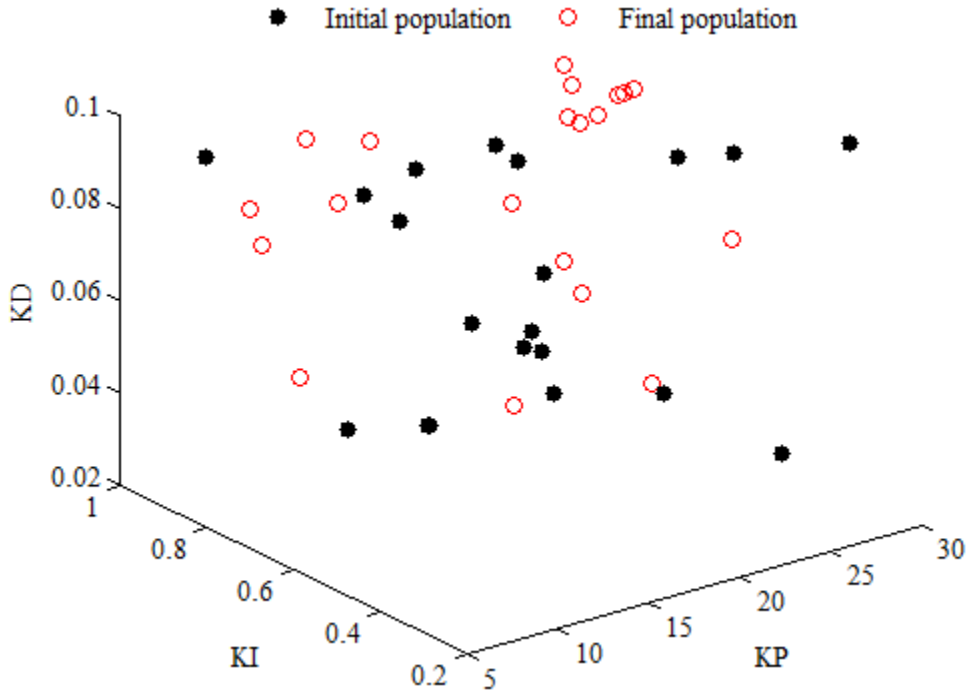


Figure 5.15(b) Initial and final particles positions of offline PSO algorithm based PID control for SR operations

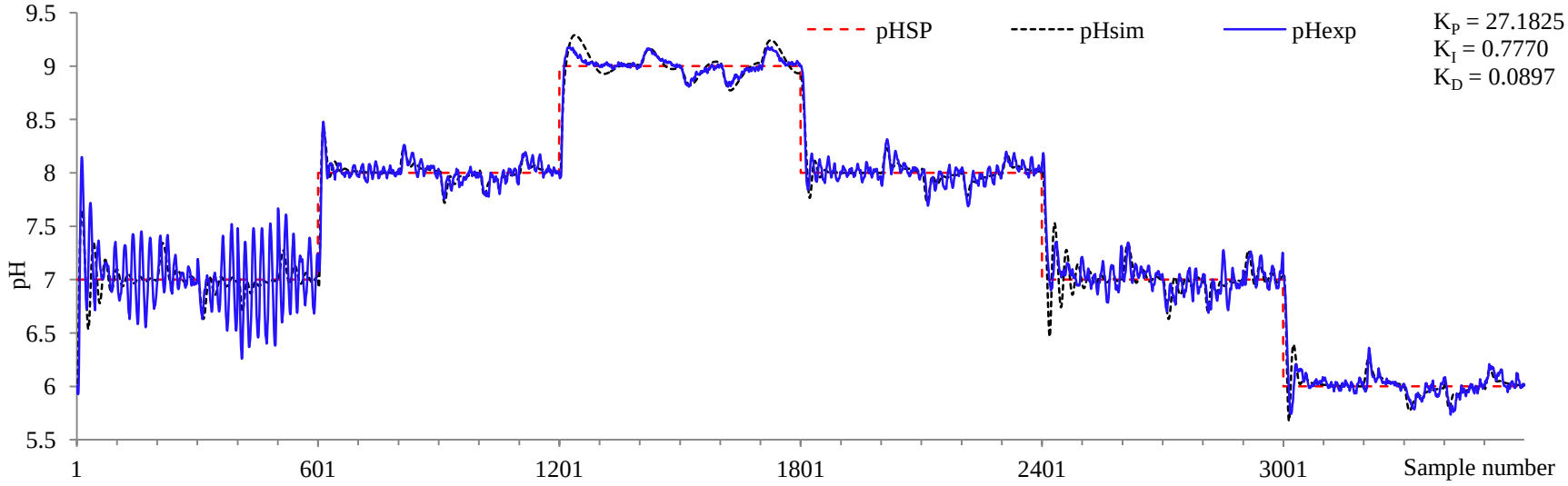


Figure 5.16(a) Simulated and experimental pH responses of offline PSO algorithm based PID control for SR operations

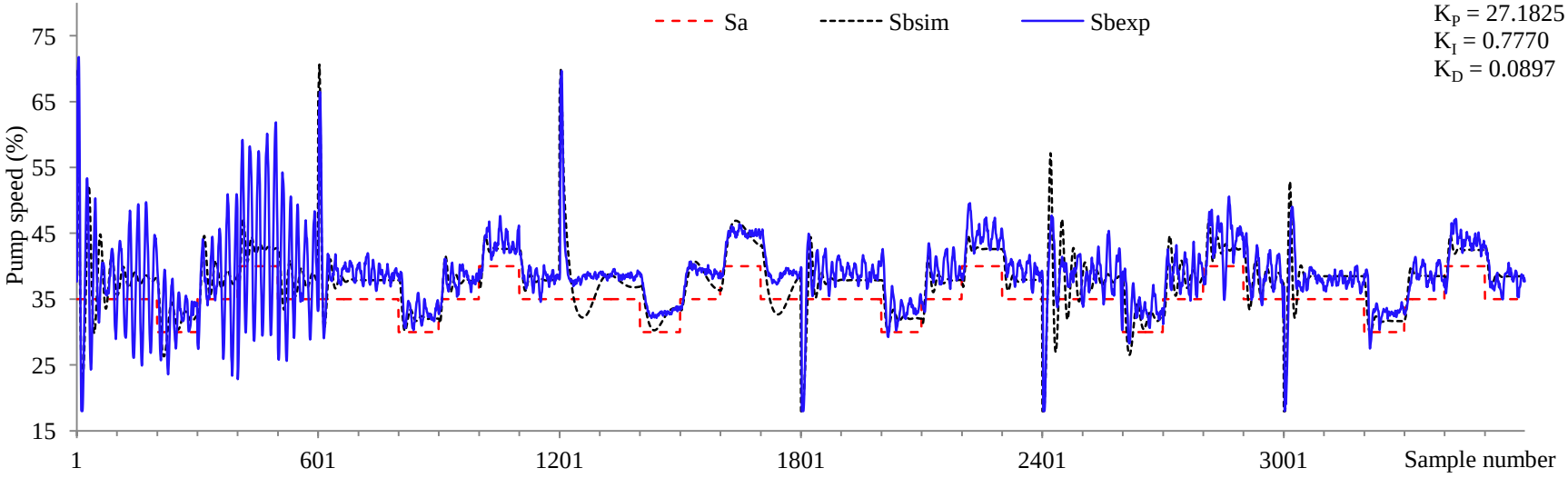


Figure 5.16(b) Simulated and experimental pumps speed variations of offline PSO algorithm based PID control for SR operations

Table 5.6(a) Simulation results of offline GA, DE, and PSO based PID control for SR operations

Optimization methods	Optimized parameters [K_P, K_I, K_D]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	For GA: [27.4506, 0.7494, 0.0970] For DE: [27.1793, 0.7598, 0.1000] For PSO: [27.1825, 0.7770, 0.0897]	6, 7, 35	10.9831, -0.6538	7, 35, 30	1.6804, -0.3430	7, 30, 35	1.6046, 0.3651	7, 35, 40	0.8638, 0.2928	7, 40, 35	0.9720, -0.2719
DE			10.8548, -0.6511		1.6847, -0.3433		1.6143, 0.3671		0.8620, 0.2938		0.9775, -0.2737
PSO			11.0167, -0.6553		1.6767, -0.3433		1.6067, 0.3667		0.8475, 0.2929		0.9804, -0.2741
GA		7, 8, 35	6.8505, -0.4715	8, 35, 30	0.8435, -0.2321	8, 30, 35	0.9142, 0.2763	8, 35, 40	0.7057, 0.2197	8, 40, 35	0.4902, -0.1808
DE			6.8574, -0.4155		0.8414, -0.2328		0.9229, 0.2781		0.6990, 0.2193		0.4944, -0.1819
PSO			6.8710, -0.4181		0.8277, -0.2326		0.9192, 0.2791		0.6821, 0.2186		0.4907, -0.1824
GA		8, 9, 35	9.6478, -0.2853	9, 35, 30	0.7679, -0.1656	9, 30, 35	0.9274, 0.1795	9, 35, 40	1.5237, 0.2236	9, 40, 35	2.0740, -0.2445
DE			9.7598, -0.2891		0.7653, -0.1655		0.9307, 0.1798		1.5440, 0.2258		2.0884, -0.2456
PSO			9.7918, -0.2916		0.7458, -0.1643		0.9100, 0.1784		1.5350, 0.2265		2.0460, -0.2441

Table 5.6(a) continued

Optimization methods	Optimized parameters [K_P, K_I, K_D]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	For GA: [27.4506, 0.7494, 0.0970] For DE: [27.1793, 0.7598, 0.1000] For PSO: [27.1825, 0.7770, 0.0897]	9, 8, 35	7.0911, 0.2147	8, 35, 30	0.8391, -0.2317	8, 30, 35	0.9147, 0.2764	8, 35, 40	0.7057, 0.2197	8, 40, 35	0.4902, -0.1808
DE			7.0688, 0.2285		0.8370, -0.2323		0.9233, 0.2782		0.6990, 0.2194		0.4944, -0.1819
PSO			7.0908, 0.2333		0.8237, -0.2322		0.9195, 0.2791		0.6822, 0.2186		0.4907, -0.1824
GA		8, 7, 35	12.8125, 0.5075	7, 35, 30	1.6146, -0.3334	7, 30, 35	1.6069, 0.3656	7, 35, 40	0.8637, 0.2928	7, 40, 35	0.9720, -0.2719
DE			12.8242, 0.5267		1.6424, -0.3370		1.6156, 0.3674		0.8619, 0.2937		0.9775, -0.2737
PSO			12.9480, 0.5352		1.6325, -0.3368		1.6080, 0.3670		0.8474, 0.2928		0.9804, -0.2741
GA		7, 6, 35	8.8189, 0.3144	6, 35, 30	1.1377, -0.2552	6, 30, 35	0.9658, 0.2257	6, 35, 40	0.5789, 0.2118	6, 40, 35	0.3834, -0.1598
DE			8.7556, 0.3212		1.1347, -0.2561		0.9739, 0.2274		0.5727, 0.2114		0.3870, -0.1607
PSO			8.7563, 0.3240		1.1141, -0.2558		0.9653, 0.2281		0.5586, 0.2107		0.3845, -0.1613

Table 5.6(b) Experimental performance of offline GA, DE, and PSO based PID control for SR operations

Optimization methods	Optimized parameters [K_P, K_I, K_D]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	For GA: [27.4506, 0.7494, 0.0970] For DE: [27.1793, 0.7598, 0.1000] For PSO: [27.1825, 0.7770, 0.0897]	6, 7, 35	34.5965 -0.8590	7, 35, 30	7.2478, -0.5000	7, 30, 35	12.5785, 0.6640	7, 35, 40	49.0433, 1.3550	7, 40, 35	36.5412, -1.5690
DE			25.6436, -0.9290		12.2941, -0.7310		18.0476, 0.6450		40.2100, 0.9010		31.3632, -1.0510
PSO			26.5633, -1.1470		2.3660, -0.4170		6.0225, 0.4790		15.7488, 0.7410		7.9042, -0.6670
GA		7, 8, 35	13.1788, -0.6520	8, 35, 30	1.5084, -0.2880	8, 30, 35	3.5932, 0.4680	8, 35, 40	1.4699, 0.2240	8, 40, 35	2.3774, -0.3960
DE			15.8410, -0.7870		1.0915, -0.2750		0.8090, 0.2690		1.3507, 0.3330		1.0129, -0.2430
PSO			6.4668, -0.4790		1.0625, -0.2620		0.7486, 0.2440		0.9975, 0.2240		0.8049, -0.1920
GA		8, 9, 35	7.8519, -0.2030	9, 35, 30	0.8998, -0.1960	9, 30, 35	0.8233, 0.2010	9, 35, 40	0.9883, 0.2130	9, 40, 35	0.8641, -0.2030
DE			6.9078, -0.1710		0.7030, -0.1710		0.7739, 0.2200		0.8942, 0.2130		0.7705, -0.1770
PSO			8.1147, -0.1770		0.7068, -0.1640		0.7833, 0.1940		0.8752, 0.1940		0.8384, -0.1830

Table 5.6(b) continued

Optimization methods	Optimized parameters [K_P, K_I, K_D]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	For GA: [27.4506, 0.7494, 0.0970] For DE: [27.1793, 0.7598, 0.1000] For PSO: [27.1825, 0.7770, 0.0897]	9, 8, 35	11.0012, 0.2560	8, 35, 30	1.2249, -0.2170	8, 30, 35	0.9376, 0.2370	8, 35, 40	1.3258, 0.2500	8, 40, 35	1.6496, -0.3580
DE			10.9259, 0.1410		0.8238, -0.2240		1.1071, 0.2820		1.3689, 0.3270		1.1772, -0.3000
PSO			8.9635, 0.1600		1.3337, -0.3130		1.1831, 0.3080		1.2455, 0.3140		0.7703, -0.1980
GA		8, 7, 35	57.9020, 0.7990	7, 35, 30	19.1405, -0.9360	7, 30, 35	6.5021, 0.4850	7, 35, 40	25.1742, 0.9200	7, 40, 35	28.2774, -1.0960
DE			14.3845, 0.2680		20.5689, -0.9360		21.5390, 0.6960		41.1555, 1.1120		28.6422, -1.2300
PSO			13.8896, 0.2160		1.2914, -0.3470		1.4758, 0.3000		2.0812, 0.3120		1.4353, -0.2640
GA		7, 6, 35	78.8163, 0.4190	6, 35, 30	1.1319, -0.3040	6, 30, 35	0.9230, 0.2080	6, 35, 40	2.6092, 0.3620	6, 40, 35	1.5383, -0.3740
DE			194.5038, 0.2590		65.9590, -1.8330		1.0529, 0.2020		3.4230, 0.3870		12.6114, -0.7260
PSO			8.4939, 0.2590		1.0429, -0.3610		0.9175, 0.2150		1.0534, 0.2660		0.8373, -0.2080

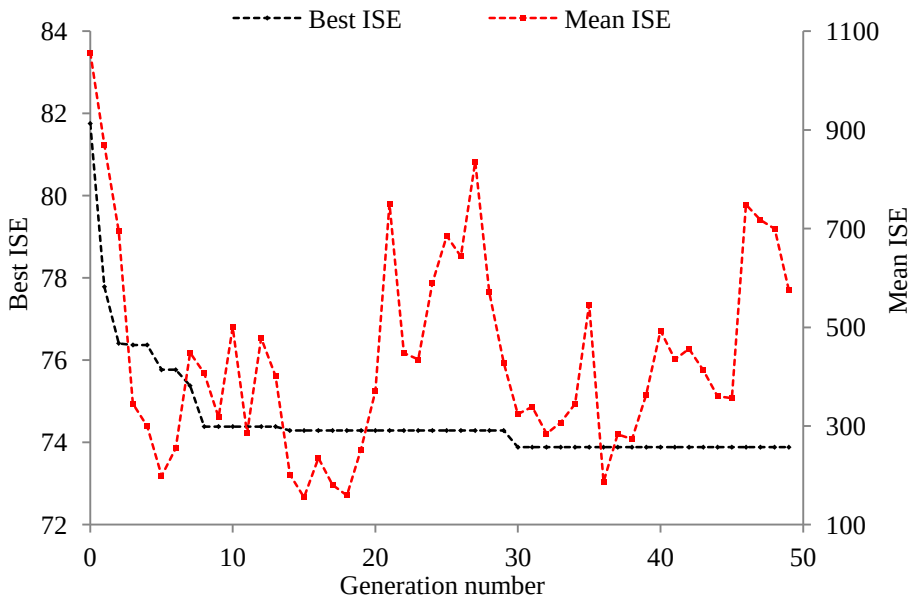


Figure 5.17(a) Best and mean ISE values of offline GA optimization based FLC for SR operations

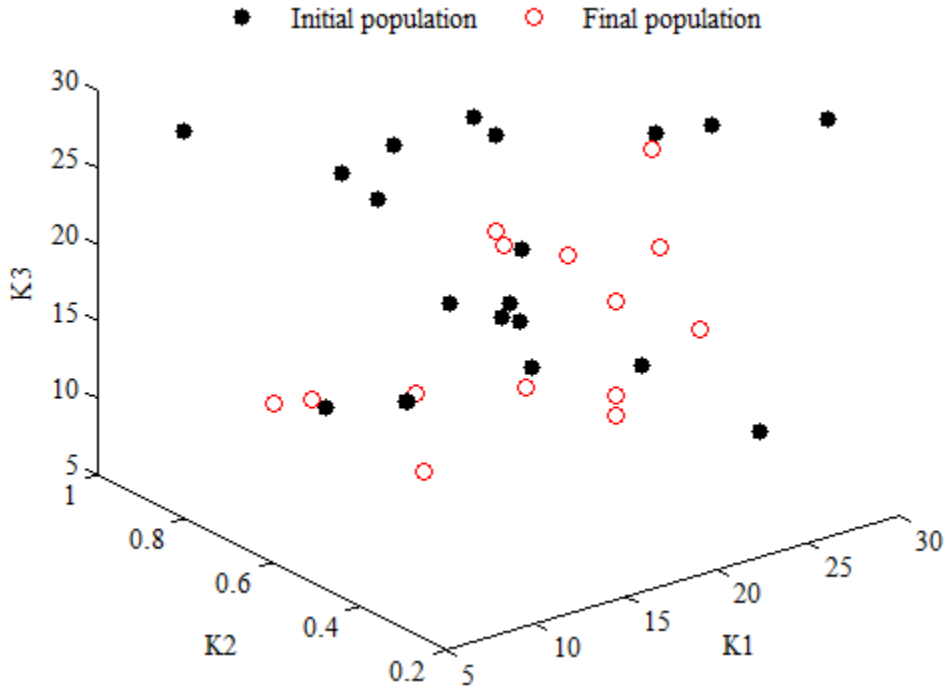


Figure 5.17(b) Initial and final population members of offline GA optimization based FLC for SR operations

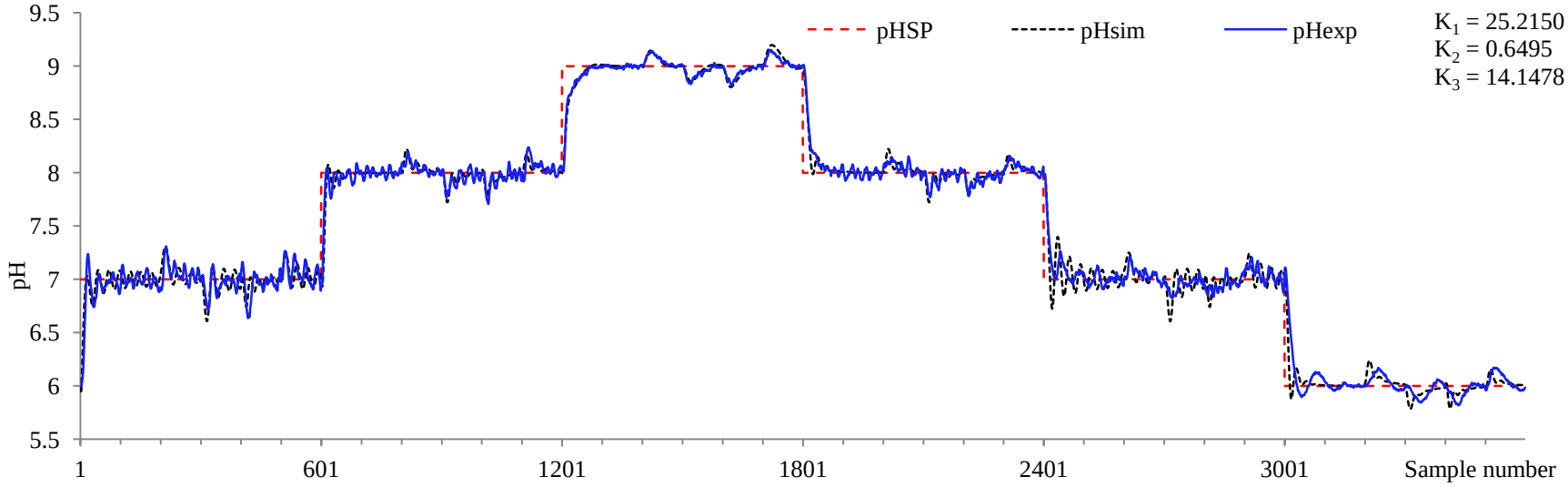


Figure 5.18(a) Simulated and experimental pH responses of offline GA optimization based FLC for SR operations

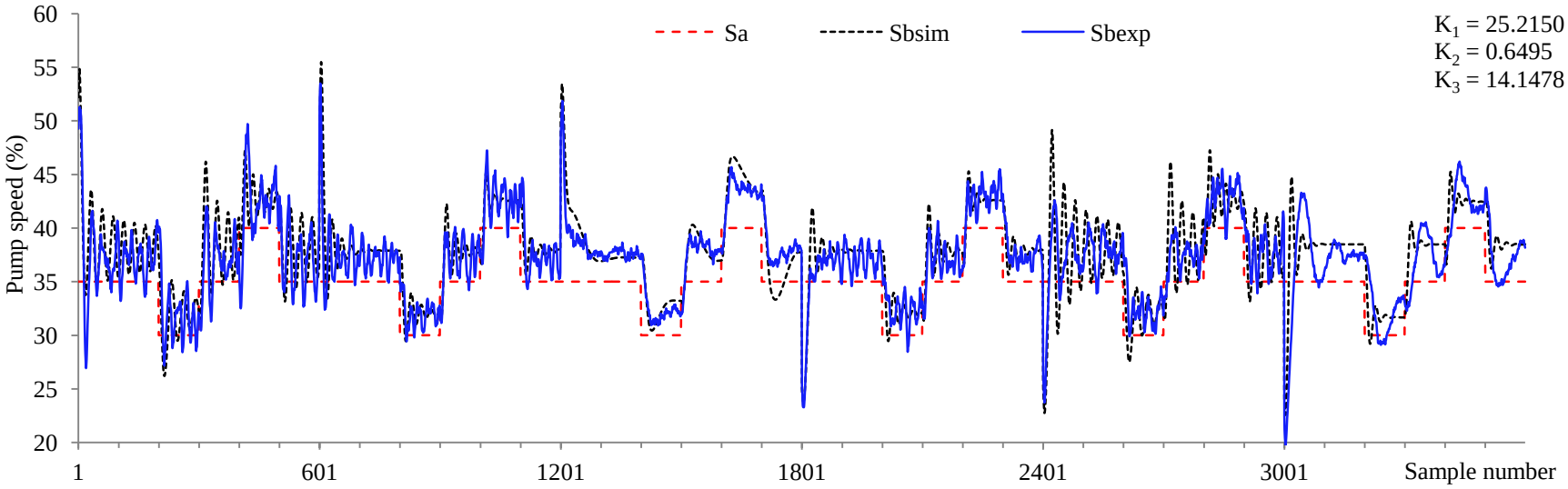


Figure 5.18(b) Simulated and experimental pumps speed variations of offline GA optimization based FLC for SR operations

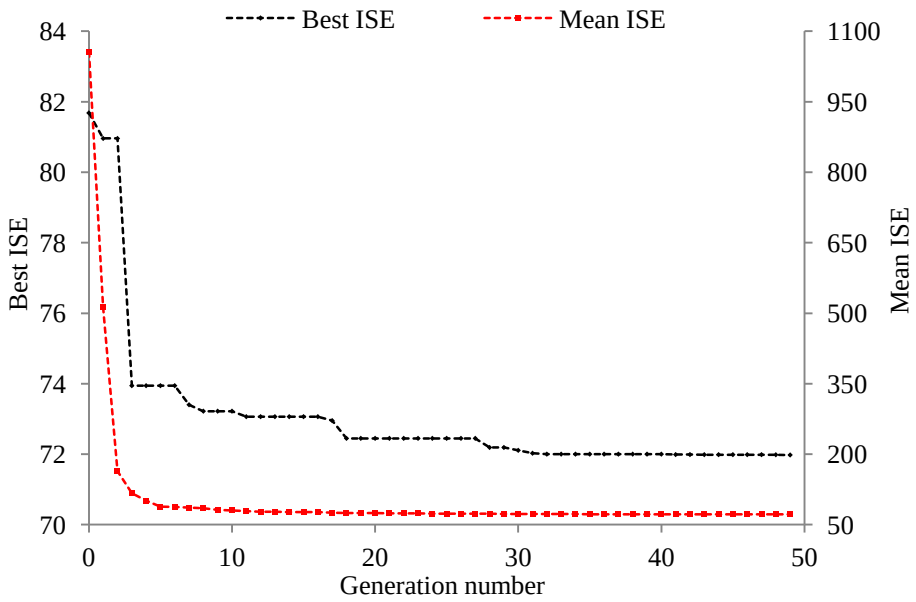


Figure 5.19(a) Best and mean ISE values of offline DE algorithm based FLC for SR operations

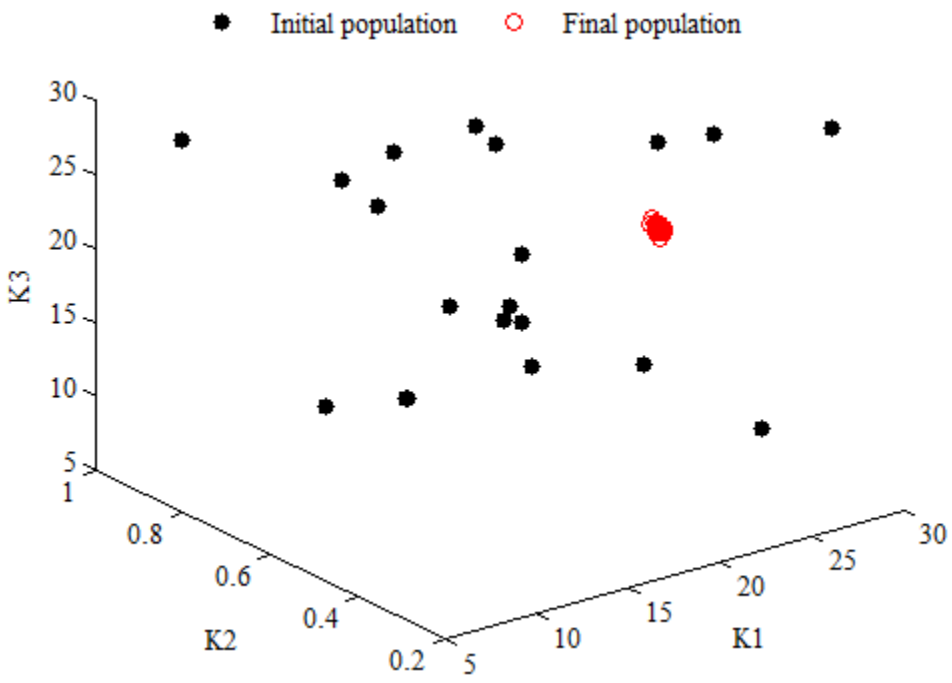


Figure 5.19(b) Initial and final population members of offline DE algorithm based FLC for SR operations

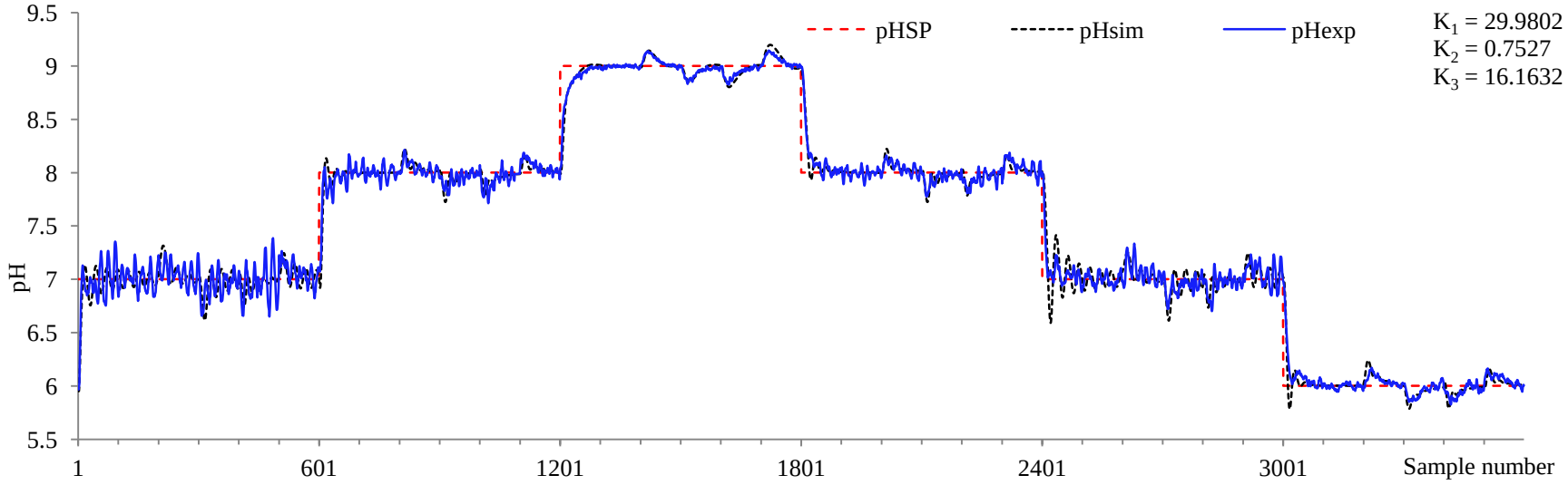


Figure 5.20(a) Simulated and experimental pH responses of offline DE algorithm based FLC for SR operations

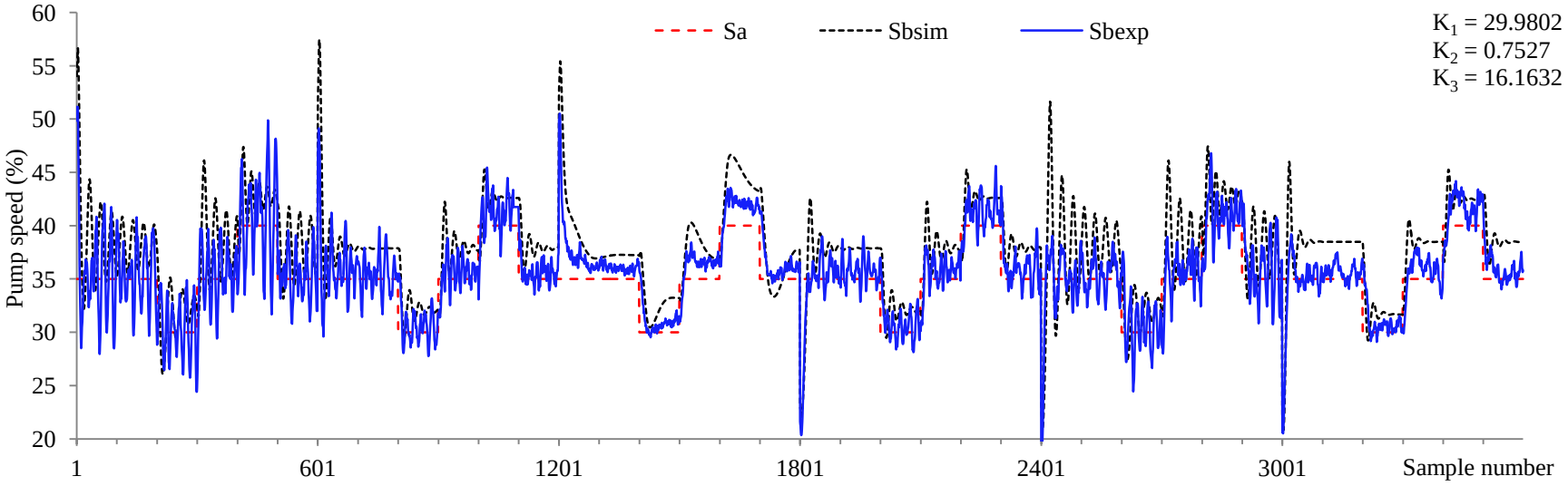


Figure 5.20(b) Simulated and experimental pumps speed variations of offline DE algorithm based FLC for SR operations

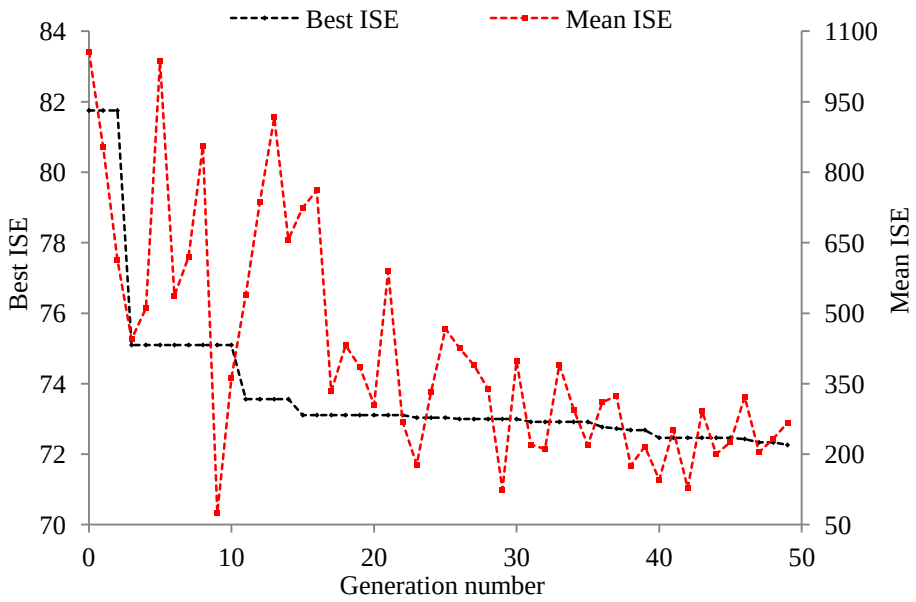


Figure 5.21(a) Best and mean ISE values of offline PSO algorithm based FLC for SR operations

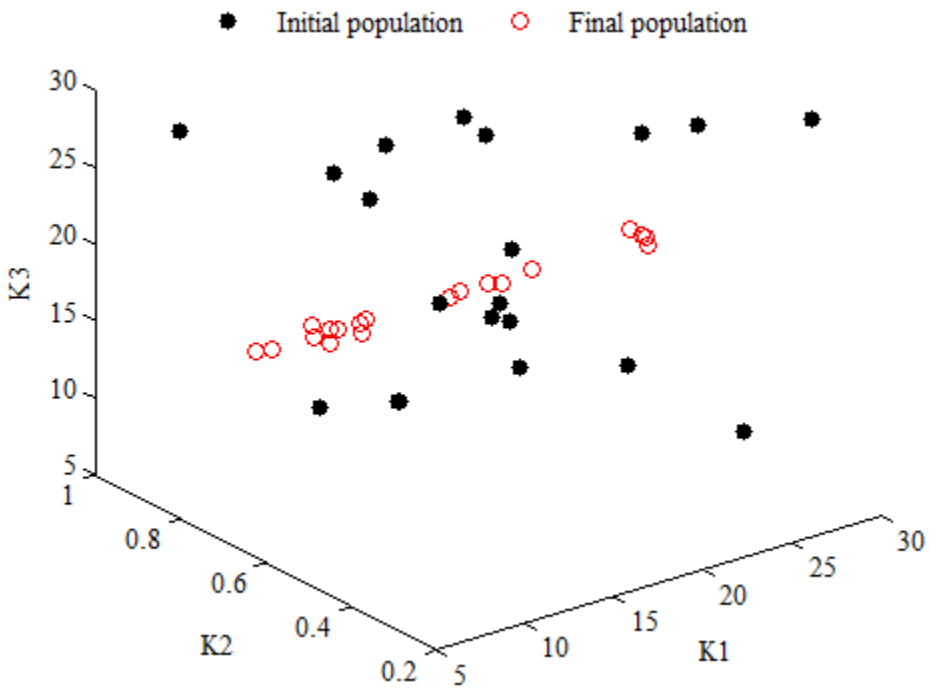


Figure 5.21(b) Initial and final particles positions of offline PSO algorithm based FLC for SR operations

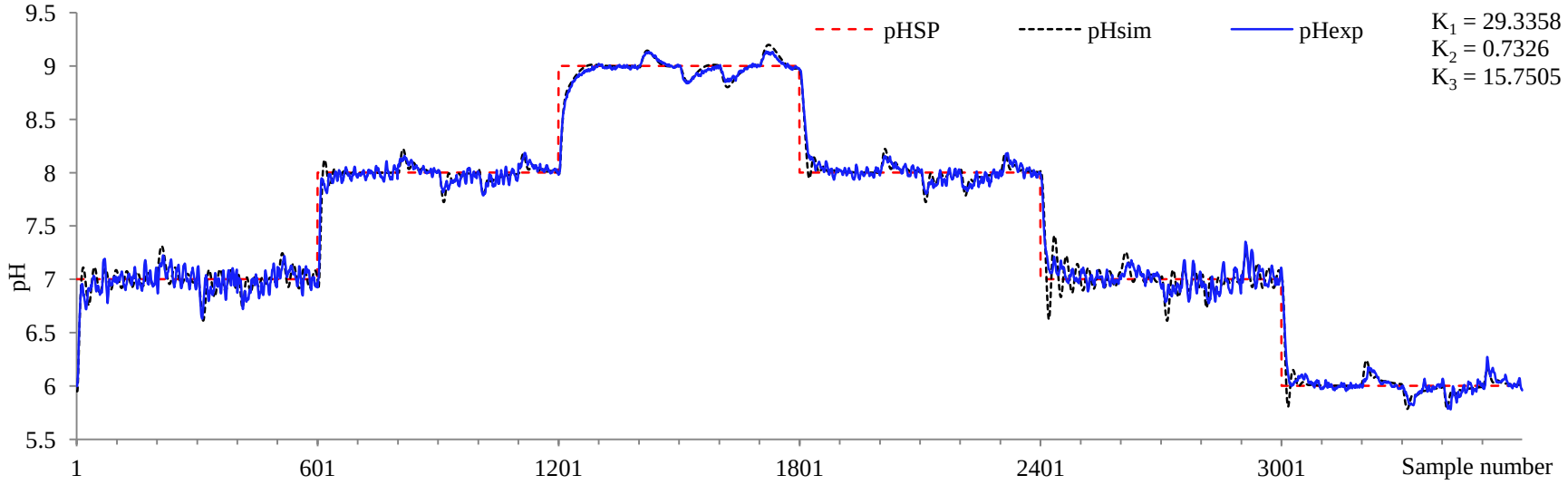


Figure 5.22(a) Simulated and experimental pH responses of offline PSO algorithm based FLC for SR operations

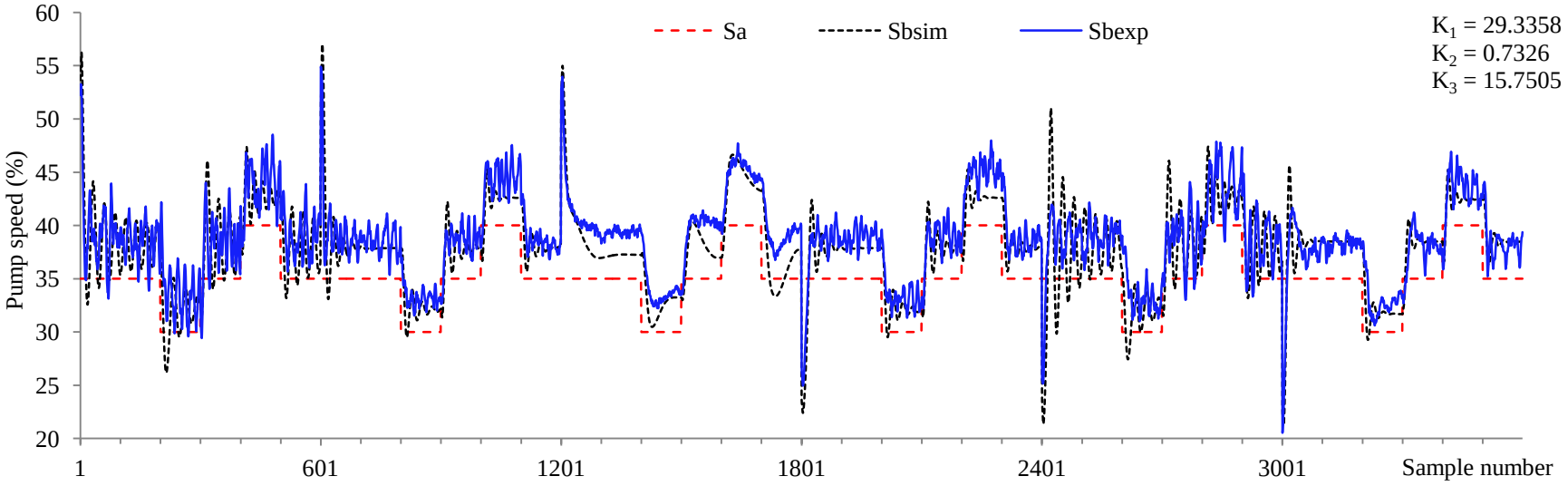


Figure 5.22(b) Simulated and experimental pumps speed variations of offline PSO algorithm based FLC for SR operations

Table 5.7(a) Simulation results of offline GA, DE, and PSO based FLC for SR operations

Optimization methods	Optimized parameters [K_1, K_2, K_3]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	For GA: [25.2150, 0.6495, 14.1478]	6, 7, 35	7.6934, -0.0893	7, 35, 30	1.3546, -0.3081	7, 30, 35	1.9131, 0.3910	7, 35, 40	0.6017, 0.2557	7, 40, 35	0.9970, -0.2442
DE			7.3849, -0.1346		1.4062, -0.3151		1.8718, 0.3868		0.6441, 0.2648		0.9878, -0.2452
PSO			7.3919, -0.1143		1.3857, -0.3126		1.8642, 0.3871		0.6467, 0.2652		0.9825, -0.2454
GA	For DE: [29.9802, 0.7527, 16.1632]	7, 8, 35	8.2813, -0.0770	8, 35, 30	0.6583, -0.2225	8, 30, 35	0.8060, 0.2768	8, 35, 40	0.5483, 0.2143	8, 40, 35	0.3767, -0.1742
DE			7.8684, -0.1350		0.6687, -0.2226		0.8036, 0.2753		0.5581, 0.2142		0.3787, -0.1737
PSO			7.9237, -0.1212		0.6725, -0.2228		0.8070, 0.2755		0.5629, 0.2144		0.3801, -0.1737
GA	For PSO: [29.3358, 0.7326, 15.7505]	8, 9, 35	10.2147, -0.0148	9, 35, 30	0.4654, -0.1443	9, 30, 35	0.5005, 0.1463	9, 35, 40	0.9825, 0.1964	9, 40, 35	1.1149, -0.1973
DE			9.2823, -0.0141		0.4805, -0.1452		0.5180, 0.1475		1.0106, 0.1971		1.1599, -0.1993
PSO			9.4820, -0.0138		0.4833, -0.1454		0.5208, 0.1477		1.0156, 0.1972		1.1674, -0.1997

Table 5.7(a) continued

Optimization methods	Optimized parameters [K_1, K_2, K_3]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	For GA: [25.2150, 0.6495, 14.1478]	9, 8, 35	9.4928, 0.0132	8, 35, 30	0.6614, -0.2228	8, 30, 35	0.8058, 0.2768	8, 35, 40	0.5482, 0.2143	8, 40, 35	0.3767, -0.1742
DE			8.8785, 0.0681		0.6707, -0.2228		0.8034, 0.2753		0.5581, 0.2142		0.3787, -0.1737
PSO			8.9915, 0.0530		0.6752, -0.2230		0.8068, 0.2754		0.5628, 0.2144		0.3801, -0.1737
GA	For DE: [29.9802, 0.7527, 16.1632]	8, 7, 35	11.4526, 0.2750	7, 35, 30	0.9286, -0.2506	7, 30, 35	1.9289, 0.3929	7, 35, 40	0.6125, 0.2582	7, 40, 35	0.9983, -0.2444
DE			11.9181, 0.4094		0.9507, -0.2540		1.8821, 0.3883		0.6523, 0.2668		0.9887, -0.2453
PSO			11.7326, 0.3755		0.9577, -0.2553		1.8749, 0.3883		0.6531, 0.2668		0.9833, -0.2455
GA	For PSO: [29.3358, 0.7326, 15.7505]	7, 6, 35	7.1640, 0.1216	6, 35, 30	0.8692, -0.2403	6, 30, 35	0.7339, 0.2161	6, 35, 40	0.4947, 0.2120	6, 40, 35	0.3084, -0.1581
DE			6.8365, 0.2192		0.8837, -0.2403		0.7413, 0.2152		0.5016, 0.2119		0.3098, -0.1573
PSO			6.9031, 0.1951		0.8907, -0.2406		0.7453, 0.2152		0.5066, 0.2123		0.3109, -0.1573

Table 5.7(b) Experimental performance of offline GA, DE, and PSO based FLC for SR operations

Optimization methods	Optimized parameters [K_1, K_2, K_3]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	For GA: [25.2150, 0.6495, 14.1478]	6, 7, 35	10.3488, -0.2380	7, 35, 30	1.2209, -0.3080	7, 30, 35	1.1533, 0.2870	7, 35, 40	1.6590, 0.3640	7, 40, 35	1.4067, -0.2640
DE			8.5643, -0.3530		1.4144, -0.2510		1.8447, 0.3440		3.1169, 0.3510		1.3759, -0.2640
PSO			7.0885, -0.1930		0.9584, -0.2250		1.5885, 0.3700		1.2468, 0.2800		0.7806, -0.2190
GA	For DE: [29.9802, 0.7527, 16.1632]	7, 8, 35	6.8719, -0.0890	8, 35, 30	0.4960, -0.1980	8, 30, 35	0.8394, 0.2370	8, 35, 40	0.9794, 0.2950	8, 40, 35	0.8325, -0.2360
DE			6.4412, -0.1720		0.5449, -0.2110		0.7182, 0.2120		1.1182, 0.2880		0.6645, -0.1850
PSO			6.1936, -0.1080		0.5310, -0.1600		0.7156, 0.1860		0.8152, 0.2120		0.5549, -0.1850
GA	For PSO: [29.3358, 0.7326, 15.7505]	8, 9, 35	9.8882, -0.0230	9, 35, 30	0.4565, -0.1390	9, 30, 35	0.6161, 0.1690	9, 35, 40	0.6406, 0.1810	9, 40, 35	0.4766, -0.1510
DE			7.2982, -0.0170		0.4240, -0.1390		0.5418, 0.1690		0.6034, 0.1750		0.4751, -0.1450
PSO			9.7712, -0.0230		0.4794, -0.1320		0.7197, 0.1620		0.6000, 0.1490		0.5111, -0.1390

Table 5.7(b) continued

Optimization methods	Optimized parameters [K_1, K_2, K_3]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	For GA: [25.2150, 0.6495, 14.1478]	9, 8, 35	11.4676, 0.0770	8, 35, 30	0.5008, -0.1530	8, 30, 35	0.7142, 0.2240	8, 35, 40	0.7668, 0.2180	8, 40, 35	0.4544, -0.1470
DE			9.3785, 0.1220		0.5287, -0.1600		0.6714, 0.2240		0.7811, 0.1920		0.7696, -0.1850
PSO			10.2793, 0.0640		0.5107, -0.1530		0.8115, 0.1990		0.8035, 0.1730		0.5544, -0.1850
GA	For DE: [29.9802, 0.7527, 16.1632]	8, 7, 35	9.6457, 0.0950	7, 35, 30	0.5869, -0.2120	7, 30, 35	0.6304, 0.1780	7, 35, 40	0.6450, 0.1650	7, 40, 35	0.7645, -0.2060
DE			6.7159, 0.1200		1.4692, -0.3340		1.0709, 0.2800		1.0196, 0.3000		1.4401, -0.2320
PSO			7.7541, 0.1140		0.6251, -0.1800		1.3011, 0.2160		1.1693, 0.2230		1.5406, -0.3530
GA	For PSO: [29.3358, 0.7326, 15.7505]	7, 6, 35	13.4889, 0.1060	6, 35, 30	0.6737, -0.1690	6, 30, 35	0.6595, 0.1570	6, 35, 40	0.7081, 0.1830	6, 40, 35	0.7855, -0.1690
DE			6.7687, 0.0550		0.4665, -0.1690		0.5959, 0.1510		0.5880, 0.1760		0.5541, -0.1630
PSO			7.2175, 0.0480		0.5689, -0.1760		0.6707, 0.1830		0.8295, 0.2210		0.7359, -0.2720

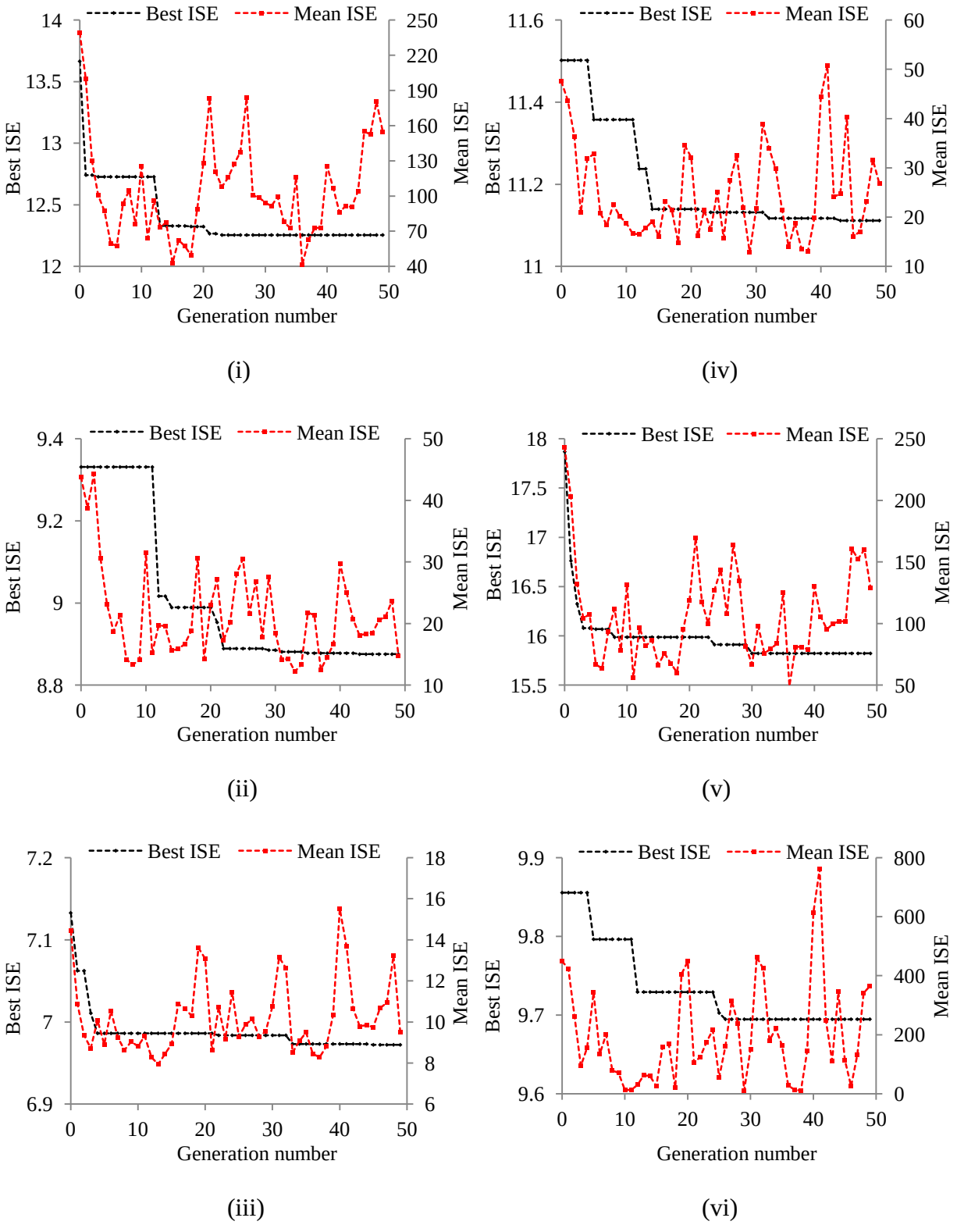


Figure 5.23(a) Best and mean ISE values of offline GA optimization based piecewise FLC for SR operations (i) SR1 (ii) SR2 (iii) SR3 (iv) SR4 (v) SR5 (vi) SR6

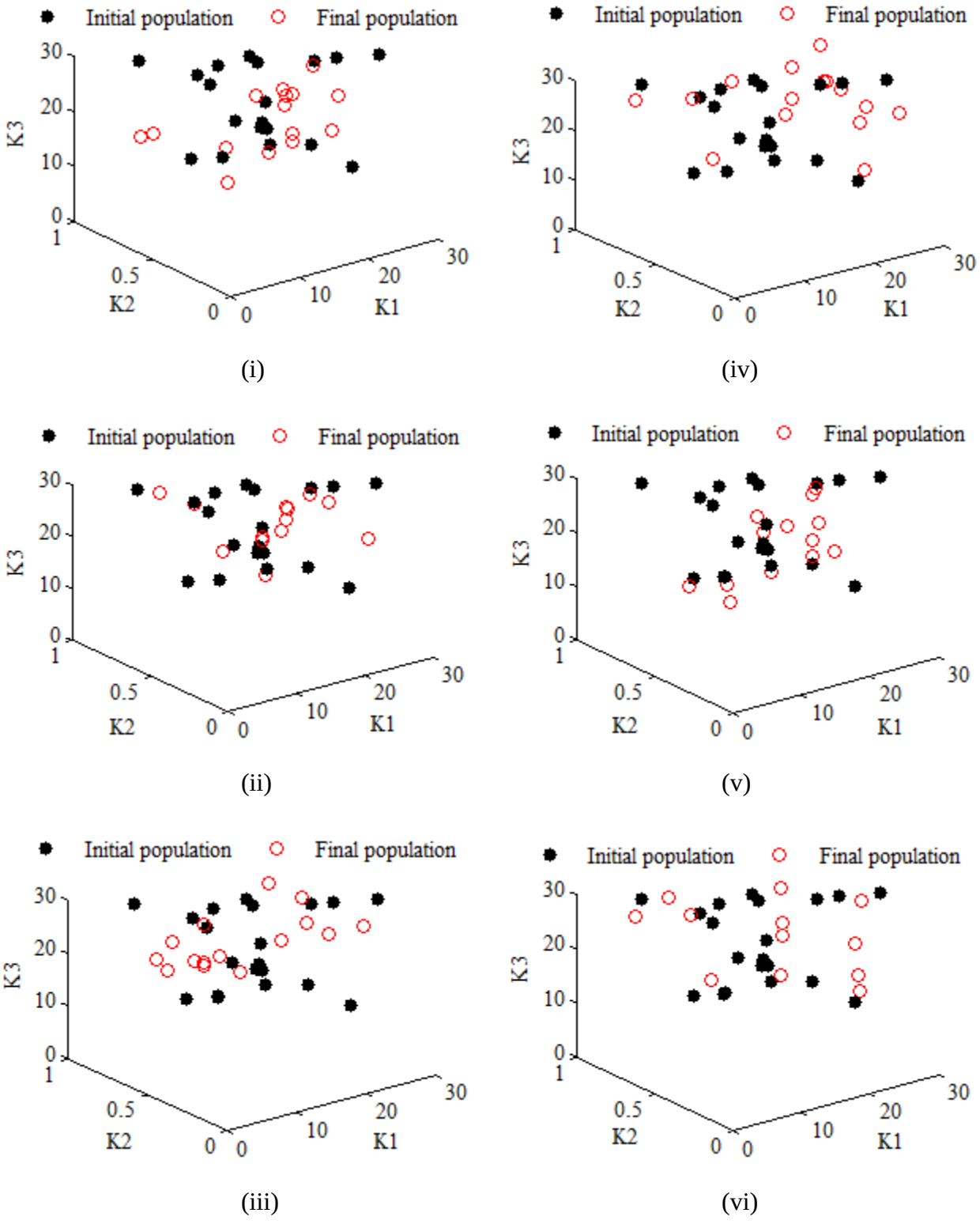


Figure 5.23(b) Initial and final population members of offline GA optimization based piecewise FLC for SR operations (i) SR1 (ii) SR2 (iii) SR3 (iv) SR4 (v) SR5 (vi) SR6

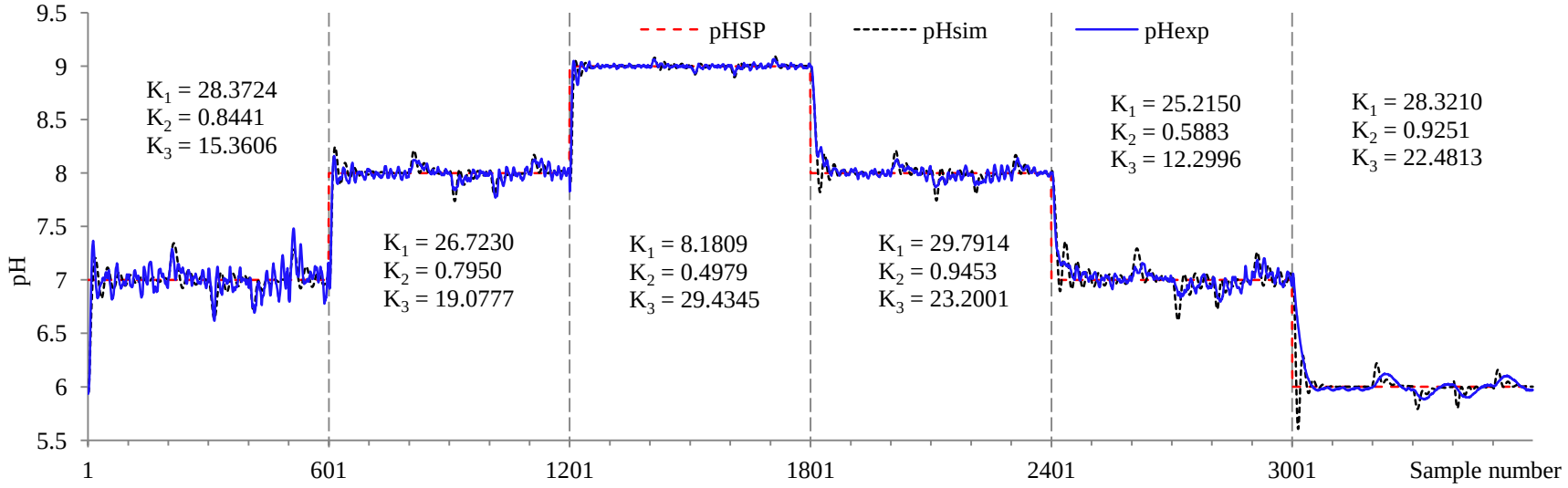


Figure 5.24(a) Simulated and experimental pH responses of offline GA optimization based piecewise FLC for SR operations

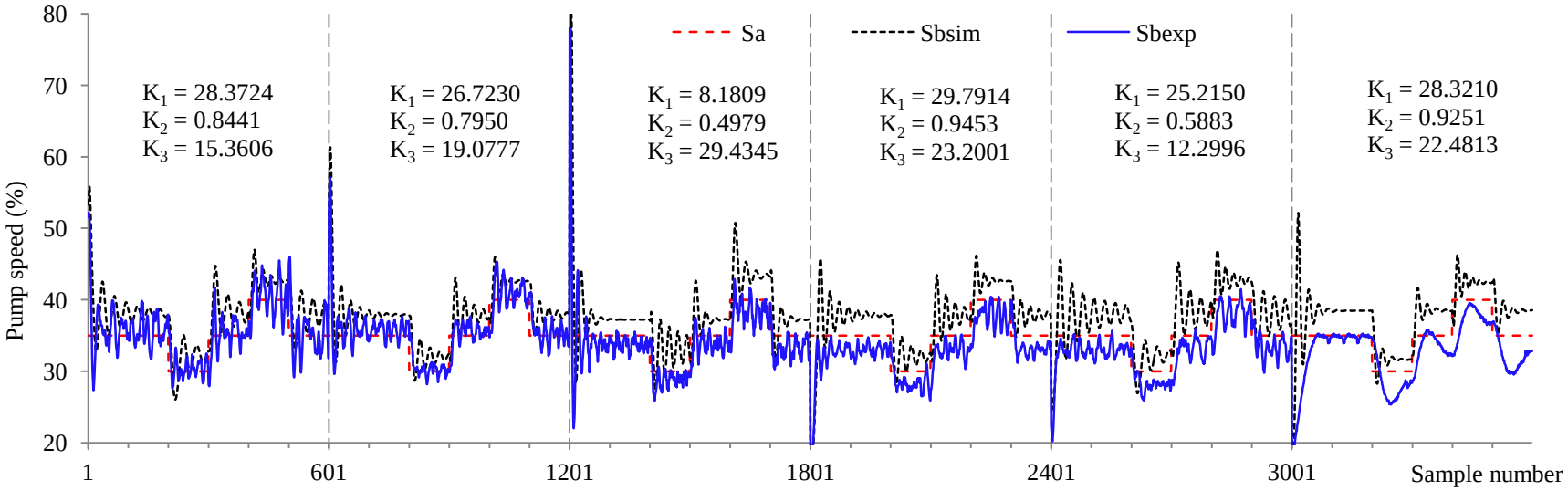


Figure 5.24(b) Simulated and experimental pumps speed variations of offline GA optimization based piecewise FLC for SR operations

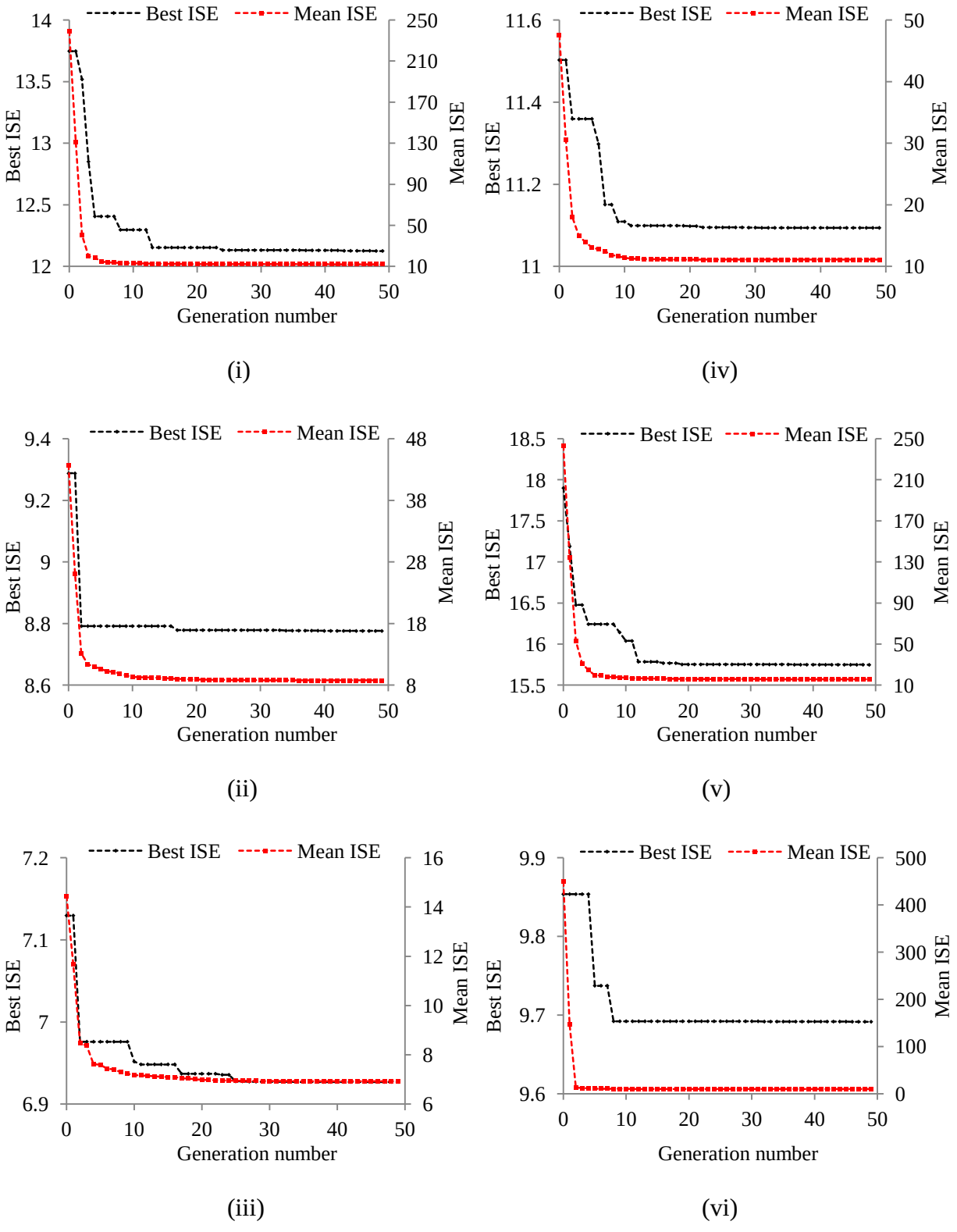


Figure 5.25(a) Best and mean ISE values of offline DE algorithm based piecewise FLC for SR operations (i) SR1 (ii) SR2 (iii) SR3 (iv) SR4 (v) SR5 (vi) SR6

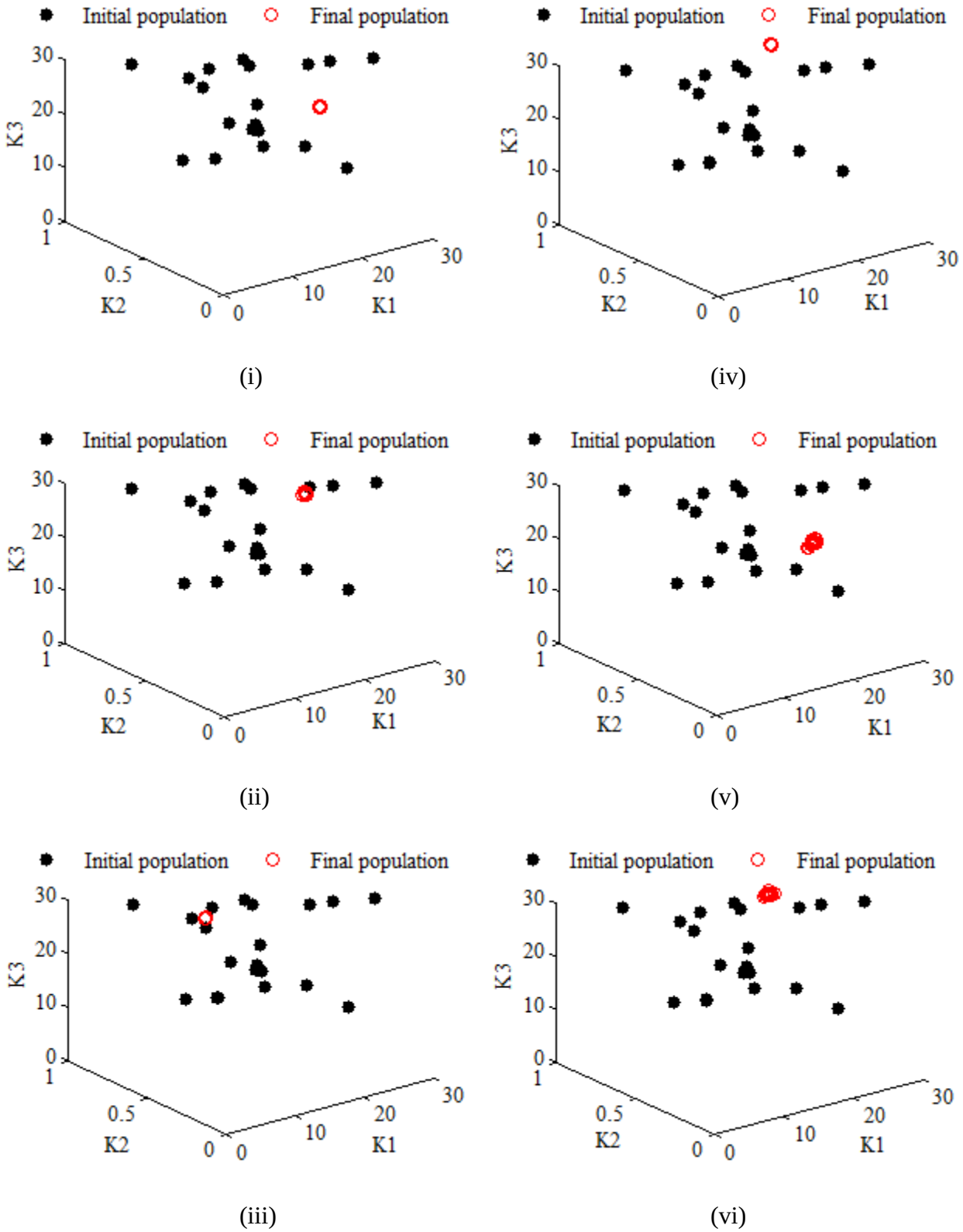


Figure 5.25(b) Initial and final population members of offline GA optimization based piecewise FLC for SR operations (i) SR1 (ii) SR2 (iii) SR3 (iv) SR4 (v) SR5 (vi) SR6

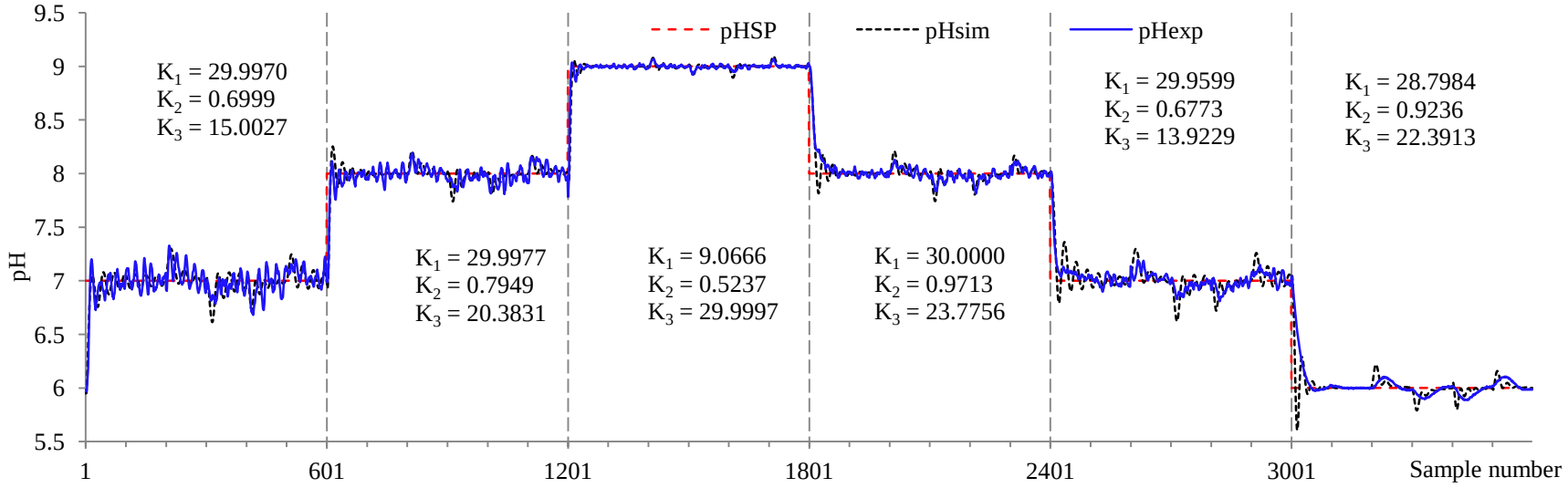


Figure 5.26(a) Simulated and experimental pH responses of offline DE algorithm based piecewise FLC for SR operations

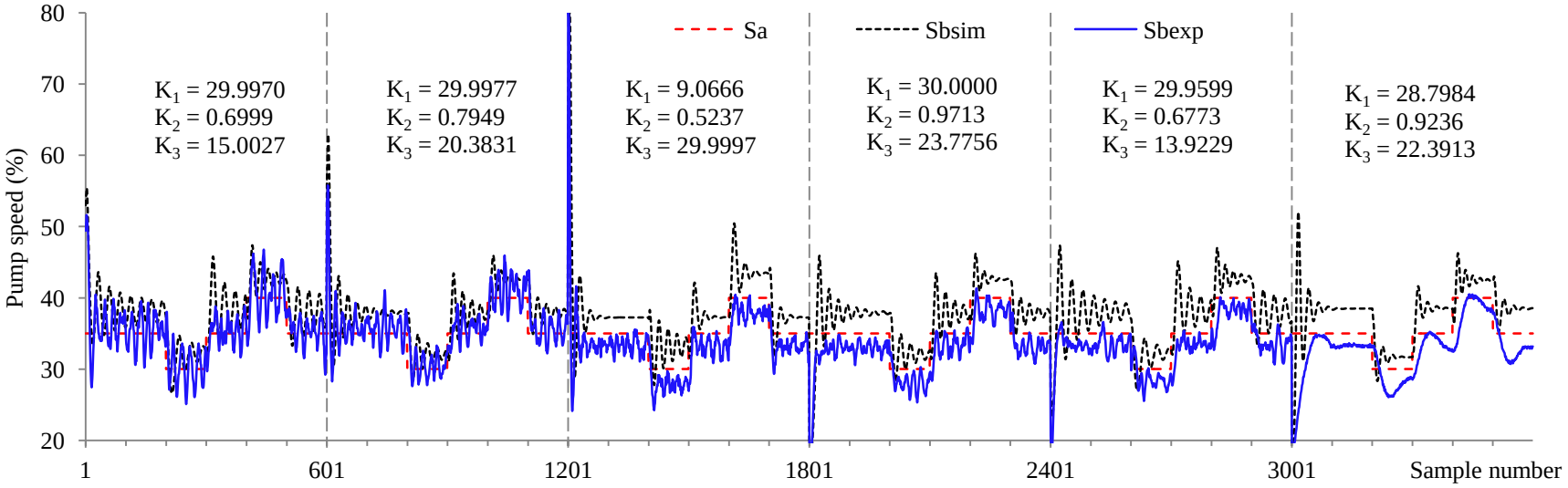


Figure 5.26(b) Simulated and experimental pumps speed variations of offline DE algorithm based piecewise FLC for SR operations

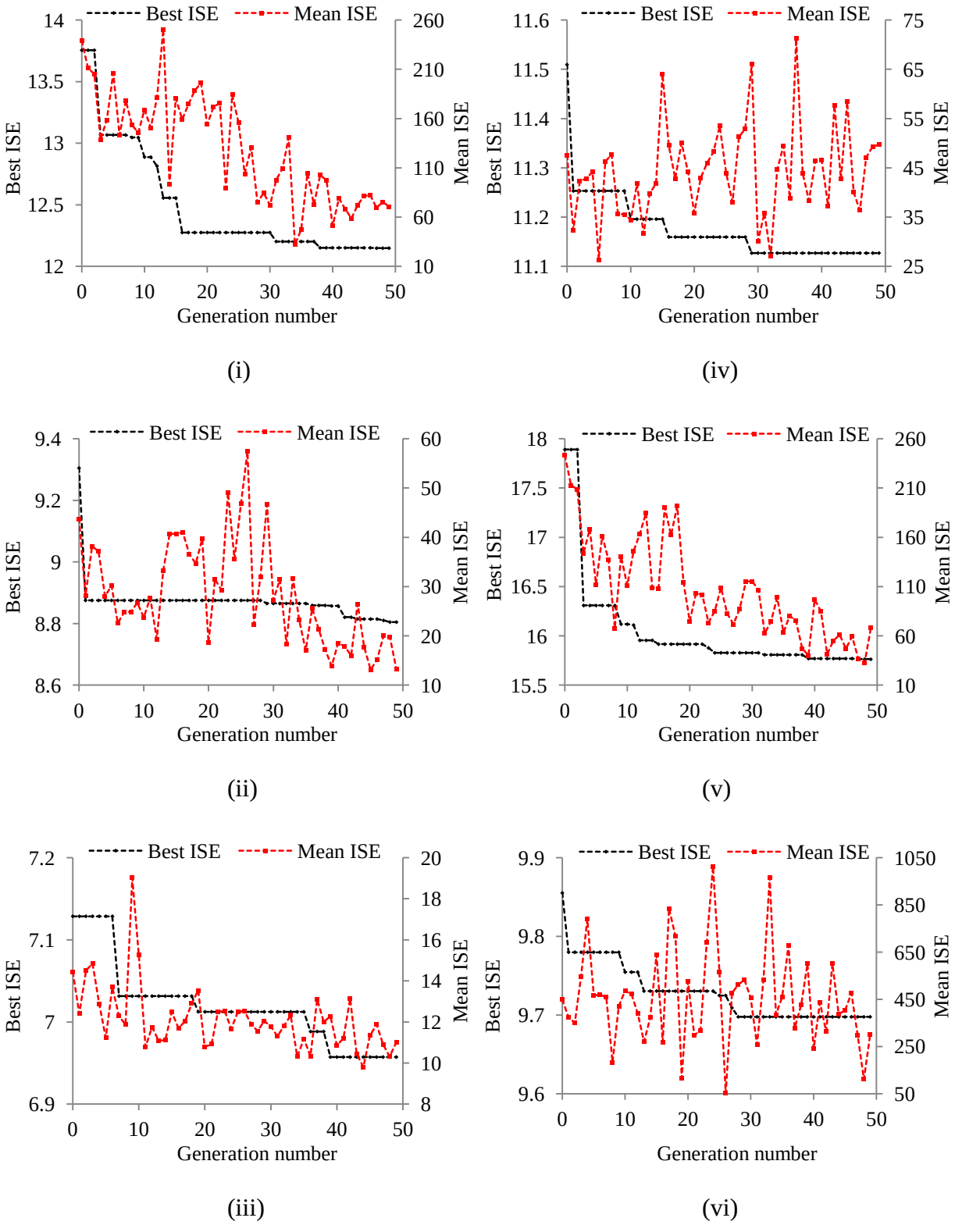


Figure 5.27(a) Best and mean ISE values of offline PSO algorithm based piecewise FLC for SR operations (i) SR1 (ii) SR2 (iii) SR3 (iv) SR4 (v) SR5 (vi) SR6

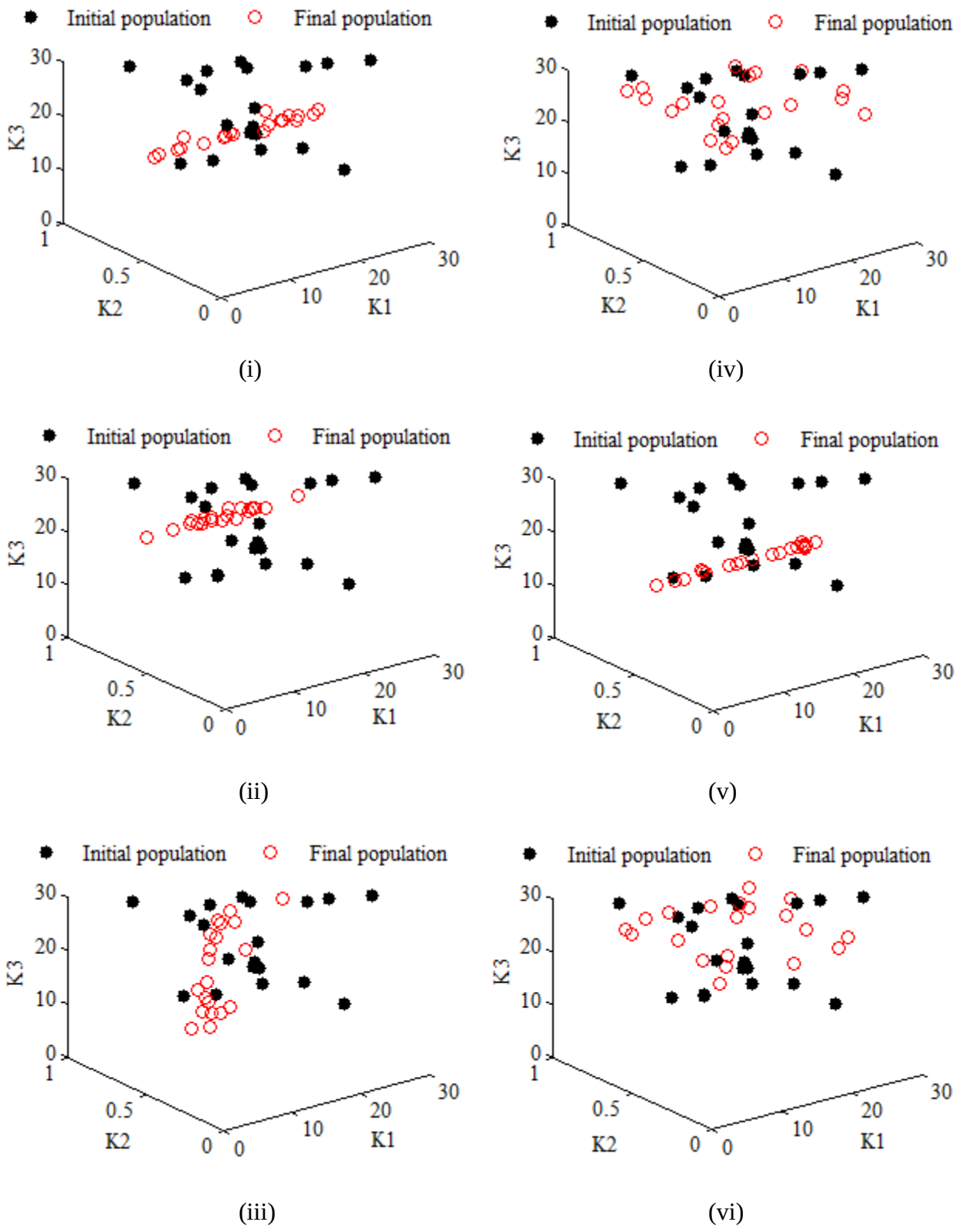


Figure 5.27(b) Initial and final particles positions of offline PSO algorithm based piecewise FLC for SR operations (i) SR1 (ii) SR2 (iii) SR3 (iv) SR4 (v) SR5 (vi) SR6

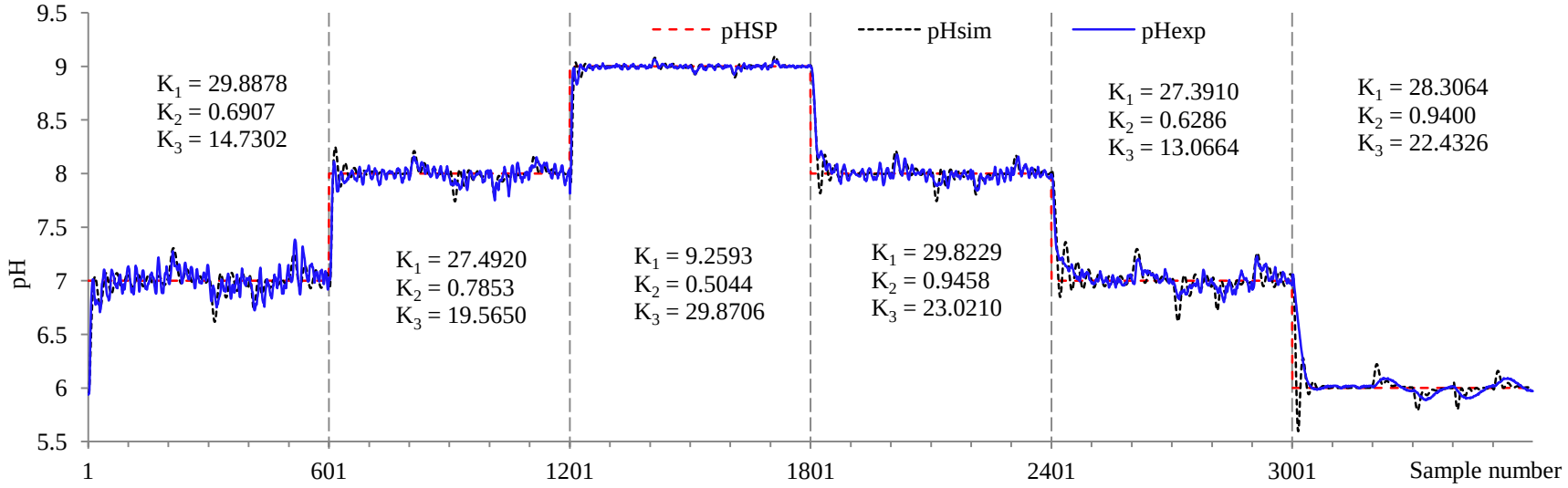


Figure 5.28(a) Simulated and experimental pH responses of offline PSO algorithm based piecewise FLC for SR operations

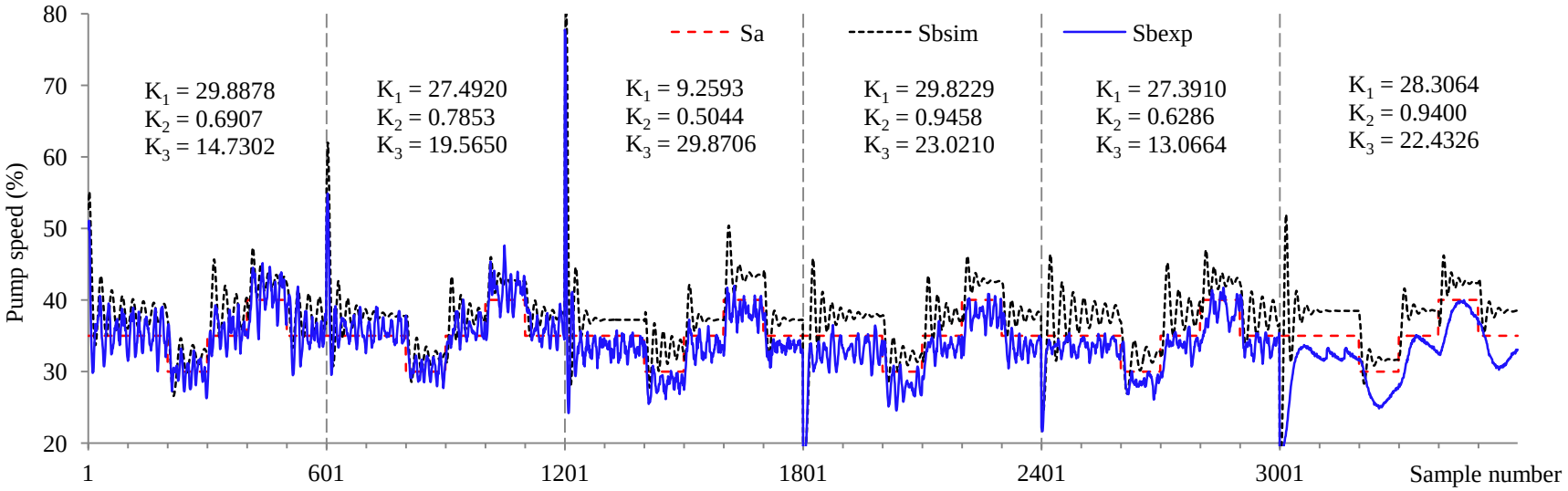


Figure 5.28(b) Simulated and experimental pumps speed variations of offline PSO algorithm based piecewise FLC for SR operations

Table 5.8(a) Simulation results of offline GA, DE, and PSO based piecewise FLC for SR operations

Optimization methods	Optimized parameters [K1, K2, K3]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	[28.3724, 0.8441, 15.3606]	6, 7, 35	7.1098, -0.2046	7, 35, 30	1.6801, -0.3427	7, 30, 35	1.5939, 0.3669	7, 35, 40	0.8035, 0.2956	7, 40, 35	1.0675, -0.2838
DE	[29.9970, 0.6999, 15.0027]		7.3889, -0.0796		1.2908, -0.3006		1.7986, 0.3855		0.7049, 0.2752		0.9421, -0.2468
PSO	[29.8878, 0.6907, 14.7302]		7.4108, -0.0701		1.3141, -0.3031		1.7706, 0.3838		0.7140, 0.2762		0.9384, -0.2485
GA	[26.7230, 0.7950, 19.0777]	7, 8, 35	6.9593, -0.2435	8, 35, 30	0.5229, -0.2111	8, 30, 35	0.6854, 0.2615	8, 35, 40	0.3908, 0.2011	8, 40, 35	0.3171, -0.1698
DE	[29.9977, 0.7949, 20.3831]		6.8867, -0.2536		0.5291, -0.2097		0.6926, 0.2592		0.3595, 0.1901		0.3082, -0.1647
PSO	[27.4920, 0.7853, 19.5650]		6.9299, -0.2464		0.5170, -0.2092		0.6804, 0.2593		0.3671, 0.1952		0.3106, -0.1673
GA	[8.1809, 0.4979, 29.4345]	8, 9, 35	6.6699, -0.0475	9, 35, 30	0.0796, -0.0785	9, 30, 35	0.0519, 0.0778	9, 35, 40	0.0965, 0.1033	9, 40, 35	0.0740, -0.0927
DE	[9.0666, 0.5237, 29.9997]		6.6309, -0.0596		0.0713, -0.0797		0.0471, 0.0743		0.0978, 0.1046		0.0800, -0.0952
PSO	[9.2593, 0.5044, 29.8706]		6.6711, -0.0360		0.0683, -0.0788		0.0463, 0.0741		0.0943, 0.1036		0.0772, -0.0940

Table 5.8(a) continued

Optimization methods	Optimized parameters [K1, K2, K3]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	[29.7914, 0.9453, 23.2001]	9, 8, 35	9.2945, 0.1786	8, 35, 30	0.5053, -0.2102	8, 30, 35	0.6628, 0.2558	8, 35, 40	0.3442, 0.1948	8, 40, 35	0.3048, -0.1680
DE	[30.0000, 0.9713, 23.7756]		9.2872, 0.1828		0.5013, -0.2098		0.6602, 0.2554		0.3410, 0.1949		0.3042, -0.1682
PSO	[29.8229, 0.9458, 23.0210]		9.3011, 0.1846		0.5051, -0.2102		0.6638, 0.2566		0.3507, 0.1963		0.3064, -0.1686
GA	[25.2150, 0.5883, 12.2996]	8, 7, 35	11.2082, 0.1048	7, 35, 30	1.2413, -0.2935	7, 30, 35	1.7024, 0.3793	7, 35, 40	0.7242, 0.2748	7, 40, 35	0.9466, -0.2583
DE	[29.9599, 0.6773, 13.9229]		11.0751, 0.2070		1.2890, -0.2983		1.6843, 0.3773		0.7559, 0.2791		0.9440, -0.2592
PSO	[27.3910, 0.6286, 13.0664]		11.1294, 0.1506		1.2555, -0.2949		1.6960, 0.3785		0.7385, 0.2769		0.9440, -0.2580
GA	[28.3210, 0.9251, 22.4813]	7, 6, 35	8.0562, 0.3936	6, 35, 30	0.5577, -0.2192	6, 30, 35	0.4999, 0.2088	6, 35, 40	0.3327, 0.1991	6, 40, 35	0.2481, -0.1581
DE	[28.7984, 0.9236, 22.3913]		8.0308, 0.3914		0.5672, -0.2198		0.5070, 0.2088		0.3364, 0.1992		0.2498, -0.1580
PSO	[28.3064, 0.9400, 22.4326]		8.0364, 0.4037		0.5672, -0.2208		0.5091, 0.2101		0.3356, 0.1995		0.2492, -0.1580

Table 5.8(b) Experimental performance of offline GA, DE, and PSO based piecewise FLC for SR operations

Optimization methods	Optimized parameters [K1, K2, K3]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	[28.3724, 0.8441, 15.3606]	6, 7, 35	7.5850, -0.3660	7, 35, 30	0.9130, -0.2890	7, 30, 35	1.1999, 0.3830	7, 35, 40	1.7668, 0.3060	7, 40, 35	2.7610, -0.4810
DE	[29.9970, 0.6999, 15.0027]		9.1798, -0.2120		1.6923, -0.3280		0.9511, 0.2160		1.9254, 0.3190		0.9066, -0.2250
PSO	[29.8878, 0.6907, 14.7302]		7.6131, -0.2190		1.0357, -0.2760		1.0554, 0.2480		1.4269, 0.2740		1.9537, -0.3850
GA	[26.7230, 0.7950, 19.0777]	7, 8, 35	6.7773, -0.1600	8, 35, 30	0.3264, -0.1210	8, 30, 35	0.4640, 0.1600	8, 35, 40	0.6399, 0.2310	8, 40, 35	0.4094, -0.1340
DE	[29.9977, 0.7949, 20.3831]		6.3891, -0.1150		0.4526, -0.1920		0.4737, 0.1670		0.7382, 0.1920		0.6300, -0.1530
PSO	[27.4920, 0.7853, 19.5650]		6.1576, -0.1150		0.4155, -0.1600		0.4996, 0.1540		0.7410, 0.2500		0.4515, -0.1530
GA	[8.1809, 0.4979, 29.4345]	8, 9, 35	5.0826, -0.0490	9, 35, 30	0.0389, -0.0620	9, 30, 35	0.0478, 0.0660	9, 35, 40	0.0623, 0.0850	9, 40, 35	0.0502, -0.0680
DE	[9.0666, 0.5237, 29.9997]		5.5223, -0.0360		0.0467, -0.0680		0.0583, 0.0790		0.0385, 0.0530		0.0393, -0.0680
PSO	[9.2593, 0.5044, 29.8706]		5.0799, -0.0300		0.0442, -0.0680		0.0529, 0.0730		0.0571, 0.0790		0.0329, -0.0550

Table 5.8(b) continued

Optimization methods	Optimized parameters [K1, K2, K3]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	[29.7914, 0.9453, 23.2001]	9, 8, 35	9.8241, 0.0580	8, 35, 30	0.2894, -0.1280	8, 30, 35	0.3756, 0.1350	8, 35, 40	0.3924, 0.1160	8, 40, 35	0.2286, -0.1400
DE	[30.0000, 0.9713, 23.7756]		9.9012, 0.0450		0.2741, -0.1280		0.3999, 0.1670		0.3855, 0.1860		0.2595, -0.1210
PSO	[29.8229, 0.9458, 23.0210]		10.0547, 0.1090		0.4258, -0.1790		0.3424, 0.1350		0.4196, 0.1600		0.2793, -0.1600
GA	[25.2150, 0.5883, 12.2996]	8, 7, 35	9.2810, 0.0880	7, 35, 30	0.4356, -0.1550	7, 30, 35	0.6468, 0.1650	7, 35, 40	0.8374, 0.2040	7, 40, 35	0.6422, -0.2000
DE	[29.9599, 0.6773, 13.9229]		8.3542, 0.1010		0.6315, -0.1930		0.5663, 0.1720		0.6041, 0.1910		0.4789, -0.1480
PSO	[27.3910, 0.6286, 13.0664]		8.4332, 0.0630		0.6920, -0.2060		0.6165, 0.1720		0.6720, 0.1970		0.8410, -0.2320
GA	[28.3210, 0.9251, 22.4813]	7, 6, 35	13.5112, 0.0350	6, 35, 30	0.4731, -0.1250	6, 30, 35	0.4183, 0.1190	6, 35, 40	0.3087, 0.0990	6, 40, 35	0.3334, -0.1050
DE	[28.7984, 0.9236, 22.3913]		12.1141, 0.0290		0.2905, -0.1050		0.3252, 0.1060		0.3864, 0.1120		0.3408, -0.1050
PSO	[28.3064, 0.9400, 22.4326]		14.1167, 0.0160		0.2916, -0.0930		0.3764, 0.1120		0.3373, 0.0990		0.2903, -0.0930

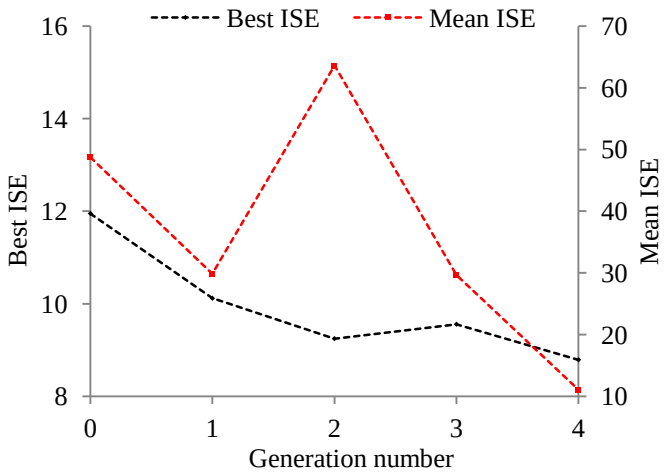


Figure 5.29(a) Best and mean ISE values of online GA optimization based FLC for SR1 operation

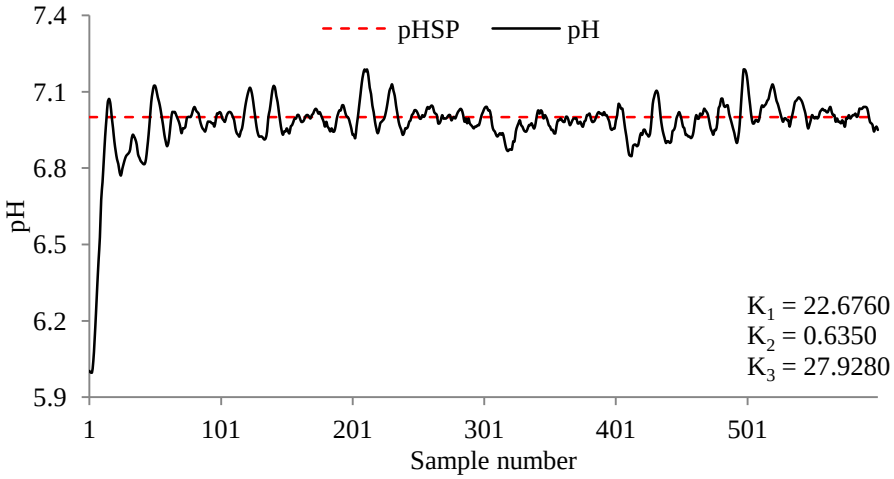


Figure 5.30(a) Experimental pH response of online GA optimization based FLC for SR1 operation

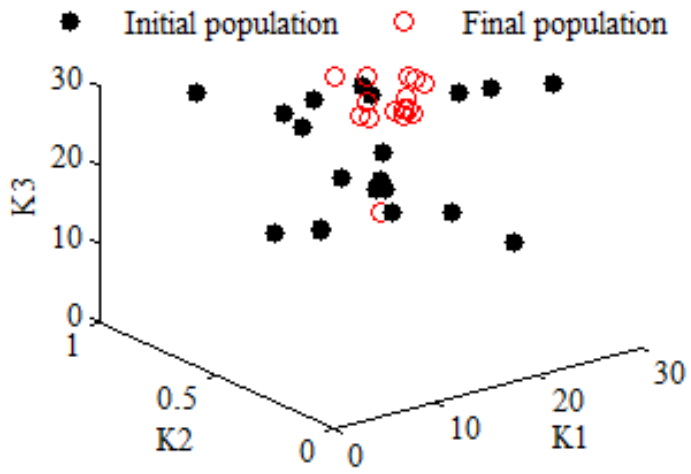


Figure 5.29(b) Initial and final population members of online GA optimization based FLC for SR1 operation

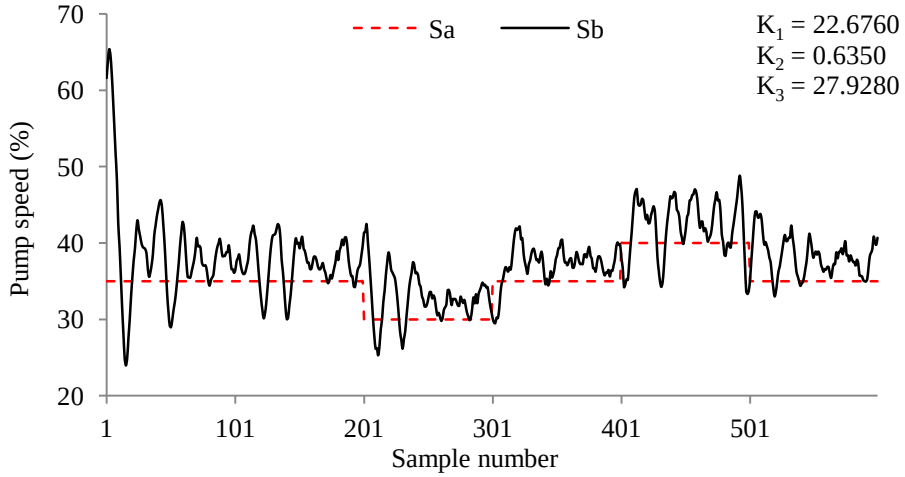


Figure 5.30(b) Experimental pumps speed variations of online GA optimization based FLC for SR1 operation

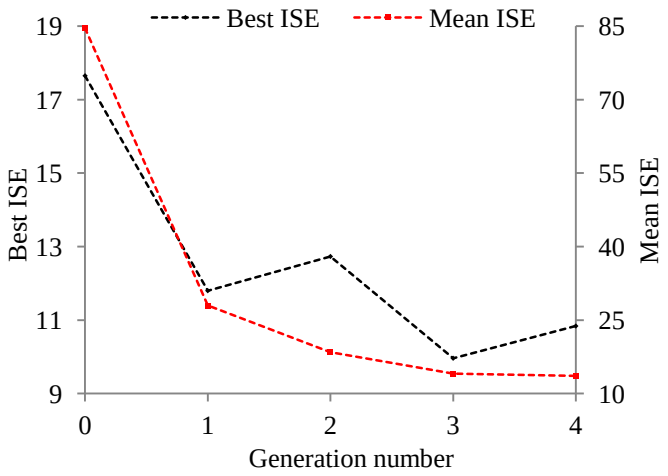


Figure 5.31(a) Best and mean ISE values of online GA optimization based FLC for SR5 operation

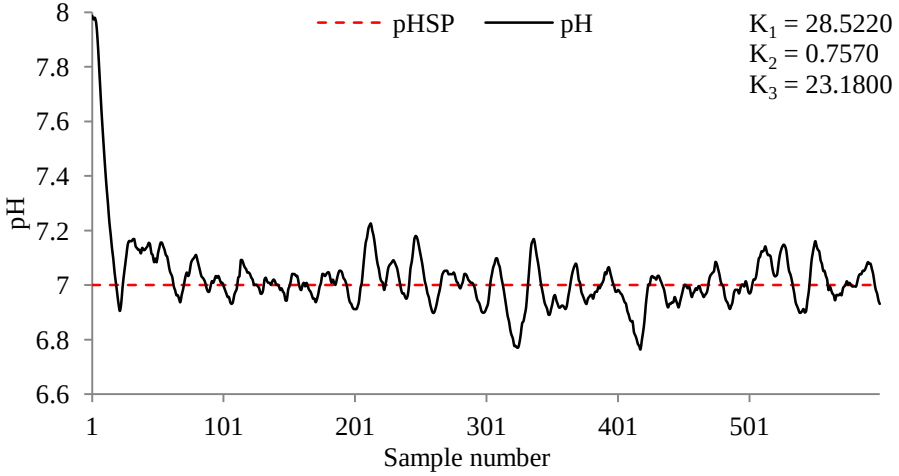


Figure 5.32(a) Experimental pH response of online GA optimization based FLC for SR5 operation

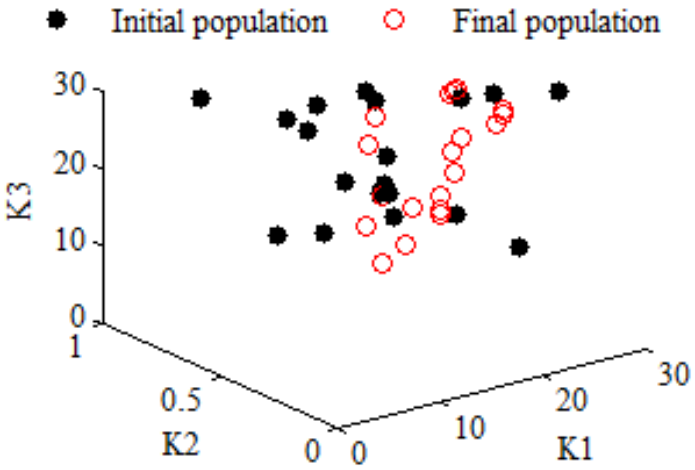


Figure 5.31(b) Initial and final population members of online GA optimization based FLC for SR5 operation

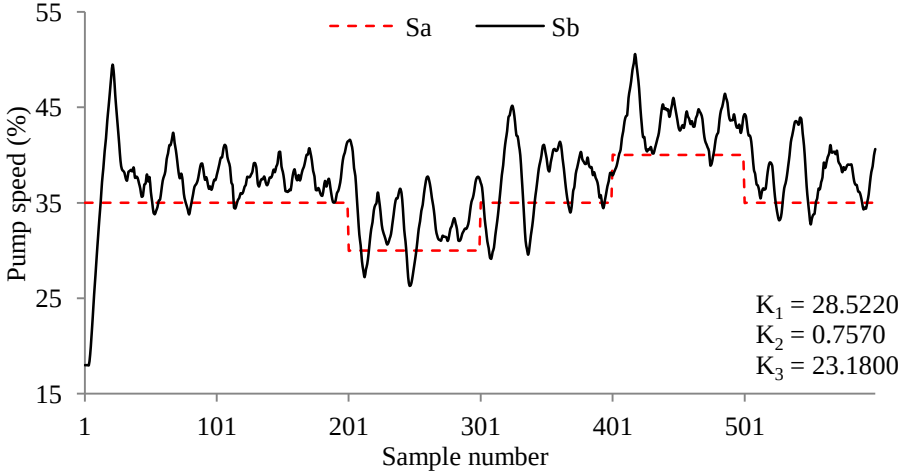


Figure 5.32(b) Experimental pumps speed variations of online GA optimization based FLC for SR5 operation

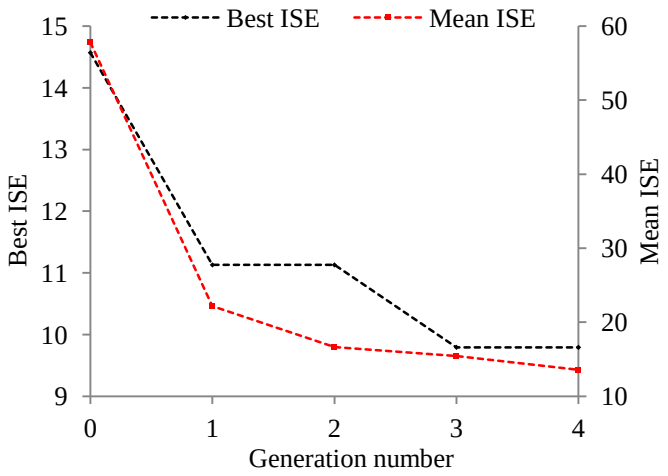


Figure 5.33(a) Best and mean ISE values of online DE algorithm based FLC for SR1 operation

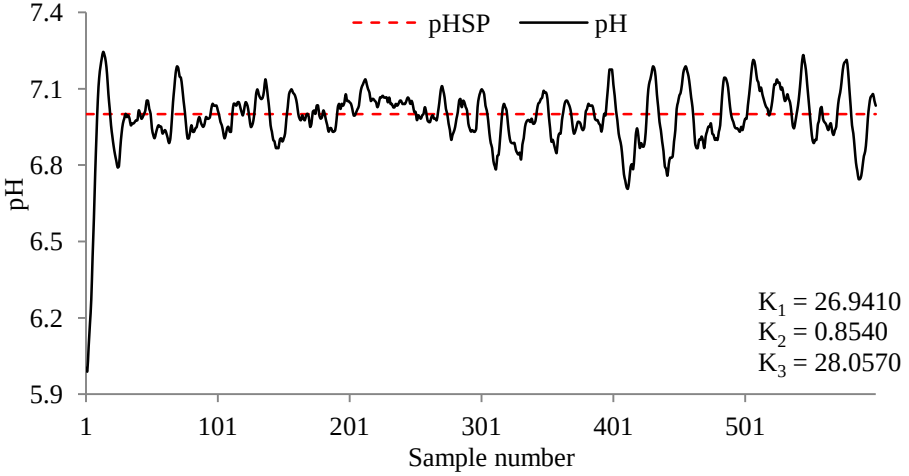


Figure 5.34(a) Experimental pH response of online DE algorithm based FLC for SR1 operation

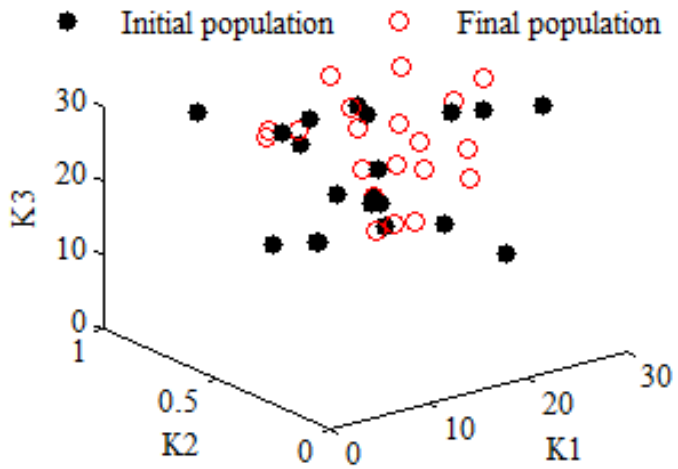


Figure 5.33(b) Initial and final population members of online DE algorithm based FLC for SR1 operation

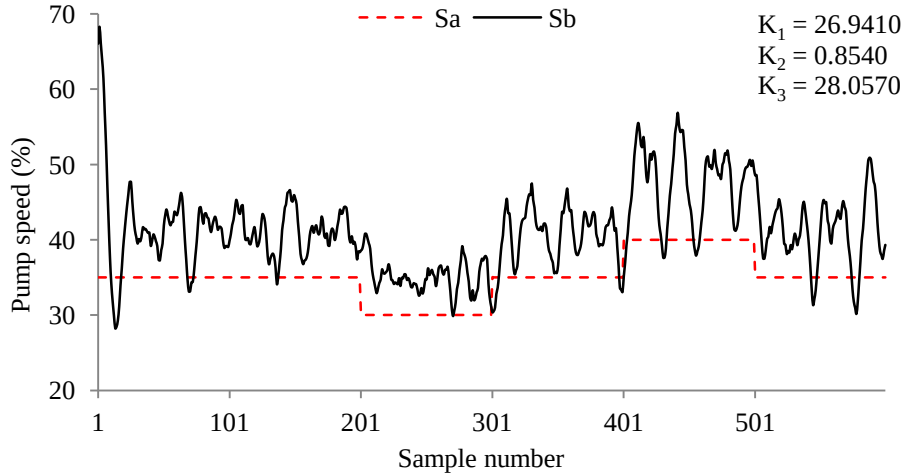


Figure 5.34(b) Experimental pumps speed variations of online DE algorithm based FLC for SR1 operation

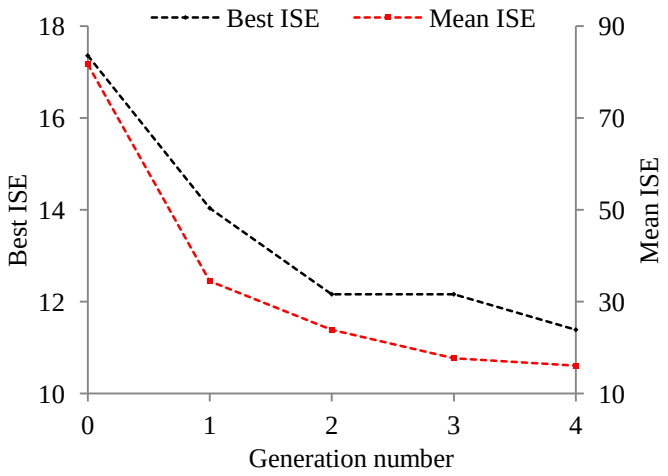


Figure 5.35(a) Best and mean ISE values of online DE algorithm based FLC for SR5 operation

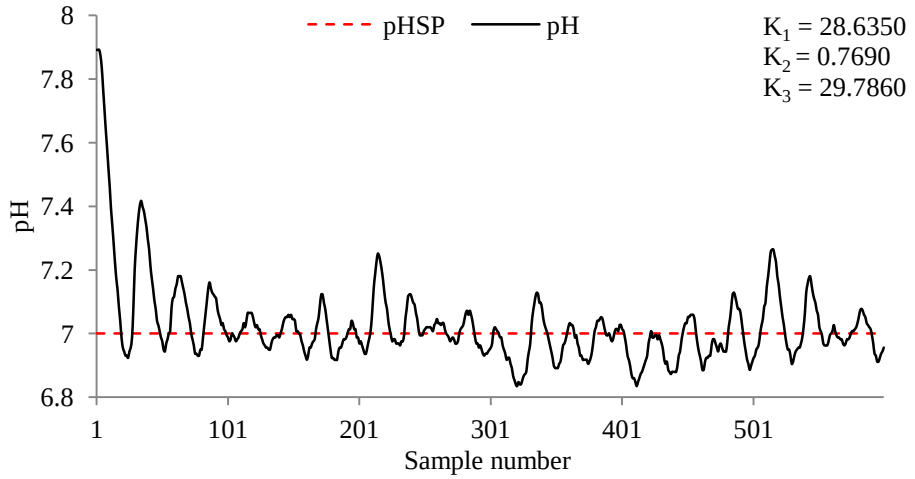


Figure 5.36(a) Experimental pH response of online DE algorithm based FLC for SR5 operation

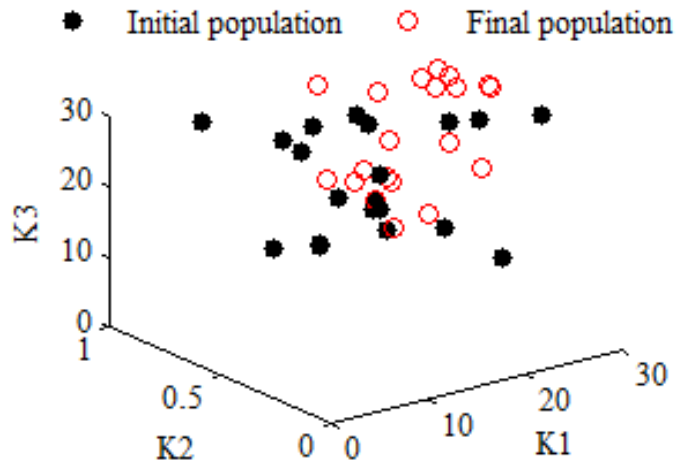


Figure 5.35(b) Initial and final population members of online DE algorithm based FLC for SR5 operation

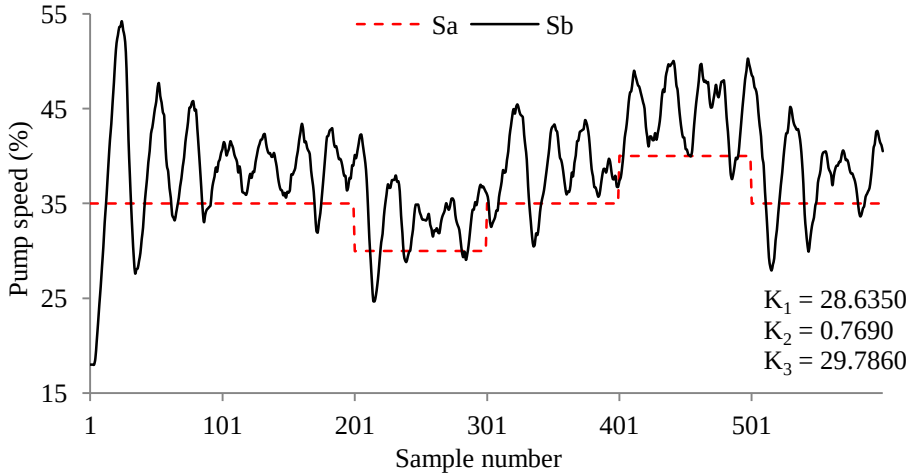


Figure 5.36(b) Experimental pumps speed variations of online DE algorithm based FLC for SR5 operation

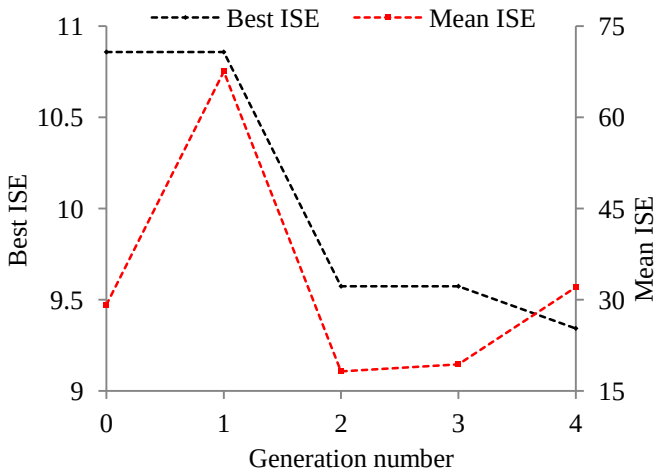


Figure 5.37(a) Best and mean ISE values of online PSO algorithm based FLC for SR1 operation

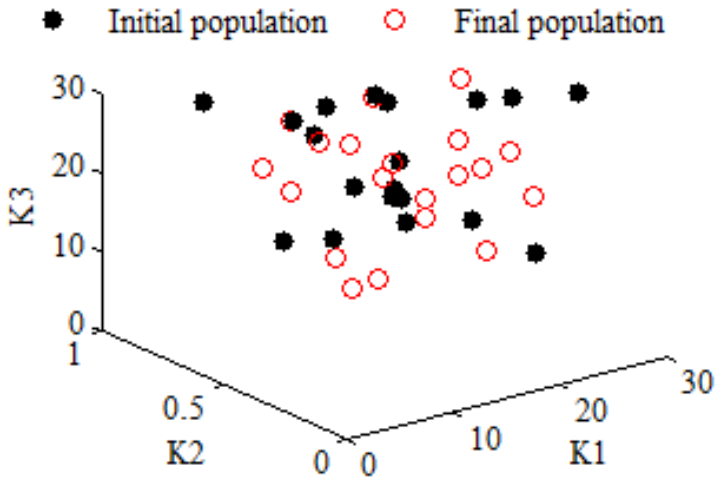


Figure 5.37(b) Initial and final particles positions of online PSO algorithm based FLC for SR1 operation

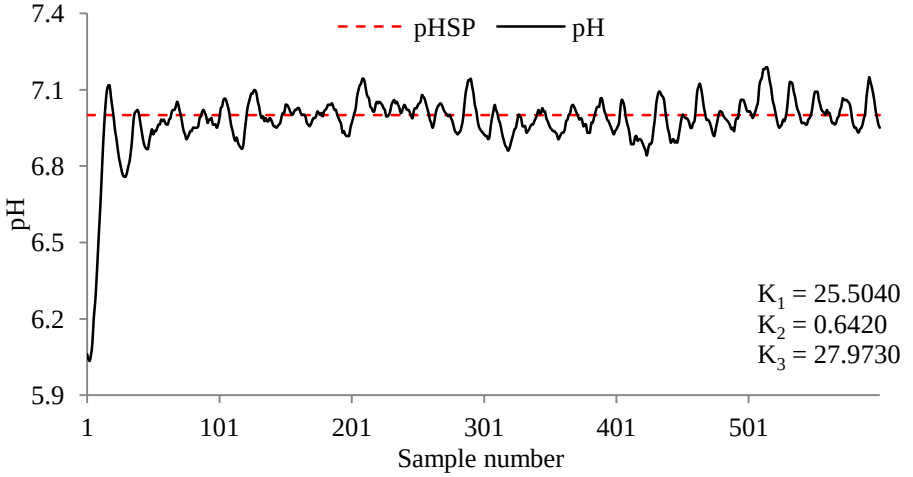


Figure 5.38(a) Experimental pH response of online PSO algorithm based FLC for SR1 operation

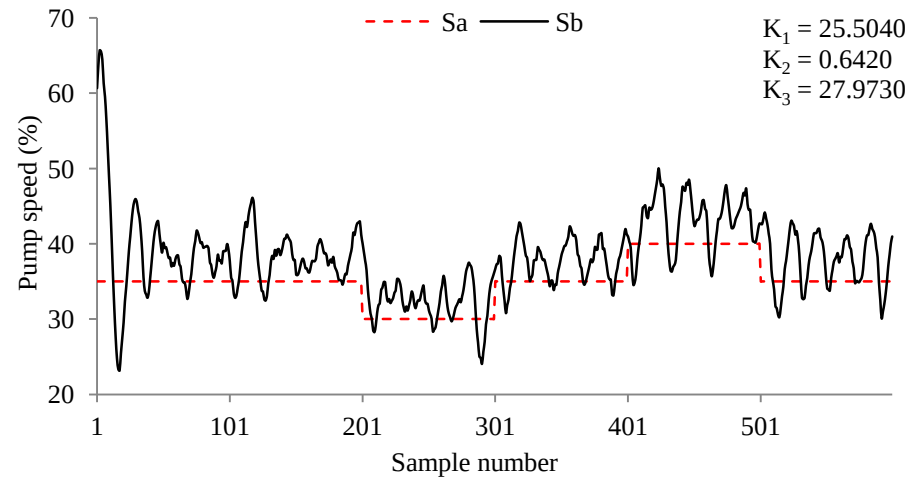


Figure 5.38(b) Experimental pumps speed variations of online PSO algorithm based FLC for SR1 operation

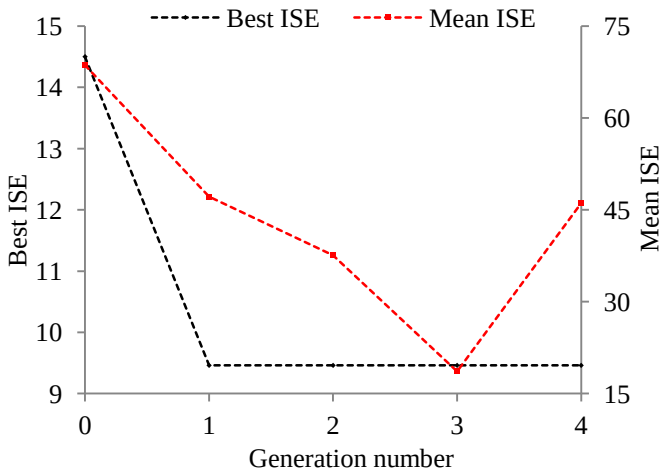


Figure 5.39(a) Best and mean ISE values of online PSO algorithm based FLC for SR5 operation

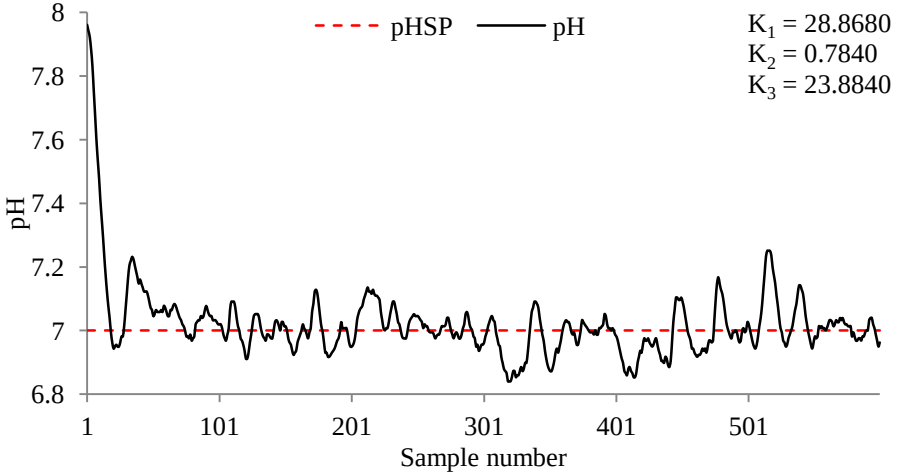


Figure 5.40(a) Experimental pH response of online PSO algorithm based FLC for SR5 operation

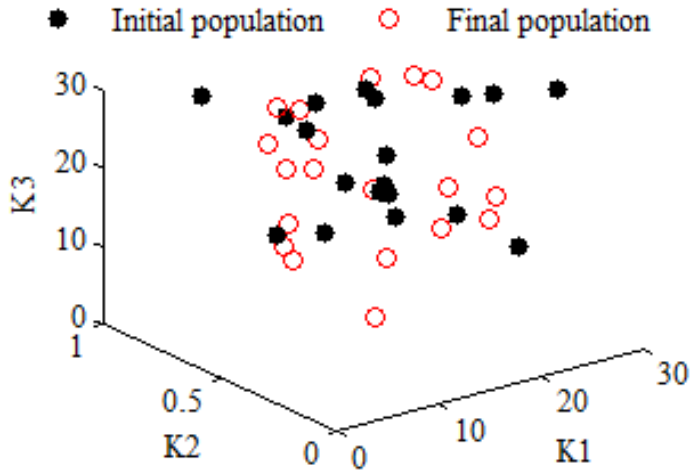


Figure 5.39(b) Initial and final particles positions of online PSO algorithm based FLC for SR5 operation

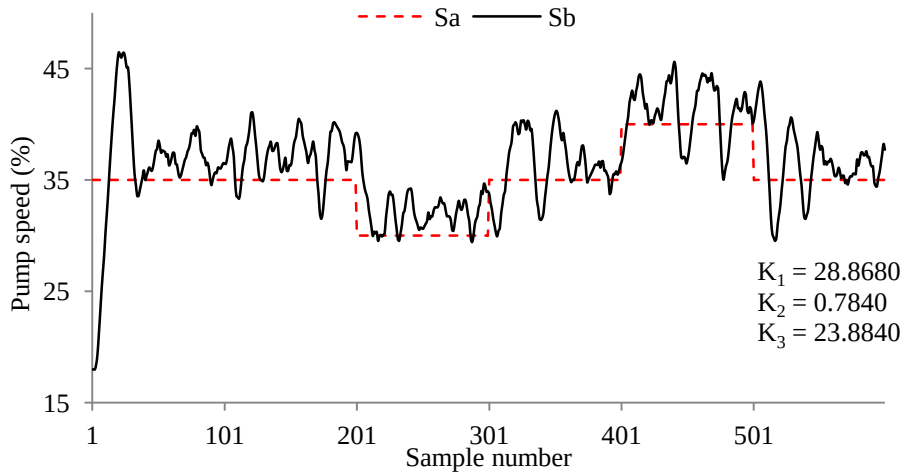


Figure 5.40(b) Experimental pumps speed variations of online PSO algorithm based FLC for SR5 operation

Table 5.9 Experimental performance of online GA, DE, and PSO based FLC for SR1 and SR5 operations

Optimization methods	Optimized parameters [K1, K2, K3]	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
		$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
GA	[22.6760, 0.6350, 27.9280]	6, 7, 35	7.4898, -0.1230	7, 35, 30	0.3514, -0.1870	7, 30, 35	0.2209, 0.1330	7, 35, 40	0.5320, 0.1520	7, 40, 35	0.1916, -0.1420
DE	[26.9410, 0.8540, 28.0570]		5.7688, -0.2440		0.3074, -0.1360		0.7655, 0.2160		1.6462, 0.2930		1.3037, -0.2320
PSO	[25.5040, 0.6420, 27.9730]		7.7128, -0.1160		0.3348, -0.1420		0.3085, 0.1400		0.4816, 0.1590		0.5047, -0.1870
GA	[28.5220, 0.7570, 23.1800]	8, 7, 35	8.1494, 0.0950	7, 35, 30	0.7152, -0.2250	7, 30, 35	0.8352, 0.2290	7, 35, 40	0.5698, 0.2360	7, 40, 35	0.5644, -0.1610
DE	[28.6350, 0.7690, 29.7860]		8.8038, 0.0820		0.5738, -0.2510		0.5451, 0.1650		0.6292, 0.1650		0.8368, -0.2640
PSO	[28.8680, 0.7840, 23.8840]		7.4239, 0.0880		0.2915, -0.1360		0.5208, 0.1590		0.5959, 0.1460		0.6293, -0.2510

5.8 Concluding Remarks

In this chapter, first feedback control of Armfield pH neutralization process using linear PID controller and nonlinear Mamdani based direct fuzzy logic controller for servo-regulatory operations has been developed. The servo and regulatory (SR) operations in pH neutralization process for 3600 seconds are defined as follows: starting with pH setpoint as 6, pH setpoint is changed to 7, 8, 9, 8, 7, and 6 at interval of 600 seconds, keeping acid flow rate at 35%; 200 seconds after every change of pH setpoint, acid flow rate is changed from 35% to 30%, 30% to 35%, 35% to 40%, and 40% to 35%, consecutively after every 100 seconds. Thus, SR operations consist of six cases, of equal duration, namely SRi where i = 1, 2, 3, 4, 5, and 6, each part involves servo operation of 200 seconds followed by total regulatory operations of 400 seconds. In order to tune pH controller parameters either offline using ANN based dynamic pH model or online using Armfield pH neutralization process, we have used global optimization techniques namely GA, DE, and PSO. The objective function for global optimization techniques is ISE. Offline simulations of GA optimization, DE optimization, and PSO based PID controller gives total ISE for SR operations as 80.6441, 80.6644, and 80.7496, whereas experimental validations gives total ISE as 411.7163, 577.2561, and 126.1982, respectively. Offline simulations of GA optimization, DE optimization, and PSO based fuzzy logic controller gives total ISE for SR operations as 73.8843, 71.9779, and 72.2608, whereas experimental validations gives total ISE as 80.3776, 67.9637, and 67.9266, respectively. These results demonstrate that nonlinear Mamdani based fuzzy controller is better than linear PID controller, for pH control of SR1, SR2, SR5, and SR6 operations. Offline simulations of GA optimization, DE optimization, and PSO based piecewise fuzzy logic controller gives total ISE for SR operations as 64.7311, 64.3618, and 64.4981, whereas experimental validations gives total ISE as 66.1221, 64.3561, and 64.8058, respectively. Use of offline optimized piecewise fuzzy logic controller for SR operations brings ISE values down by amount 14.2555 for GA, 3.6076 for DE, and 3.1208 for PSO. The offline optimization results show that pH control for SR1 and SR5 operations are most challenging task. Online optimization of piecewise fuzzy logic controller gives ISE values for SR1 and SR5 operations as follows: 8.7858 and 10.8342 for GA, 9.7916 and 11.3888 for DE, and 9.3424 and 9.4614 for PSO, respectively. From the above results it is clear that all three global optimization techniques give approximately similar solutions, but final population members from DE optimization have better convergence than PSO followed by GA optimization.

CHAPTER 6

SELF-TUNED FUZZY LOGIC BASED pH CONTROL

6.1 Introduction

Nonlinear processes require automatic control since their parameter values alter either as the operating point changes or over time or both. A tuned nonlinear controller can give good performance at a particular operating point for a limited period of time. The controller needs to be: (i) retuned if the operating point changes as in servo operations, (ii) retuned periodically if either process parameters change with time or process transfer function changes with introduction of disturbance, as in regulatory operations. This necessity to retune controller has driven the need for self-tuned controllers that can automatically retune themselves to match the current process conditions or characteristics. In this chapter, first design of self-tuned FLC scheme has been presented. Next self-tuned fuzzy logic controller performance is evaluated experimentally. Also the self-tuned controller performance is compared with GA, DE, and PSO based optimized fuzzy logic controller for servo and regulatory operations.

6.2 Design of Self-Tuned Fuzzy Logic Control (FLC) Scheme

In Chapter 5, section 5.4 presents design of FLC scheme for pH neutralization process. Self-tuned FLC scheme contain two additional components on top of the FLC scheme. The first component is a 'process monitor' that detects changes in the process characteristics in terms of performance measure to assess how well the controller is controlling. The second component is the 'tuning mechanism' which uses information passed to it by the process monitor to update the controller parameters. Figure 6.1(a) shows block diagram of self-tuned FLC for Armfield pH neutralization process using Mamdani FIS based fuzzy logic controller structure as shown in Figure 5.3. Table 5.2 shows summary of Mamdani FIS specifications. The self-tuned fuzzy logic controller for pH neutralization process has input variables defined as error $e(k) = pH_{SP}(k) - pH(k)$ and change in error $ce(k) = e(k) - e(k-1) = pH(k-1) - pH(k)$, where pH_{SP} is the setpoint, at k^{th} sampling instant. After dividing the input variables $e(k)$ and $ce(k)$ with scaling

factors K_1 and K_2, we obtain normalized error and change in error, $e^*(k)$ and $ce^*(k)$ respectively. After multiplying the normalized change in output $co^*_{FLC}(k)$, which is defuzzified output of Mamdani FIS, by the scaling factor K_3, we obtain output variable $co_{FLC}(k)$ of self-tuned fuzzy logic controller. The fuzzy membership functions for $e^*(k)$, $ce^*(k)$, and $co^*_{FLC}(k)$ are shown in Figures 5.4(a), 5.4(b), and 5.4(c) respectively. Table 5.3 shows the fuzzy rule table. The individual fuzzy rule can be represented using Table 5.3 and equation (5.22). In this thesis, we have used feedback control of Armfield pH neutralization process in which under nominal operating conditions Controlled Variable (CV) i.e. pH is maintained at a set-point value (pH_{SP}) with zero error as input to the pH controller, and Manipulated Variable (MV) i.e. speed of base pump B (S_b) and Disturbance Variable (DV) i.e. speed of acid pump A (S_a) have values MV0 = 38.5% and DV0 = 35% respectively.

The basic idea of self-tuned FLC scheme is to assign various discrete values to scaling factor K_3 depending upon instantaneous values of variables e(k) and ce(k). Equation (6.1) gives expression for K_3 as follows:

$$K_3 = K_{3A} \times K_{3M} \tag{6.1}$$

where K_{3A} is the discrete component to be determined using Table 6.1 and K_{3M} is the integral multiplier from 1 to 4.

The input variables e(k) and ce(k) are divided into following seven identical regions: e_1, $ce_1 \in$ [-6,-1); e_2, $ce_2 \in$ [-1,-0.5); e_3, $ce_3 \in$ [-0.5,-0.1); e_4, $ce_4 \in$ [-0.1,0.1]; e_5, $ce_5 \in$ (0.1,0.5]; e_6, $ce_6 \in$ (0.5,1]; e_7, $ce_7 \in$ (1,6]. To determine the values of K_{3A}, first appropriate region of e(k) and ce(k) needs to be identified. Suppose e(k) is e_1 and ce(k) is ce_1 i.e. pH is far away from pH_{SP} and moving rapidly away from pH_{SP}, so pH controller needs to take large corrective action. Thus we assign K_{3A} equals 8 for this case. Next suppose e(k) is e_1 and ce(k) is ce_7 i.e. pH is far away from pH_{SP} and moving rapidly toward pH_{SP}, so pH controller needs to take least corrective action. Thus we assign K_{3A} equals 2 for this case. Thus from Table 6.1 it is evident that K_{3A} has been assigned large values in order to ensure reduced settling time and as error decreases smaller values are assigned in order to ensure steady-state response within settling band. Therefore Table 6.1 entries based on empirical knowledge validate use of coarse control and fine control techniques. Figures 6.1(b) and 6.1(c) are showing LabVIEW front panel appearance and block

diagram implementation of self-tuned fuzzy logic based pH control on Armfield system. The LabVIEW block diagrams for Mamdani FIS based self-tuned fuzzy logic controller (AFL01) and adaptive gain calculator (AG01) are shown in Figures 6.1(d) to 6.1(e) respectively.

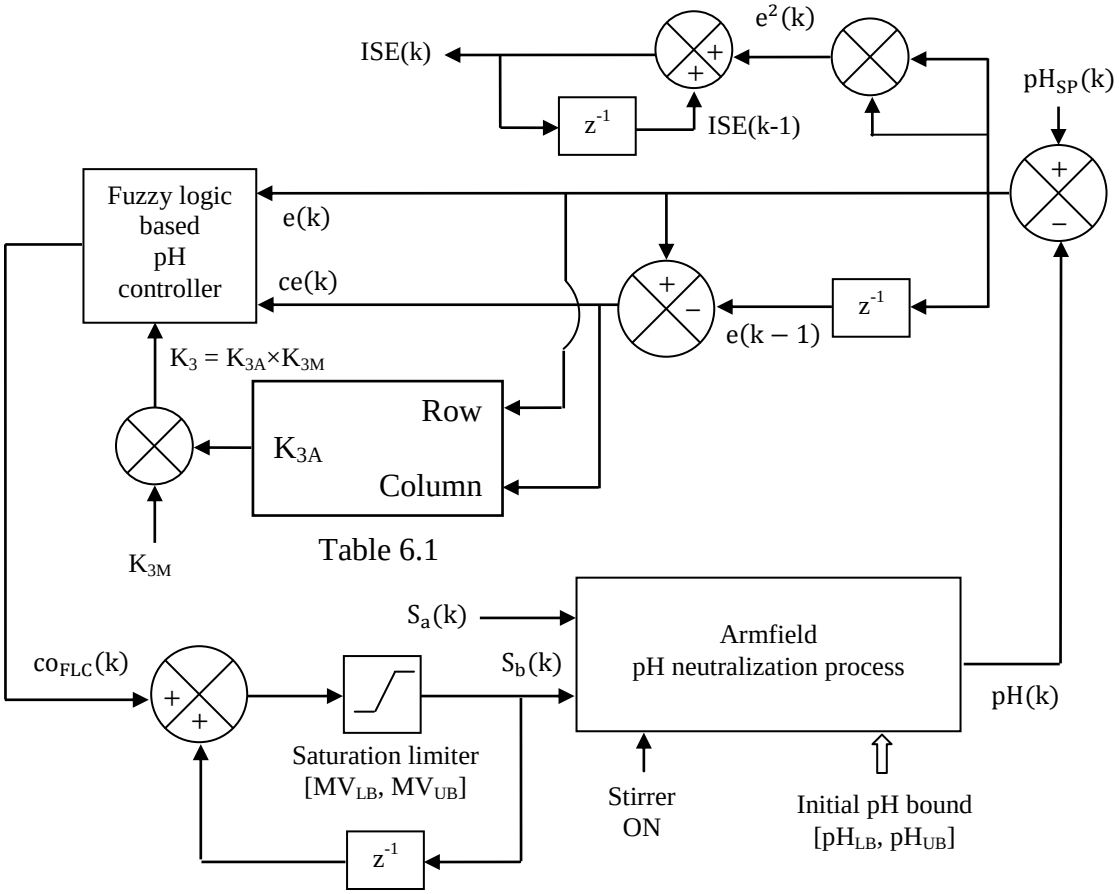


Figure 6.1(a) Block diagram of self-tuned FLC for Armfield pH neutralization process

Table 6.1 Determination of K_{3A}

		Range for 'ce'						
		ce_1	ce_2	ce_3	ce_4	ce_5	ce_6	ce_7
Range for 'e'	e_1	8	8	8	8	6	4	2
	e_2	8	8	8	6	4	2	4
	e_3	8	8	6	4	2	4	6
	e_4	8	6	4	2	4	6	8
	e_5	6	4	2	4	6	8	8
	e_6	4	2	4	6	8	8	8
	e_7	2	4	6	8	8	8	8

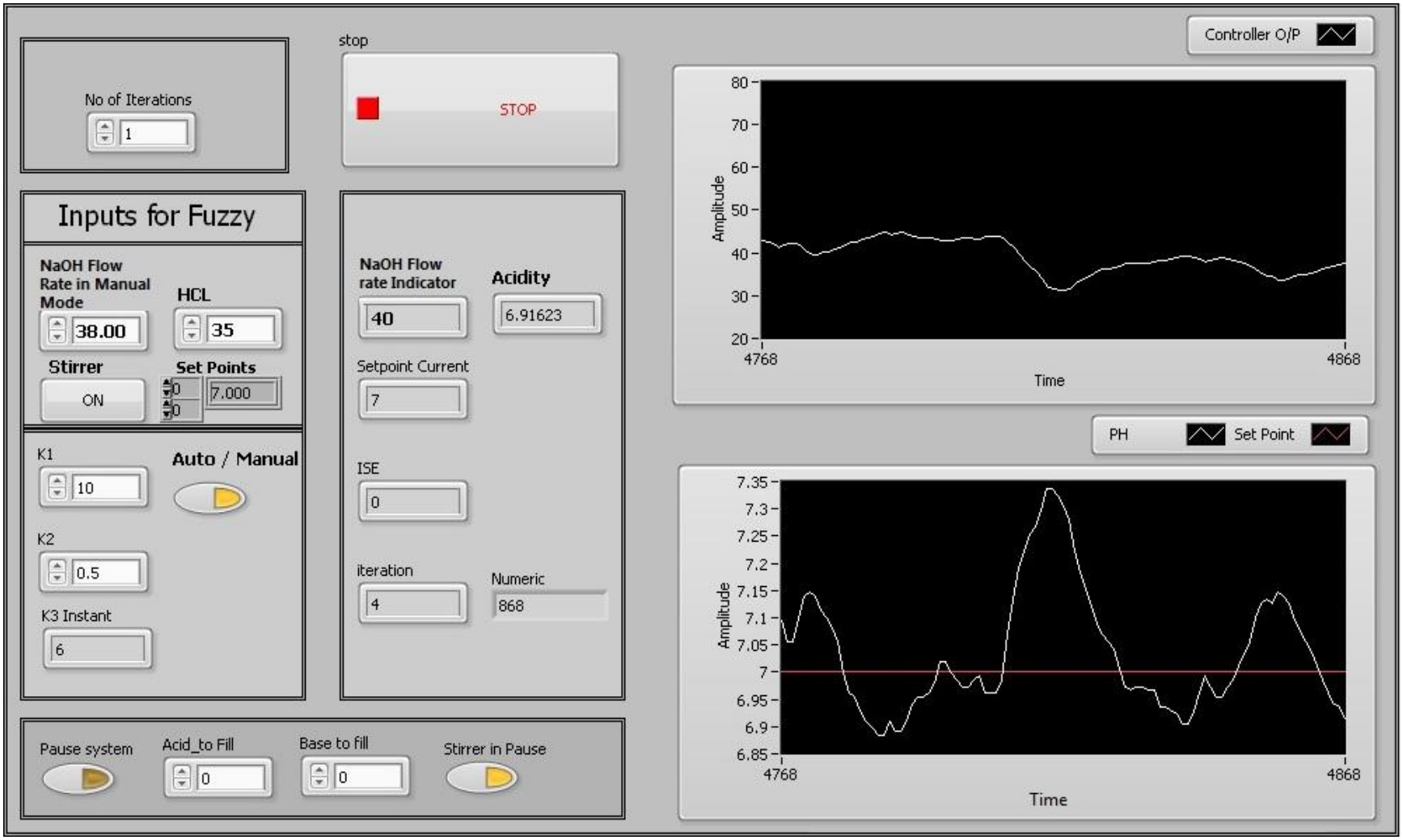


Figure 6.1(b) LabVIEW front panel appearance of self-tuned fuzzy logic based pH control

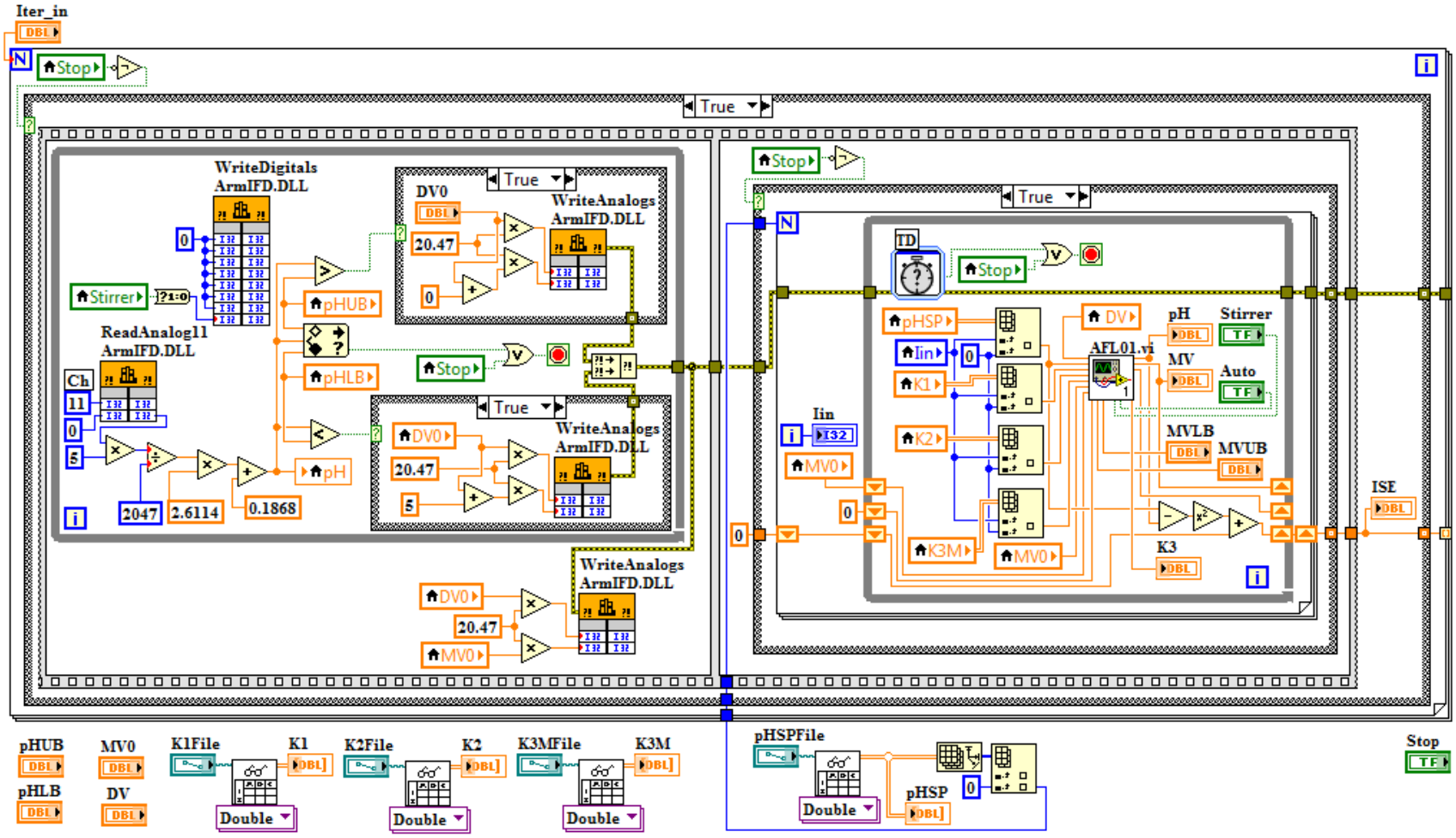


Figure 6.1(c) LabVIEW block diagram implementation of self-tuned fuzzy logic based pH control

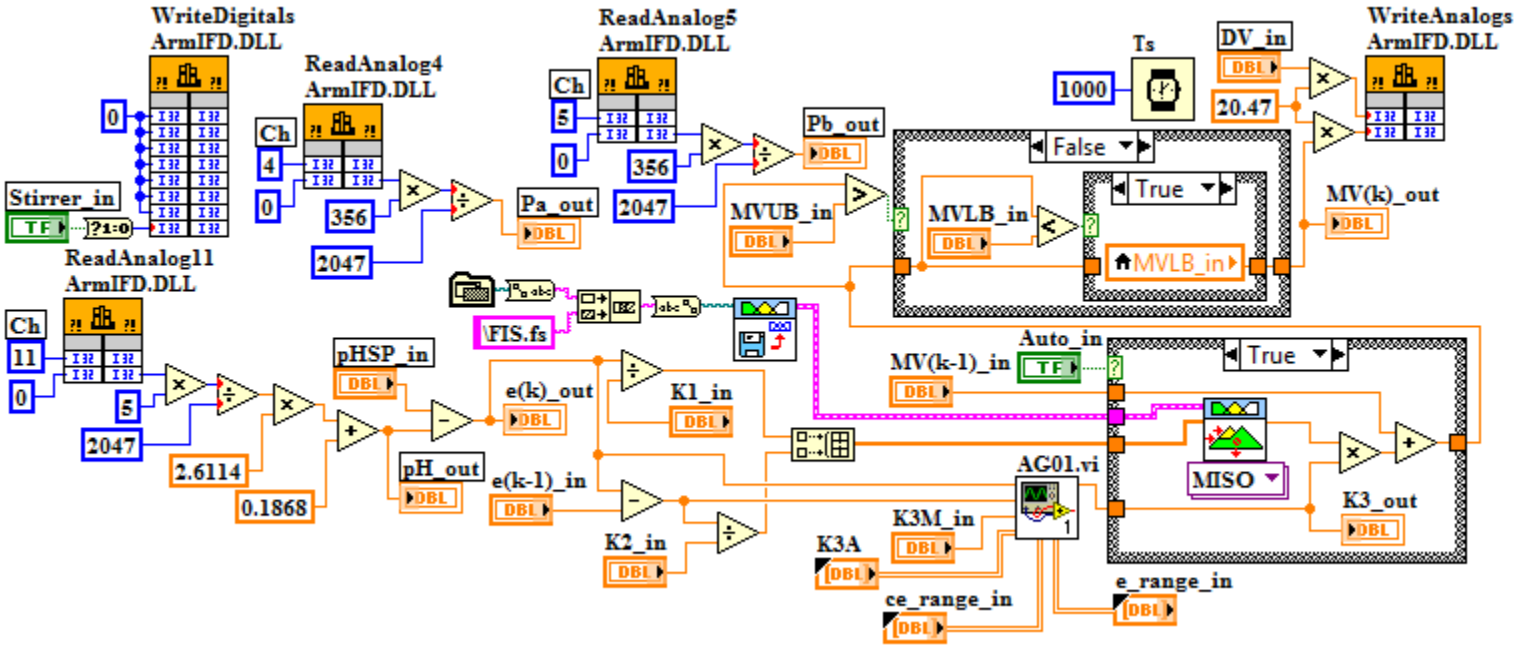


Figure 6.1(d) LabVIEW block diagram for Mamdani FIS based self-tuned fuzzy logic controller (AFL01)

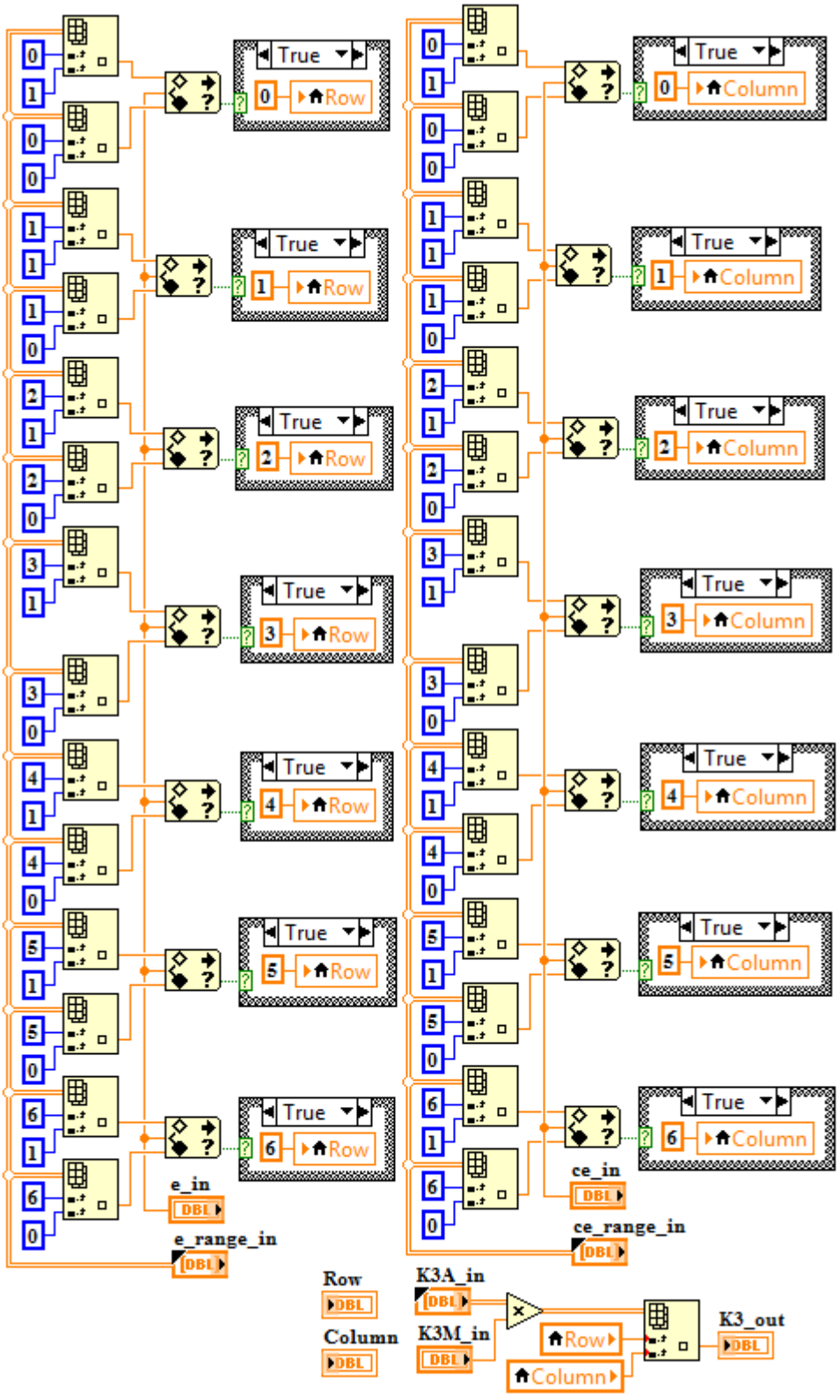


Figure 6.1(e) LabVIEW block diagram for adaptive gain calculator (AG01)

Start ***% Begin LabVIEW implementation***

for n = 1 to No. of Iterations

write in DLL to start the stirrer

read from DLL to obtain pH sensor voltage

estimate pH

while initial pH is not within range [pH_{LB} pH_{UB}] ***% Begin pH process initialization***

if pH < $pH_{LB} = (pH_{SP})_{initial} - 0.1$

write in DLL to set $S_a = 35$ and $S_b = 40$

end

if pH > $pH_{UB} = (pH_{SP})_{initial} + 0.1$

write in DLL to set $S_a = 35$ and $S_b = 35$

end

end ***% End pH process initialization***

initialize $S_a = 35$, $S_b = 38$, ISE = 0, final pH_{SP}

for m = 1 to Number of Set points ***% Begin self-tuned fuzzy control***

for k = 1 to Sampling Duration ***% For each set point***

estimate pH(k), e(k), and ce(k)

obtain $e^*(k) = e(k)/K_1$, $ce^*(k) = ce(k)/K_2$

obtain $co^*_{STFLC}(k)$ using FIS based on Table 5.3

obtain $K_{3A}(k)$ using Table 6.1

multiply $K_{3A}(k)$ by factor K_{3M} to obtain $K_3(k)$

scale $co^*_{FLC}(k)$ by factor $K_3(k)$ to obtain $co_{FLC}(k)$

obtain $S_b(k) = S_b(k-1) + co_{FLC}(k)$

write in DLL to update S_a and S_b

update $ISE(k) = ISE(k-1) + (e(k))^2$

update sampling time k = k + 1

end ***% For each set point***

end ***% End self-tuned fuzzy control***

end ***% End LabVIEW implementation***

Figure 6.1(f) Pseudocode of self-tuned FLC

Figure 6.1(f) shows pseudocode of the proposed self-tuned FLC algorithm which consists of two parts for sequential execution: first process initialization and second self-tuned FLC. Process initialization ensures that initial pH value falls within user specified lower bound (pH_{LB}) and upper bound (pH_{UB}). For initial pH setpoint as $(pH_{SP})_{initial}$, pH_{LB} and pH_{UB} are fixed 0.1 pH unit below and above $(pH_{SP})_{initial}$ respectively.

6.3 Discussion on Experimental Results

In order to evaluate performance of self-tuned fuzzy logic controller for only servo operations, pH_{SP} is changed from 6 to 7, and from 8 to 7, for various values of K_1, K_2, and K_{3M}. With sampling time of 1 second, total 501 samples are taken for servo operation. Table 6.2 summarizes ISE performance of the self-tuned fuzzy logic controller for various values of scaling factors K_1, K_2, and K_{3M}. For both positive and negative step change, control objectives are met i.e. pH response ultimately reached 7±0.2 pH settling band. Figures 6.2(a), 6.2(b), 6.2(c) to 6.7(a), 6.7(b), 6.7(c) shows variations in pH, pump speeds, and K_3 for few of those various settings of K_1, K_2, and K_{3M}. Following additional observations can be made from the obtained results.

(i) For K_{3M} = 1, pH response is slower than the same for K_{3M} = 2, 3, 4. Since variables e and ce mostly attains values within range given by e_3, e_4, e_5 and ce_3, ce_4, ce_5 respectively, magnification of K_{3A} results in faster pH response.

(ii) For K_{3M} = 1, magnitude of first overshoot or undershoot is largest as compared to the same for K_{3M} = 2, 3, 4. A magnified K_{3A} provides better neutralization of acidic process stream using basic manipulated stream.

(iii) For K_{3M} = 1, 2, 3, pH response remains within 7±0.2 pH settling band in a better way as compared to the same for K_{3M} = 4. A magnified K_{3A} sometimes drives the pH response outside the settling band.

(iv) For K_{3M} = 4, pH response indicates self-controlling property of the adaptive FLC. If pH response is showing tendency to go unbound, the self-tuned fuzzy controller subsequently adjusts the value of K_{3A} so that pH response comes within the desired settling band.

The performance of self-tuned fuzzy logic controller has been also evaluated for six cases of servo and regulatory operations in pH neutralization process as given in Table 5.5. Figures 6.8(a) to 6.8(c) shows pH response, pump speed variations, and K_3 for case $K_1 = 10$, $K_2 = 0.5$, and $K_{3M} = 3$ and 4. Table 6.3 gives experimental performance summary of self-tuned fuzzy logic controller for servo-regulatory operations. Following observation can be made from obtained results.

(i) The self-tuned fuzzy logic controller for SR operations with $K_1 = 10$, $K_2 = 0.5$, and $K_{3M} = 3$, gives total ISE as 96.0062 of which SR5 and SR1 contributions are 27.4581 and 17.6177 respectively.

(ii) The self-tuned fuzzy logic controller for SR operations with $K_1 = 10$, $K_2 = 0.5$, and $K_{3M} = 4$, gives total ISE as 79.8271 of which SR5 and SR1 contributions are 18.3122 and 17.5504 respectively. It is evident that fall time, maximum undershoot, and settling time reduced considerably for SR5 in this case.

(iii) In comparison, as mentioned in section 5.7.1, experimental validation of GA, DE, and PSO based optimized FLC for SR operations gives total ISE as 80.3776, 67.9637, and 67.9266 respectively.

(iv) Also section 5.7.2 shows that experimental validation of GA, DE, and PSO based optimized piecewise FLC for SR operations gives total ISE as 66.1221, 64.3561, and 64.8051 respectively.

From the above discussion it is clear that although performance index ISE obtained in cases of optimized FLC and optimized piecewise FLC are better than the same in case of self-tuned FLC scheme for SR operations. The optimization requires known operating conditions and it may happen that the optimized controller settings are not applicable to a new operating condition. On the other hand the self-tuned FLC scheme has unique advantage that it independent of operating conditions and works very well for unknown operating conditions too.

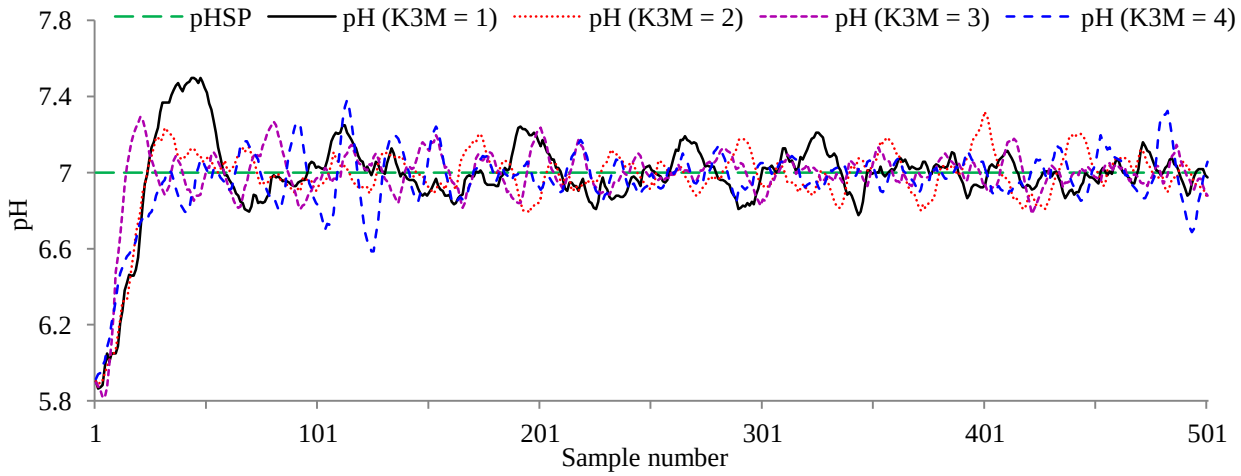


Figure 6.2(a) pH responses for $K_1 = 10$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

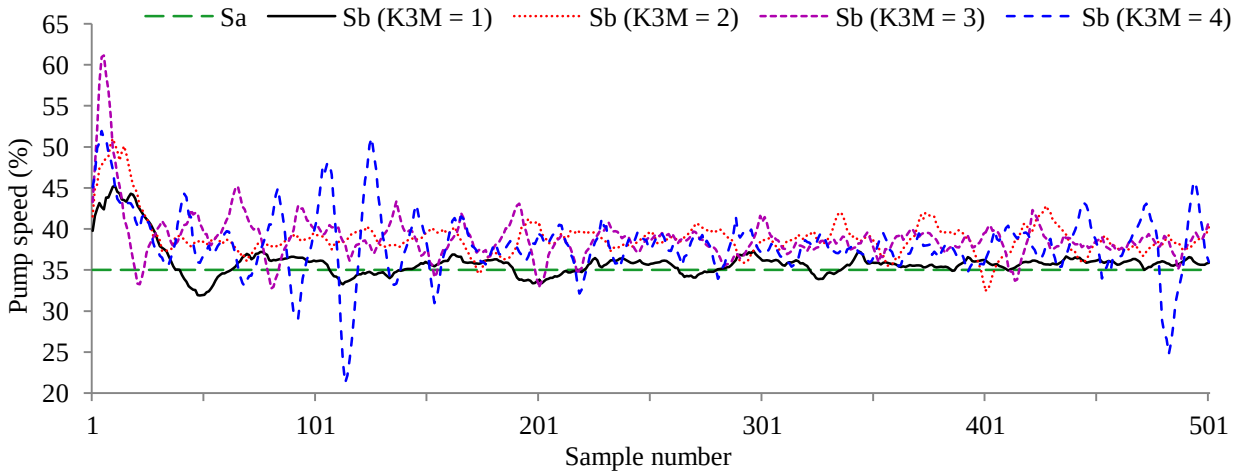


Figure 6.2(b) Pumps speed variations for $K_1 = 10$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

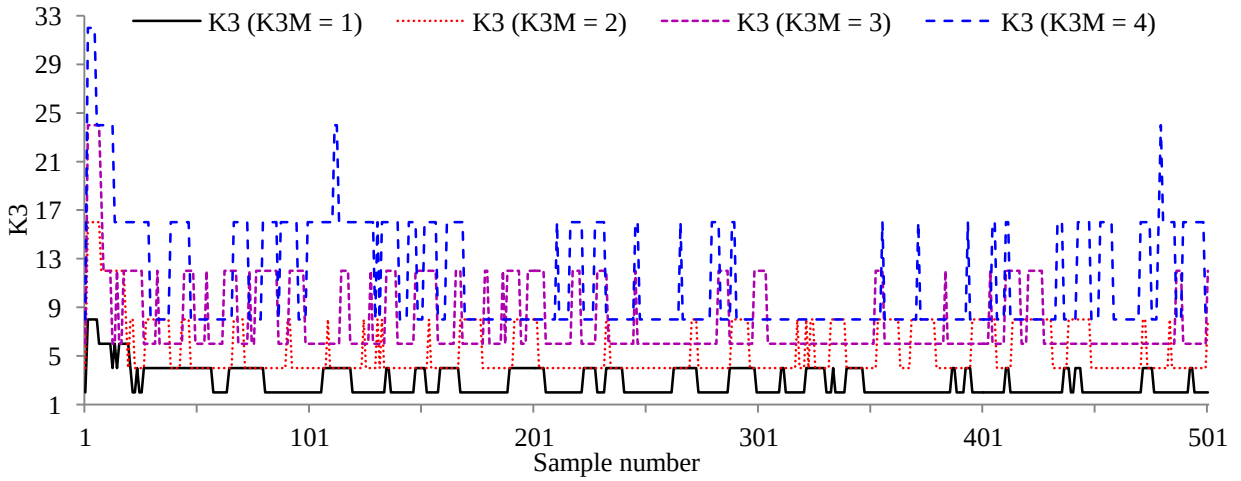


Figure 6.2(c) K_3 for $K_1 = 10$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

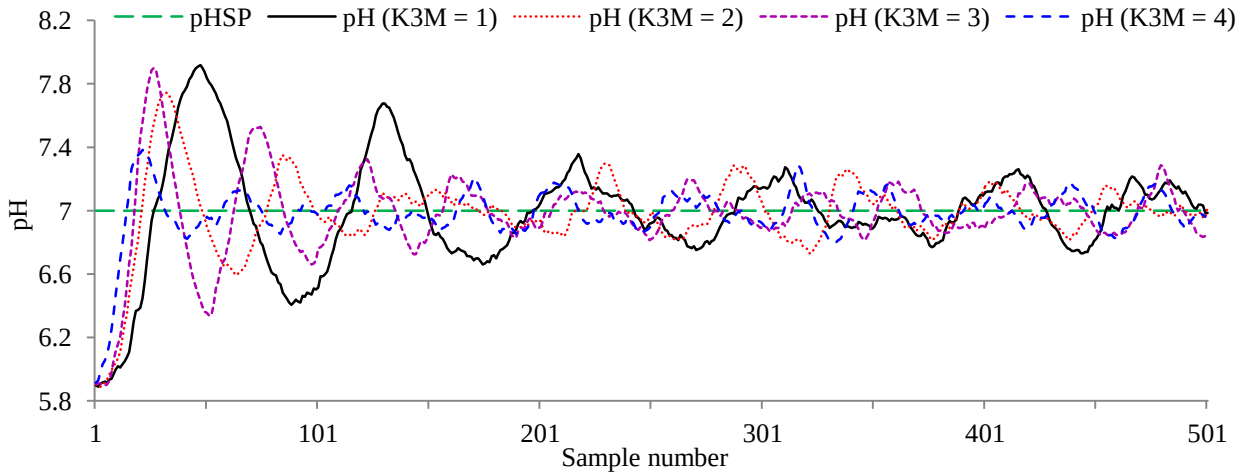


Figure 6.3(a) pH responses for $K_1 = 10$, $K_2 = 1$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

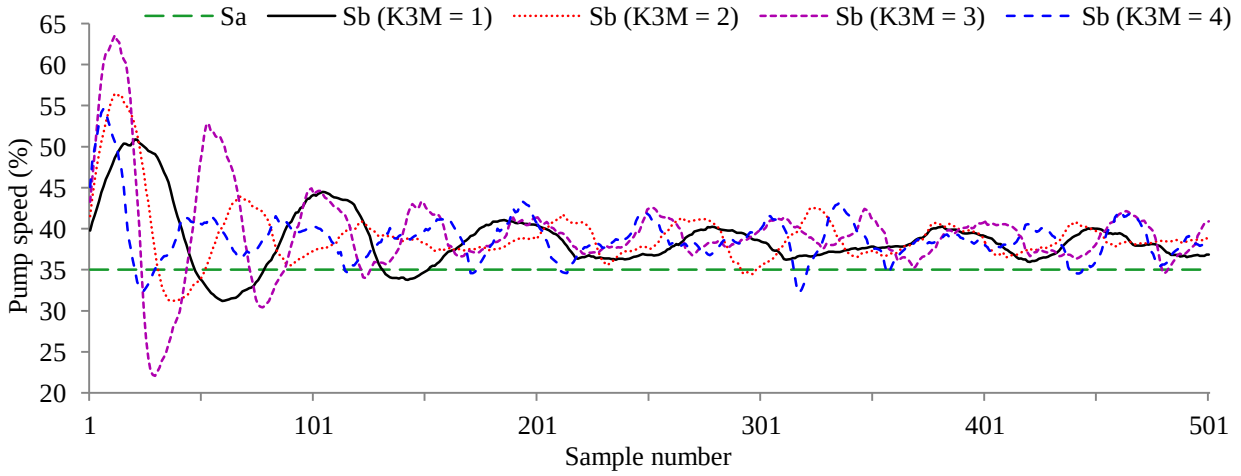


Figure 6.3(b) Pumps speed variations for $K_1 = 10$, $K_2 = 1$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

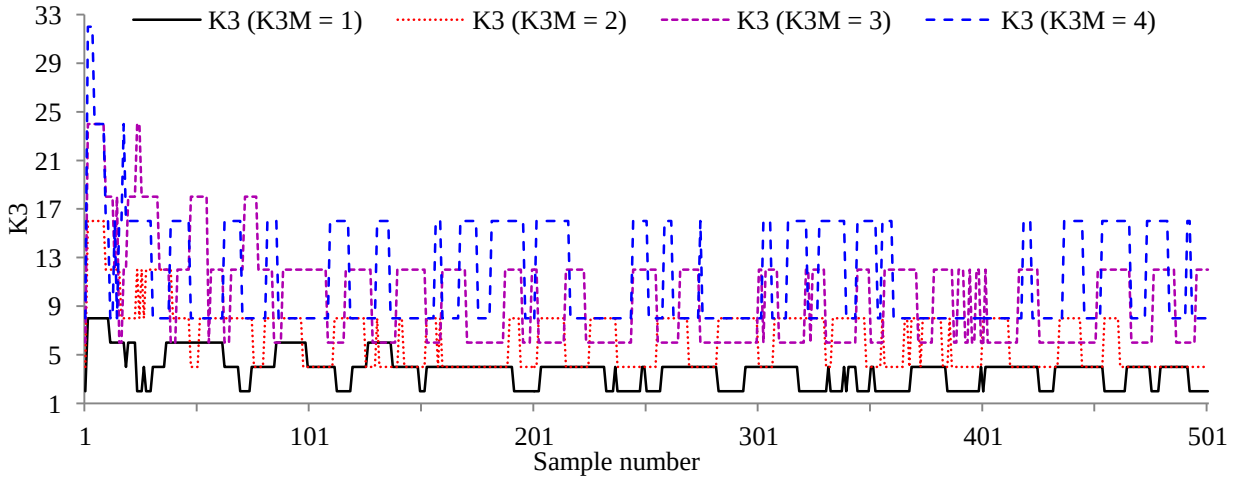


Figure 6.3(c) K_3 for $K_1 = 10$, $K_2 = 1$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

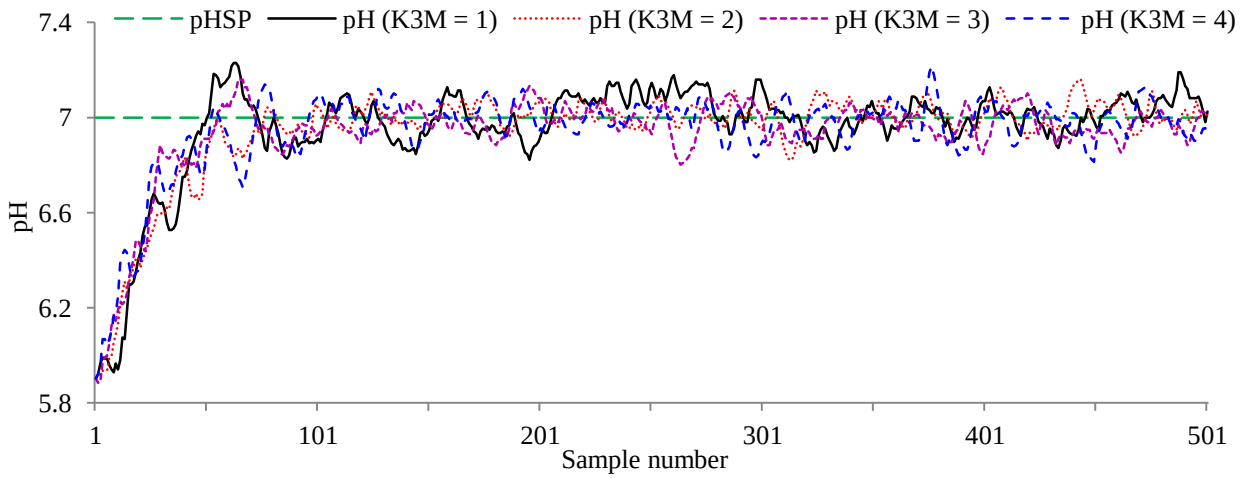


Figure 6.4(a) pH responses for $K_1 = 20$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

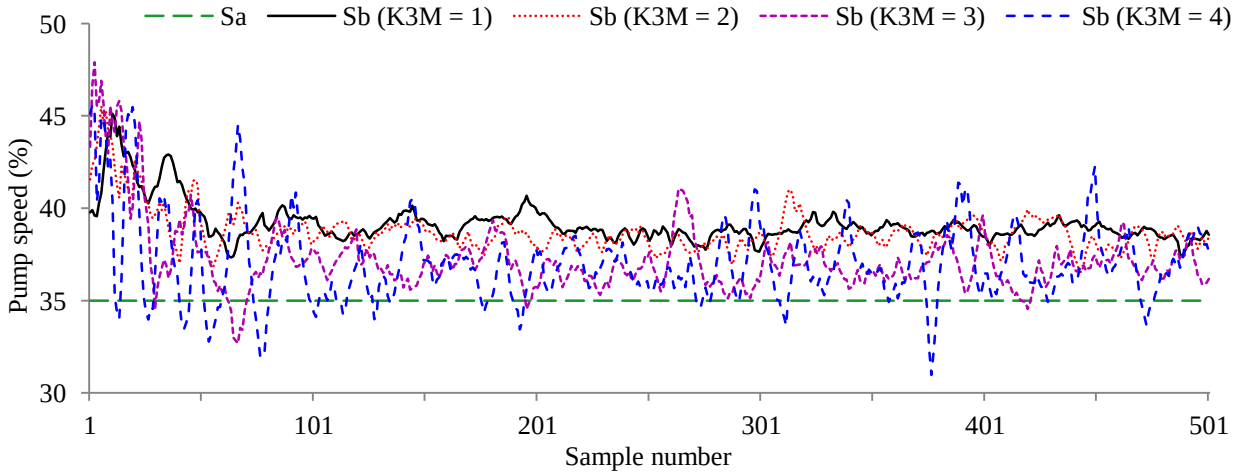


Figure 6.4(b) Pumps speed variations for $K_1 = 20$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

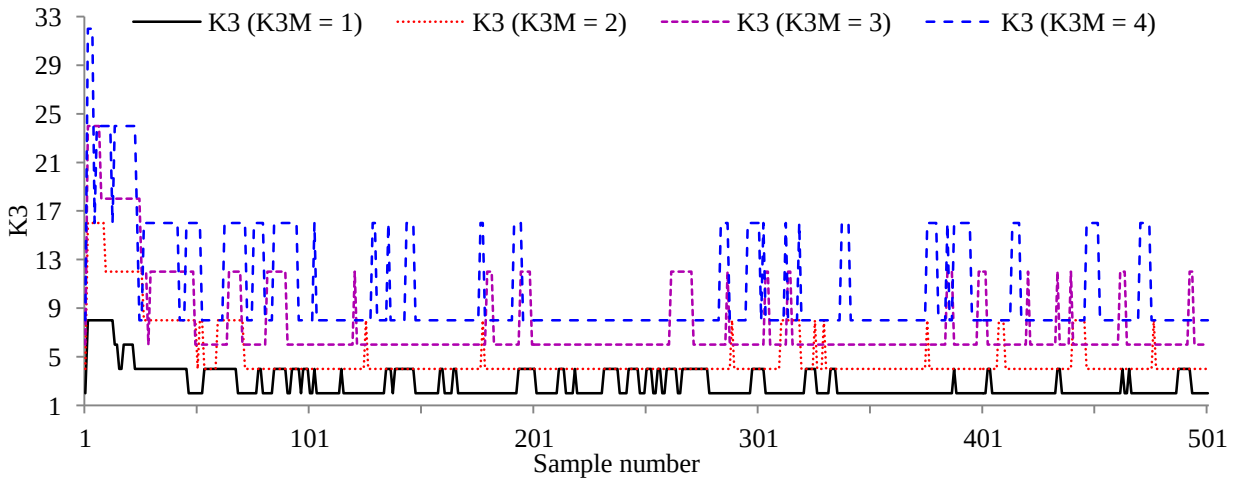


Figure 6.4(c) K_3 for $K_1 = 20$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

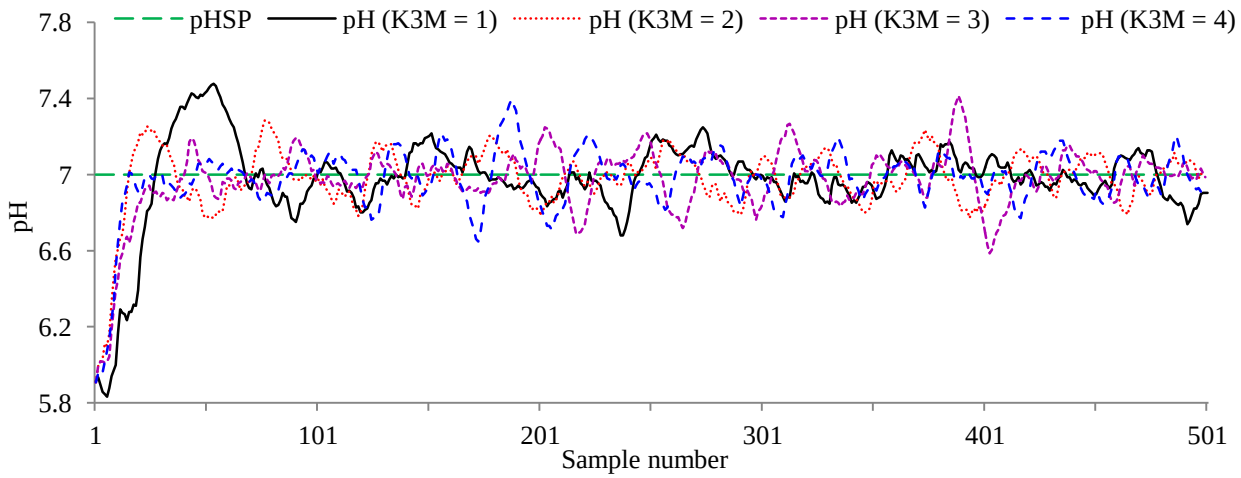


Figure 6.5(a) pH responses for $K_1 = 20$, $K_2 = 1$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

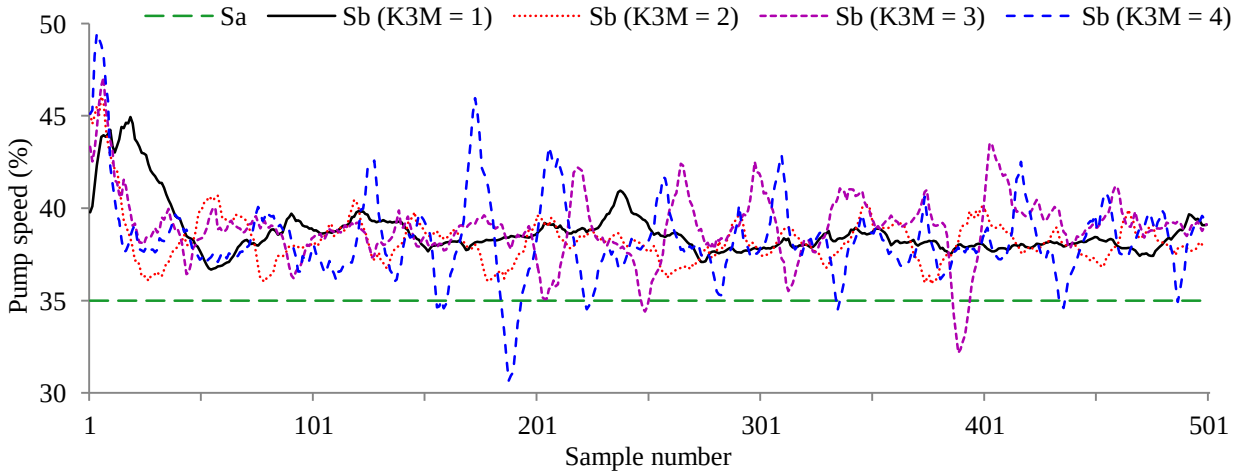


Figure 6.5(b) Pumps speed variations for $K_1 = 20$, $K_2 = 1$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

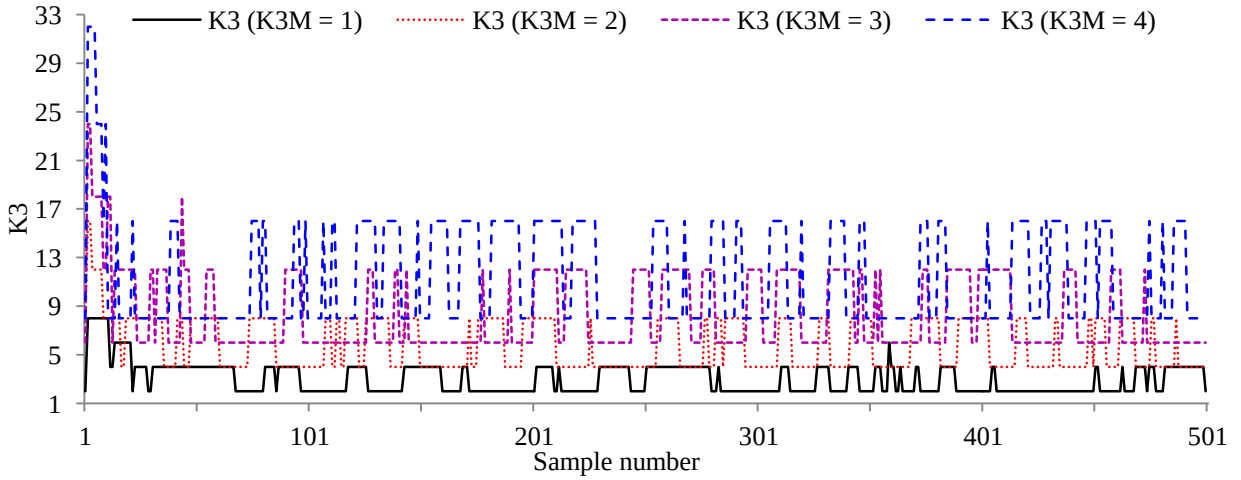


Figure 6.5(c) K_3 for $K_1 = 20$, $K_2 = 1$, $(pH_{SP})_{initial} = 6$, $(pH_{SP})_{final} = 7$

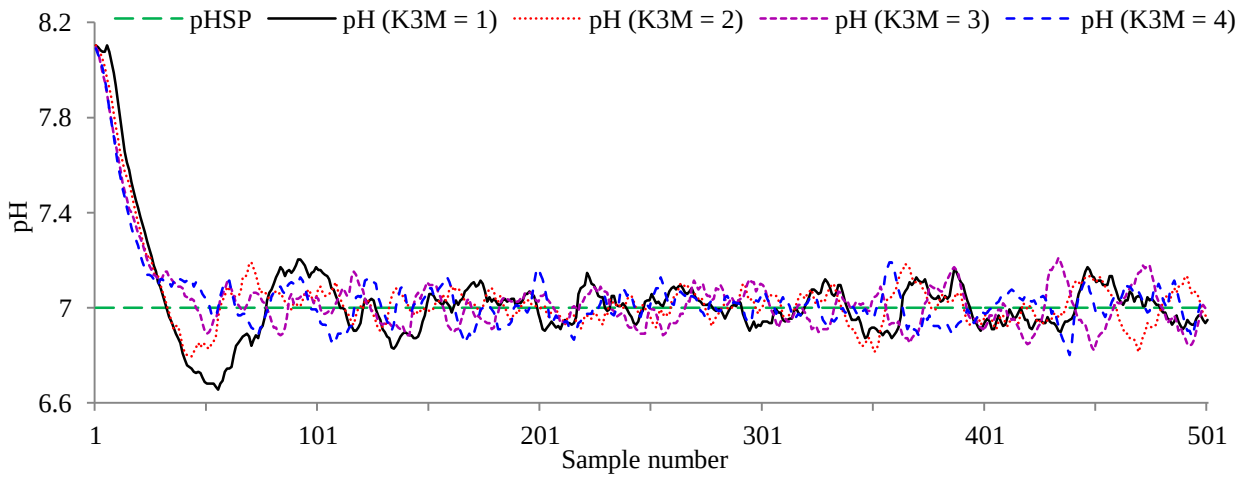


Figure 6.6(a) pH responses for $K_1 = 10$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 8$, $(pH_{SP})_{final} = 7$

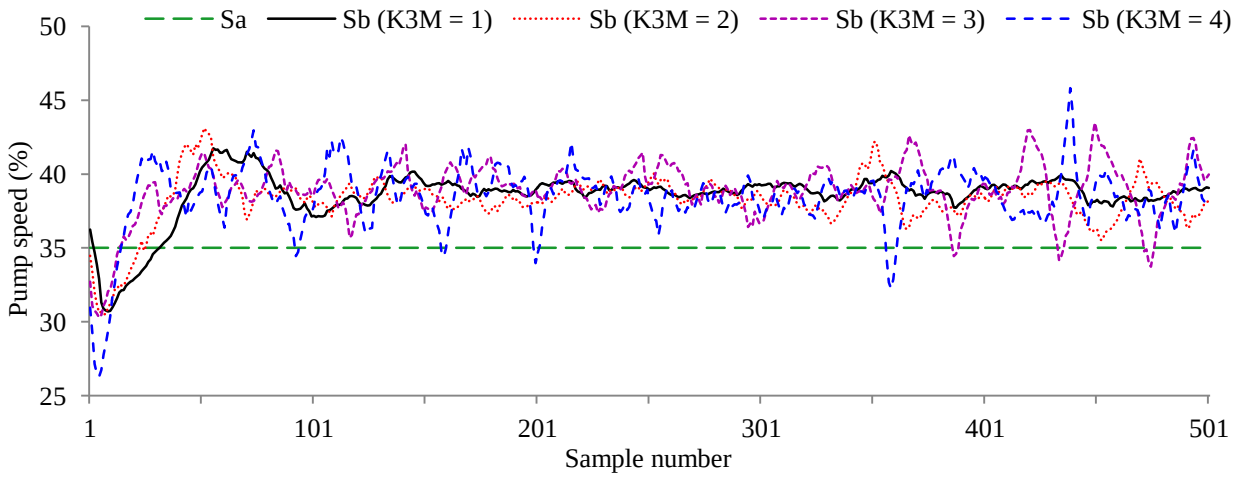


Figure 6.6(b) Pumps speed variations for $K_1 = 10$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 8$, $(pH_{SP})_{final} = 7$

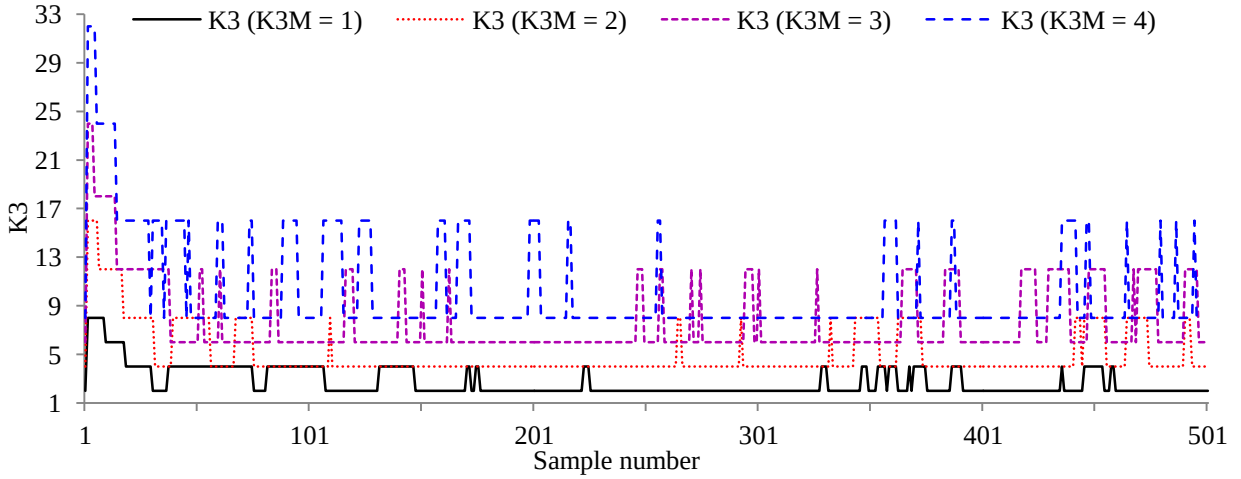


Figure 6.6(c) K_3 for $K_1 = 10$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 8$, $(pH_{SP})_{final} = 7$

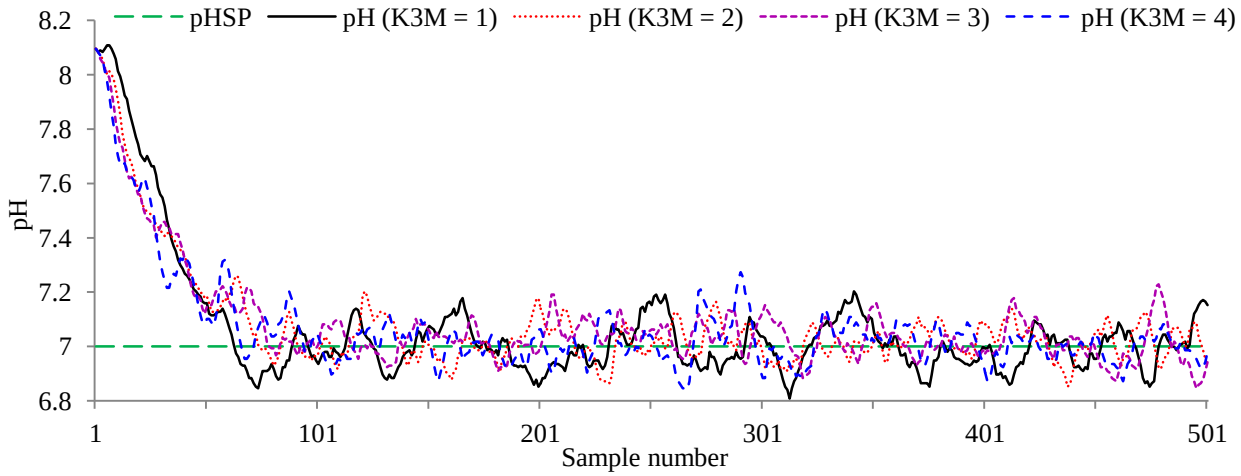


Figure 6.7(a) pH responses for $K_1 = 20$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 8$, $(pH_{SP})_{final} = 7$

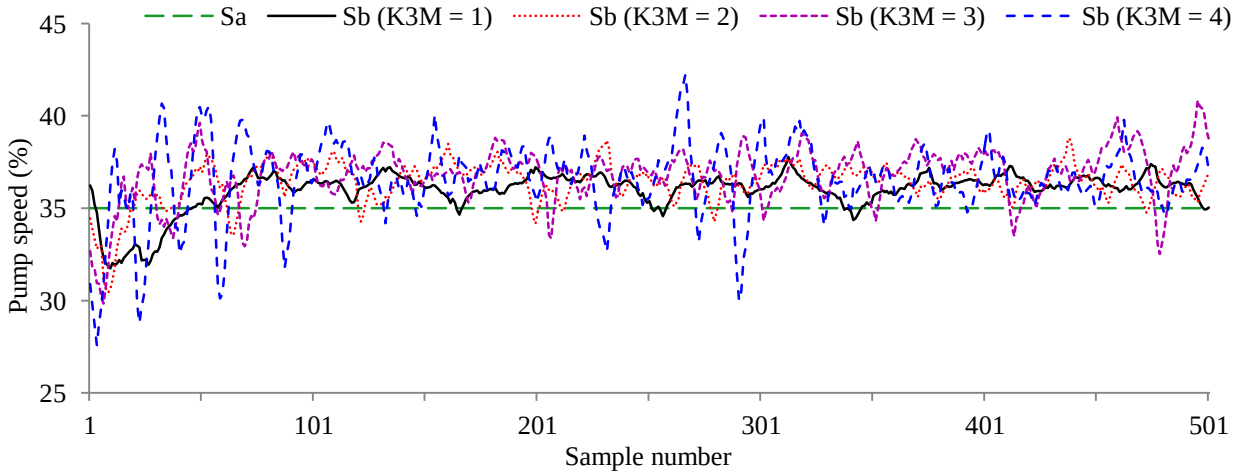


Figure 6.7(b) Pumps speed variations for $K_1 = 20$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 8$, pH_{SP} (final) = 7

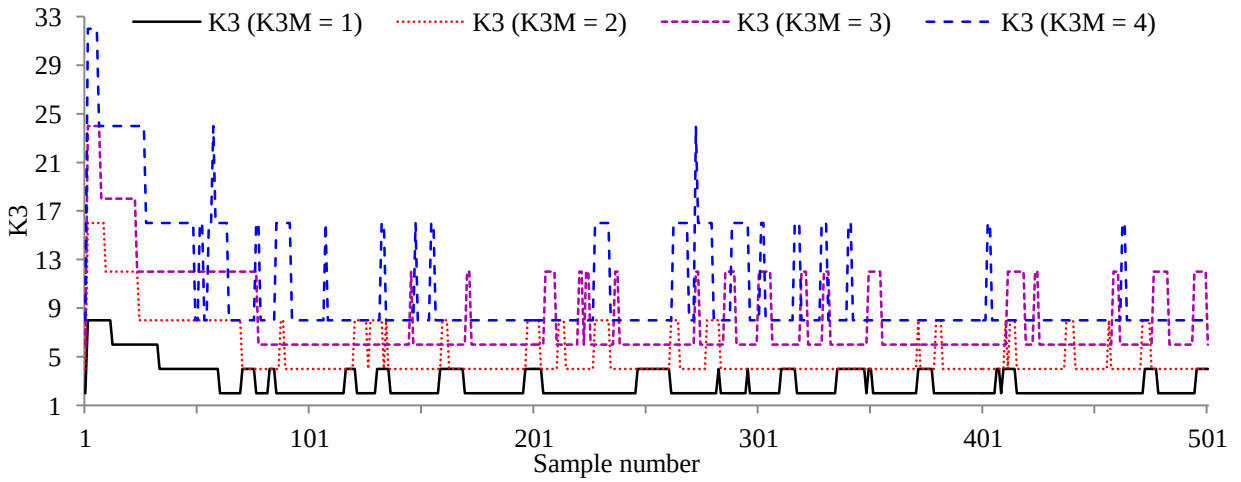


Figure 6.7(c) K_3 for $K_1 = 20$, $K_2 = 0.5$, $(pH_{SP})_{initial} = 8$, $(pH_{SP})_{final} = 7$

Table 6.2 Servo performance of self-tuned adaptive FLC at $pH_{SP} = 7$ and $\Delta pH_{SP} = \pm 1$

Sr. No.	$(pH_{SP})_{initial}$	$(pH_{SP})_{final}$	K_1	K_2	K_{3M}	ISE (501 samples)
1	6	7	10	0.5	1	23.75
2	6	7	10	0.5	2	18.91
3	6	7	10	0.5	3	14.72
4	6	7	10	0.5	4	17.42
5	6	7	10	1	1	58.67
6	6	7	10	1	2	30.98
7	6	7	10	1	3	35.07
8	6	7	10	1	4	13.79
9	6	7	20	0.5	1	24.58
10	6	7	20	0.5	2	21.74
11	6	7	20	0.5	3	18.90
12	6	7	20	0.5	4	18.22
13	6	7	20	1	1	26.94
14	6	7	20	1	2	14.03
15	6	7	20	1	3	16.55
16	6	7	20	1	4	14.88
17	6	7	30	0.5	1	34.95
18	6	7	30	0.5	2	31.65
19	6	7	30	0.5	3	28.82
20	6	7	30	0.5	4	30.14
21	6	7	30	1	1	35.99
22	6	7	30	1	2	21.63
23	6	7	30	1	3	19.15
24	6	7	30	1	4	15.40
25	8	7	10	0.5	1	20.55
26	8	7	10	0.5	2	15.34
27	8	7	10	0.5	3	13.73
28	8	7	10	0.5	4	12.69
29	8	7	20	0.5	1	29.26
30	8	7	20	0.5	2	22.86
31	8	7	20	0.5	3	21.56
32	8	7	20	0.5	4	20.12

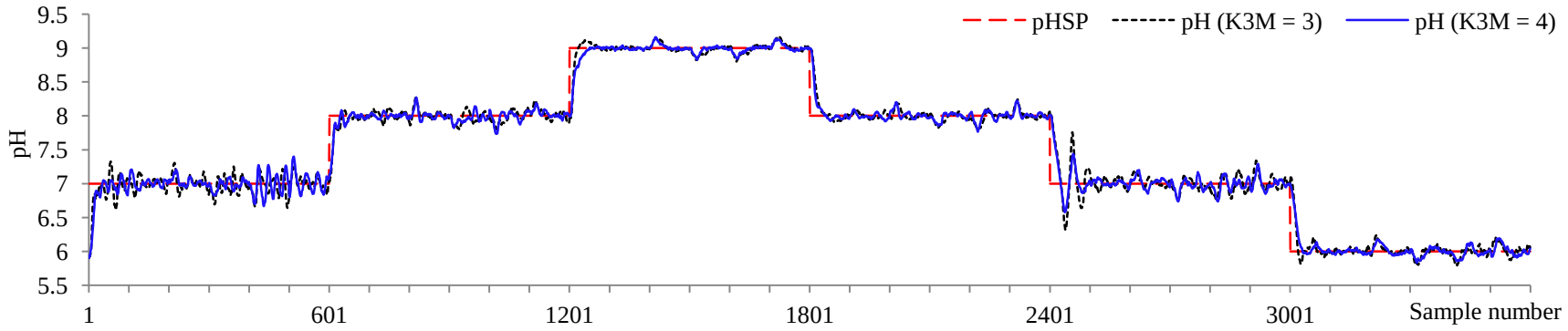


Figure 6.8(a) pH responses for K_1 = 10, K_2 = 0.5, K_{3M} = 3 and 4

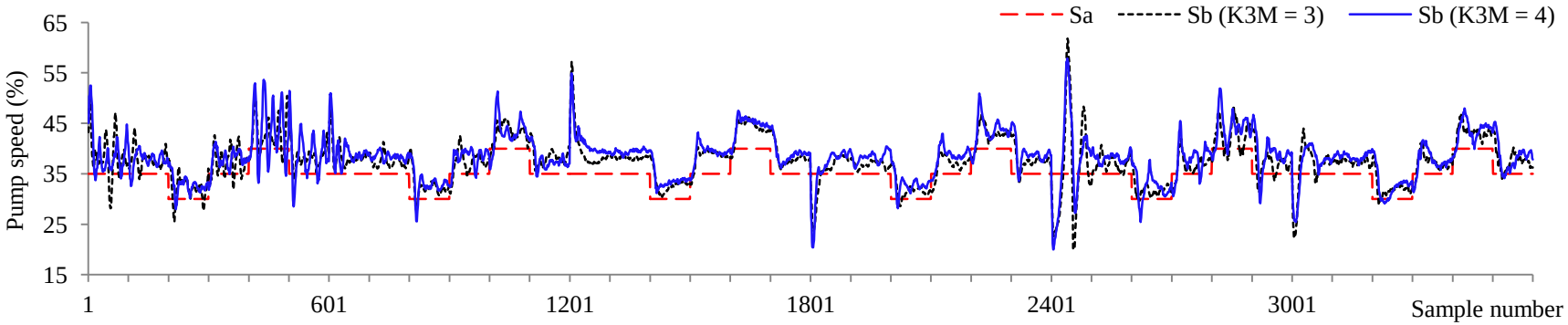


Figure 6.8(b) Pumps speed variations for K_1 = 10, K_2 = 0.5, K_{3M} = 3 and 4

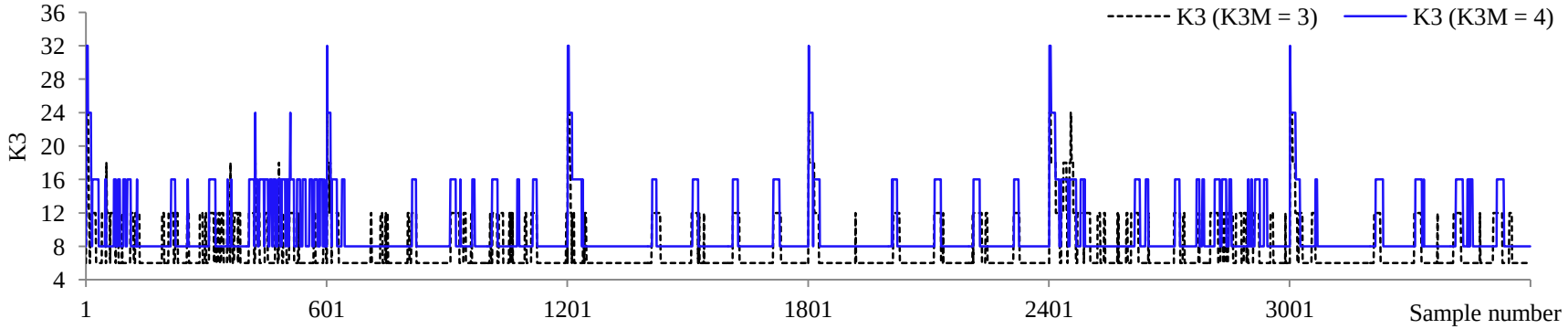


Figure 6.8(c) K_3 for K_1 = 10, K_2 = 0.5, K_{3M} = 3 and 4

Table 6.3 Experimental performance of self-tuned fuzzy logic controller for SR operations

Parameters K_1, K_2, K_{3M}	Servo operation (200 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)		Regulatory operation (100 samples)	
	$(pH_{SP})_{initial}$, $(pH_{SP})_{final}$, DV	ISE, maximum overshoot/ undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum undershoot	pH_{SP}, $(DV)_{initial}$, $(DV)_{final}$	ISE, maximum overshoot
10, 0.5, 3	6, 7, 35	11.3593, -0.3240	7, 35, 30	1.2801, -0.3050	7, 30, 35	1.6775, 0.3070	7, 35, 40	2.5100, 0.3580	7, 40, 35	0.7908, -0.2480
10, 0.5, 4		11.3040, -0.2100		0.4747, -0.2100		0.6003, 0.1730		3.0686, 0.3330		2.1028, -0.4010
10, 0.5, 3	7, 8, 35	7.6813, -0.1090	8, 35, 30	0.7509, -0.2680	8, 30, 35	0.9747, 0.2040	8, 35, 40	0.7832, 0.2040	8, 40, 35	0.7255, -0.2300
10, 0.5, 4		6.2474, -0.0840		0.5960, -0.2750		0.6036, 0.1780		0.8839, 0.2670		0.4092, -0.1860
10, 0.5, 3	8, 9, 35	8.8530, -0.1170	9, 35, 30	0.4648, -0.1490	9, 30, 35	0.5276, 0.1770	9, 35, 40	0.5824, 0.1960	9, 40, 35	0.5040, -0.1680
10, 0.5, 4		9.5923, -0.0400		0.3008, -0.1610		0.3171, 0.1700		0.3469, 0.1570		0.3274, -0.1420
10, 0.5, 3	9, 8, 35	11.6065, 0.0890	8, 35, 30	0.4825, -0.1860	8, 30, 35	0.5388, 0.1780	8, 35, 40	0.7198, 0.1910	8, 40, 35	0.5510, -0.2430
10, 0.5, 4		9.7252, 0.0820		0.4366, -0.1980		0.3592, 0.1460		0.6448, 0.2350		0.4676, -0.2240
10, 0.5, 3	8, 7, 35	22.9464, 0.6770	7, 35, 30	0.5639, -0.1840	7, 30, 35	0.7934, 0.2240	7, 35, 40	1.8408, 0.2370	7, 40, 35	1.3136, -0.3370
10, 0.5, 4		14.7684, 0.4220		0.6536, -0.2030		0.8880, 0.2620		1.0729, 0.2620		0.9293, -0.2930
10, 0.5, 3	7, 6, 35	12.4186, 0.1870	6, 35, 30	0.6763, -0.2460	6, 30, 35	0.6551, 0.2000	6, 35, 40	0.6065, 0.2060	6, 40, 35	0.8280, -0.2020
10, 0.5, 4		10.3599, 0.0530		0.5604, -0.1700		0.4816, 0.1680		0.6032, 0.1430		0.7016, -0.1890

6.4 Concluding Remarks

In this chapter Mamdani FIS based self-tuned FLC scheme has been implemented on Armfield pH neutralization process. The self-tuned FLC has input scaling factors K_1 and K_2, and output scaling factor K_3. Keeping K_1 and K_2 constant, self-tuning mechanism actually determines the value of K_3 which consist of two components: K_{3A} which has discrete values 2, 4, 6, and 8 based on present error and change in error values, and K_{3M} which is magnifier that can take integer values from 1 to 4. First performance of self-tuned fuzzy logic controller has been evaluated for servo operation by introducing step change in pH setpoint from 6 to 7, and from 8 to 7 at acid flowrate at 35%. It is noted that in almost all test results, pH response finally settles within 7±0.2 pH band. In some cases, when pH response occasionally overshoots and undershoots the above band, controller adjusts its output universe of discourse and again brings the pH response back within the desired band. Also self-tuned fuzzy controller gives better performance in terms of ISE for magnified K_3. In addition, self-tuned fuzzy logic controller is used for servo and regulatory (SR) operations in pH neutralization process, and its performance index ISE comes as 96.0062 for K_{3M} = 3, and 79.8271 for K_{3M} = 4. In comparison, experimental validation of GA, DE, and PSO based optimized FLC for SR operations gives total ISE as 80.3776, 67.9637, and 67.9266 respectively. Also experimental validation of GA, DE, and PSO based optimized piecewise FLC for SR operations gives total ISE as 66.1221, 64.3561, and 64.8051 respectively. Thus ISE of self-tuned FLC for SR operations is greater than the same for optimized FLC and optimized piecewise FLC. However, self-tuned FLC has reduced design complexity and execution time as compared to optimized FLC, and although its performance is not better than optimized FLC but acceptable.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

In this thesis, Armfield® Process Control Teaching System (PCT40) along with Process Vessel Accessory (PCT41) and pH Sensor Accessory (PCT42) has been used for testing performance of modeling and control strategies developed for strong acid-strong base i.e. Hydrochloric acid (HCl)-Sodium Hydroxide (NaOH) neutralization process. LabVIEW® has been used for online interfacing of Armfield pH neutralization system.

From the research work carried out and presented in the thesis, the following conclusions are drawn:

(i) For development of dynamic models for Armfield pH neutralization system, two approaches have been used: (i) first principles technique proposed by McAvoy et al. (1972), and (ii) feedforward Artificial Neural Network (ANN). The specifications of Armfield pH neutralization system are comparable with those of McAvoy (1972) experimental set up except that Armfield pH neutralization system involves reaction of strong acid-strong base with reduced concentration whereas McAvoy (1972) experimental set up involves weak acid-strong base reaction with relatively higher concentration. The first principles method for Armfield pH neutralization system uses material balances on sodium and chloride ions, water equilibrium relationships, and electroneutrality equations. Using experimental results of various step tests conducted at pH 7, it has been shown that first principles based dynamic pH model is not able to account for mixing dynamics of strong acid-strong based streams in dynamic pH range of 4 to 10.

Dynamic feedforward ANN structure using Tapped Delay Line (TDL) to model Armfield pH neutralization system uses total 32740 data samples covering pH range from 4 to 10 for training, validation, and testing of the network. For various number of delayed input-output samples of the nonlinear pH neutralization process, offline performance of training functions namely Gradient-Descent method with constant learning rate (GD), Gradient-Descent method with constant learning rate and Momentum (GDM), Gradient-Descent method with Adaptive learning rate (GDA), Gradient-Descent with Adaptive learning rate and Momentum (GDAM), and Levenberg-

Marquardt algorithm (LM) are compared based on Mean Squared Error (MSE) values for training, validation and testing data sets using MATLAB®. It is found that LM gives best performance values for all test cases. Further it is found that for three delayed input-output samples, dynamic ANN model using LM training function gives reasonably acceptable performance values.

(ii) The feedback control of Armfield pH neutralization process for servo and regulatory operations has been done using optimized conventional Proportional-Integral-Derivative (PID), and optimized Mamdani Fuzzy Inference System (FIS) based Fuzzy Logic Control (FLC). The global optimization techniques namely Genetic algorithm (GA), Differential Evolution (DE), and Particle Swarm Optimization (PSO) are used to optimize pH controller parameters i.e. proportional, integral and derivative gains K_P, K_I and K_D respectively for conventional PID, and scaling factors K_1, K_2 and K_3 for error, change in error and change in output respectively for FLC. Offline optimization uses dynamic ANN model developed, and online optimization uses Armfield neutralization process. Servo and regulatory operations incorporate dynamic pH variations from 6 to 9, and disturbance variable variations from 30% to 40% in acidic stream flowrates.

The offline optimized conventional PID controller performance in terms of Integral Square of Error (ISE) does not match with the experimentally obtained result, and the experimental responses of PID controller shows that pH control is very difficult at pH setpoint of 7. Based on experimental validation results it is concluded that PID controller is not suitable for pH control around highly nonlinear region of pH equals 7. On the other hand, the offline optimized fuzzy logic controller performance using GA, DE and PSO in terms of ISE is near to the experimentally obtained result. The experimental validation shows that fuzzy logic controller results in much less ISE than conventional PID controller. Based on final population convergence result for offline optimization with moderate number of generations it is concluded that DE is best followed successively by PSO and GA.

To address nonlinearity of pH neutralization process, fuzzy logic controllers are designed for six different regions of dynamic pH range from 6 to 9 in piecewise manner. The offline optimized piecewise fuzzy logic controllers results in slightly less ISE than the overall optimized fuzzy logic controller which is experimentally validated too. Based on experimental responses of fuzzy

logic controller it is concluded that piecewise optimization using GA, DE and PSO results in improved performance.

Finally, GA, DE and PSO based online optimization of piecewise fuzzy logic controller are carried for pH setpoint changes from 6 to 7, and 8 to 7, with acidic flow rate variations from 30% to 40% at pH setpoint of 7. The ISE performance values confirm that all three global optimization techniques gives approximately similar results. Based on ease of implementation for online optimization with small number of generations it is concluded that DE has most simplistic algorithm followed successively by PSO and DE.

(iii) To overcome the problem of random variations in process operating condition, self-tuned FLC scheme has been implemented on Armfield pH neutralization system. The self-tuned FLC scheme takes different values for output scaling factor assigned by designer based on present error and change in error. In particular, when the magnitude of error or change in error are greater than 1 pH, output scaling factor is large, say 8, and when the magnitude of error and change in error are less than 0.1 pH, output scaling factor is small, say 2. In this way, for 49 different conditions, assignment for output scaling factor has been done. For various values of input scaling factors and output scaling factor magnifier, self-tuned fuzzy logic controller performance has been first evaluated for step change in pH setpoint from 6 to 7, and from 8 to 7. In all test results, pH response finally settles within 7±0.2 pH band. Based on experimental validation results it is concluded self-tuned fuzzy controller gives satisfactory and comparable performance compared to offline GA, DE and PSO based optimized fuzzy controller.

7.2 Summary of Contributions

The main contributions of the author in this thesis work are briefly listed below:

(i) Performed experimental validation of dynamic model of Armfield pH neutralization system based on first principles technique proposed by McAvoy et al. (1972).

(ii) Developed dynamic feedforward Artificial Neural Network (ANN) architecture using Tapped Delay Line (TDL) approach, and demonstrated performance comparisons of different training functions.

(iii) Proposed tuning of pH controller parameters based on Proportional-Integral-Derivative (PID) and Fuzzy Logic Control (FLC) schemes using global optimization techniques namely Genetic Algorithm (GA), Differential Evolution (DE), and Particle Swarm Optimization (PSO).

(iv) Performed comparative study of GA, DE, and PSO based PID controller performances through simulations on ANN model and experimental validations on Armfield system for servo and regulatory operations in dynamic pH region of 6 to 9.

(v) Performed comparative study of GA, DE, and PSO based fuzzy logic controller performances through simulations and experimental validations for servo and regulatory operations in dynamic pH region of 6 to 9.

(vi) Performed comparative study of piecewise tuning of fuzzy logic controller using offline GA, DE, and PSO by dividing dynamic pH region of 6 to 9 into smaller segments, for servo and regulatory operations.

(vii) Proposed self-tuned FLC scheme, and a comparative performance study of optimized FLC and self-tuned FLC schemes through experimental validations for servo and regulatory operations in dynamic pH region of 6 to 9 is done.

7.3 Recommendations for Future Work

The objectives laid down in Chapter 1 for this thesis have been successfully met, there exists scope for further development. Some of the suggestions for future work are outlined below:

(i) Development of dynamic model based on first principles to account for mixing in strong acid-strong base neutralization process.

(ii) Online system identification methods may be explored for development of dynamic model of strong acid-strong base neutralization process.

(iii) Hybrid evolutionary optimization methods can be developed for online tuning of the parameters of fuzzy logic controller to achieve faster convergence and better performance.

(iv) Multi-objective optimization methods can be applied for online tuning of the parameters of fuzzy logic controller.

(v) Self-organizing fuzzy controllers can be developed that either modify the existing set of fuzzy rules, or they start with no fuzzy rules at all and learn their control strategy online.

(vi) Various control strategies and optimization methods proposed may be suitably amended to investigate highly nonlinear and complex system consisting of multiple streams of strong and weak acids and bases.

LIST OF PUBLICATIONS

1. Singh P.K., Bhanot S., Mohanta H.K., 2013, 'Optimized Adaptive Neuro-Fuzzy Inference System for pH Control', in *Advanced Electronic Systems (ICAES), 2013 International Conference on, Pilani*, pp. 1-5.

2. Singh P.K., Bhanot S., Mohanta H.K., 2013, 'Optimized and Self-Organized Fuzzy Logic Controller for pH Neutralization Process', *International Journal of Intelligent Systems and Applications*, vol. 5, no. 12, pp. 99-112.

3. Singh P.K., Bhanot S., Mohanta H.K., 2014, 'Differential Evolution based Optimal Fuzzy Logic Control of pH Neutralization Process', in *Convergence of Technology (I2CT), 2014 International Conference for, Pune*, pp. 1-5.

4. Singh P.K., Bhanot S., Mohanta H.K., 2014, 'Genetic Optimization based Adaptive Fuzzy Logic Control of a pH Neutralization Process', *International Journal of Control and Automation*, vol. 7, no. 11, pp. 233-248.

5. Singh P.K., Bhanot S., Mohanta H.K., 2015, 'Particle Swarm Optimization based Fuzzy Logic Control of pH Neutralization Process', *International Journal of Applied Engineering Research*, vol. 10, no. 63, special issue, pp. 211-215.

6. Singh P.K., Bhanot S., Mohanta H.K., Bansal V., 2014, 'Self-Tuned Fuzzy Logic Control of a pH Neutralization Process', in *Automation & Computing (ICAC 2015), 21st International Conference on, Glasgow*, pp. 155-160.

REFERENCES

Adroer M., Alsina A., Aumatell J., Poch M., 1999, 'Wastewater Neutralization Control based in Fuzzy Logic: Experimental Results', *Ind. Eng. Chem. Res.*, 38, pp. 2709-2719.

Agarwal M., 1997, 'A systematic classification of neural-network-based control', *Control Systems, IEEE*, 17, pp. 75-93.

Ahmed S., Chandra U., Rathi R.K., 2010, 'Waste Water Treatment Technologies Commonly Practiced in Major Steel Industries of India', in *16th Annual International Sustainable Development Research Conference 2010, Hong Kong*, pp. 537-552.

Åkesson B.M., Toivonen H.T., Waller J.B., Nyström R.H., 2005, 'Neural Network Approximation of a Nonlinear Model Predictive Controller applied to a pH Neutralization Process', *Computers & Chemical Engineering*, 29, pp. 323-335.

Alpbaz M., Hapoğlu H., Özkan G., Altuntas S., 2006, 'Application of Self-Tuning PID Control to a Reactor of Limestone Slurry Titrated with Sulfuric Acid', *Chemical Engineering Journal*, 116, pp. 19-24.

Aoyama A., Doyle III F.J., Venkatasubramanian V., 1995, 'A Fuzzy Neural-Network Approach for Nonlinear Process Control', *Engineering Applications of Artificial Intelligence*, 8, pp. 483-498.

Aras Ö., Bayramoglu M., Hasiloglu A.S., 2011, 'Optimization of Scaled Parameters and Setting Minimum Rule Base for a Fuzzy Controller in a Lab-Scale pH Process', *Ind. Eng. Chem. Res.*, 50, pp. 3335–3344.

Arefi M.M., Montazeri A., Poshtan J., Jahed-Motlagh M.R., 2006, 'Nonlinear Model Predictive Control of Chemical Processes with a Wiener Identification Approach', in *Industrial Technology, 2006. ICIT 2006. IEEE International Conference on, Mumbai*, pp. 1735-1740.

Armfield Limited, 2005, 'Instruction Manual PCT40', Issue 4.

Armfield Limited, 2006a, 'Instruction Manual PCT41', Issue 3.

Armfield Limited, 2006b, 'Instruction Manual PCT42', Issue 2.

Asbjørnsen O.A., 1972, 'Reaction Invariants in the Control of Continuous Chemical Reactors', *Chemical Engineering Science*, 27, pp. 709-717.

Åström K.J., Borisson U., Ljung J., Wittenmark B., 1977, 'Theory and Applications of Self-Tuning Regulators', *Automatica*, 13, pp. 457-476.

Åström K.J., Wittenmark B., 2008, *Adaptive Control*, Dover Publications, Inc., New York.

Babuska R., Oosterhoff J., Oudshoorn A., Bruijn P.M., 2002, 'Fuzzy Self-Tuning PI Control of pH in Fermentation', *Engineering Applications of Artificial Intelligence*, 15, pp. 3-15.

Babuška R., Verbruggen H., 2003, 'Neuro-Fuzzy Methods for Nonlinear System Identification', *Annual Reviews in Control*, 27, pp. 73-85.

Bates R.G., 1965, *Determination of pH: Theory and Practice*, John Wiley & Sons, Inc., USA.

Beale M.H., Hagan M.T., Demuth H.T., 2015, *Neural Network Toolbox User's Guide*, The MathWorks Inc., USA.

Behera L., Anand K.K., 1999, 'Guaranteed Tracking and Regulatory Performance of Nonlinear Dynamic Systems using Fuzzy Neural Networks', *Control Theory and Applications, IEE Proceedings*, 146, pp. 484-491.

Bennett S., 1996, 'A Brief History of Automatic Control', *Control Systems, IEEE*, 16, pp. 17-25.

Bharathi N., Shanmugam J., Rangaswamy T.R., 2006, 'Control of Neutralization Process using Neuro and Fuzzy Controller', in *Power Electronics, 2006. IICPE 2006. India International Conference on, Chennai*, pp. 177-182.

Bhat N., McAvoy T.J., 1989, 'Use of Neural Nets for Dynamic Modeling and Control of Chemical Process Systems' in *American Control Conference, 1989, Pittsburgh*, pp. 1342-1348.

Brown M., Lightbody G., Irwin G., 1997, 'Nonlinear Internal Model Control using Local Model Networks', *Control Theory and Applications, IEE Proceedings*, 144, pp. 505-514.

Central Pollution Control Board, 2007, *The Environment (Protection) Rules, 1986, Schedule I: Standards for Emission or Discharge of Environmental Pollutants from Various Industries*, Available: (http://www.cpcb.nic.in/Industry_Specific_Standards.php)

Chan H.-C., Yu C.-C., 1995, 'Autotuning of Gain-Scheduled pH Control: An Experimental Study', *Ind. Eng. Chem. Res.*, 34, pp. 1718-1729.

Chen C.-L., Chang F.-Y., 1996, 'Design and Analysis of Neural/Fuzzy Variable Structural PID Control Systems', *Control Theory and Applications, IEE Proceedings*, 143, pp. 200-208.

Chen C.-L., Chang M.-H., 1998, 'Optimal Design of Fuzzy Sliding-Mode Control: A Comparative Study', *Fuzzy Sets and Systems*, 93, pp. 37-48.

Chen X.P., Ke D., Wei L., Duxiaoning, 1996, 'Nonlinear Neural Network Internal Model Control with Fuzzy Adjustable Parameter', in *Industrial Technology, 1996. (ICIT '96), Proceedings of The IEEE International Conference on, Shanghai*, pp. 834-838.

Cheng Y., Himmelblau D.M., 1995, 'Identification of nonlinear processes with dead time by recurrent neural networks', in *American Control Conference, Proceedings of the 1995 (Volume: 4), Seattle*, pp. 2677-2681 (Vol. 4).

Cho K.-H., Yeo Y.-K., Kim J.-S., Koh S.-t., 1999, 'Fuzzy Model Predictive Control of Nonlinear pH Process', *Korean Journal of Chemical Engineering*, 16, pp. 208-214.

Choi J.-Y., Rhinehart R.R., 1987, 'Internal Adaptive-Model Control of Wastewater pH', in *American Control Conference 1987, Minneapolis*, pp. 2084-2089.

Chou C.-H., 2006, 'Genetic Algorithm-based Optimal Fuzzy Controller Design in the Linguistic Space', *IEEE Transactions on Fuzzy Systems*, 14, pp. 372-385.

Cohen W., Friedmann P., 1974, '*Adaptive Feedforward-Feedback Control of the Concentration of a Selected Ion of a Solution*', Available: (http://www.google.co.in/patents/US3791793).

Corriou J.-P., 2008, *Process Control: Theory and Applications*, Springer (India) Private Limited, New Delhi.

Das S., Suganthan P.N., 2011, 'Differential Evolution: A Survey of the State-of-the-Art', *IEEE Transactions on Evolutionary Computation*, 15, pp. 4-31.

Díaz-Mendoza R., Budman H., 2010, 'Structured Singular Valued based Robust Nonlinear model Predictive Controller using Volterra Series Models', *Journal of Process Control*, 20, pp. 653-663.

Draeger A., Ranke H., Engell S., 1994, 'Neural Network Based Model Predictive Control of a Continuous Neutralization Reactor', in *Control Applications 1994, Proceedings of the Third IEEE Conference on (Volume 1), Glasgow*, pp. 427-432.

Edgar C.R., Postlethwaite B.E., 2000, 'MIMO Fuzzy Internal Model Control', *Automatica*, 34, pp. 867-877.

Eikens B., Karim M.N., Saucedo V., Morris A.J., 1995, 'Waste Water Neutralization using a Fuzzy Neural Network Controller', in *American Control Conference, Proceedings of the 1995 (Volume: 4), Seattle*, pp. 2662-2666 (Vol. 4).

Elarafi M.G.M.K., Hisham S.K., 2008, 'Modeling and Control of pH Neutralization using Neural Network Predictive Controller', in *Control, Automation and Systems, 2008. ICCAS 2008. International Conference on, Seoul*, pp. 1196-1199.

Food Safety and Standards Authority of India, 2011, *The Food Safety and Standards Regulations 2011*, Available: (http://www.fssai.gov.in/GazettedNotifications.aspx#regulations2011).

Fruzzetti K.P., Palazoğlu A., McDonal K.A., 1997, 'Nonlinear Model Predictive Control using Hammerstein Models', *Journal of Process Control*, 7, pp. 31-41.

Fuente M.J., Robles C., Casado O., Syafiie S., Tadeo F., 2006, 'Fuzzy Control of a Neutralization Process', *Engineering Applications of Artificial Intelligence*, 19, pp. 905-914.

Fuente M.J., Robles C., Casado O., Tadeo F., 2002, 'Fuzzy Control of a Neutralization Process', in *Control Applications, 2002. Proceedings of the 2002 International Conference on (Volume: 2), Glasgow*, pp. 1032-1037 (Vol. 2).

Garcia C.E., Morari M., 1982, 'Internal Model Control. 1. A Unifying Review and Some New Results', *Ind. Eng. Chem. Process Des. Develop.*, 21, pp. 308-323.

Garrido R., Adroer M., Poch M., 1997, 'Wastewater Neutralization Control based in Fuzzy Logic: Simulation Results', *Ind. Eng. Chem. Res.*, 36, pp. 1665-1674.

Goldberg D.E., 1989, *Genetic Algorithms in Search, Optimization, & Macine Learning*, Dorling Kindersley (India) Pvt. Ltd., India.

Gómez J.C., Jutan A., Baeyens E., 2004, 'Wiener Model Identification and Predictive Control of a pH Neutralisation Process', *Control Theory and Applications, IEE Proceedings*, 151, pp. 329-338.

Gustafsson T.K., 1985, 'An Experimental Study of a Class of Algorithms for Adaptive pH Control', *Chemical Engineering Science*, 40, pp. 827-837.

Gustafsson T.K., Waller K.V., 1983, 'Dynamic Modeling and Reaction Invariant Control of pH', *Chemical Engineering Science*, 38, pp. 389-398.

Hadjiski M., Boshnakov K., Galibova M., 2002, 'Neural Networks based Control of pH Neutralization Plant', in *Intelligent Systems, 2002. Proceedings. 2002 First International IEEE Symposium (Volume: 2)*, pp. 7-12 (Vol. 2).

Hagan M.T., Demuth H.B., 1999, 'Neural Networks for Control', in *American Control Conference, 1999. Proceedings of the 1999 (Volume: 3), San Diego*, pp. 1642-1656 (Vol. 3).

Hagan M.T., Menhaj M.B., 1994, 'Training Feedforward Network with Marquardt Algorithm', *IEEE Transactions on Neural Networks*, 5, pp. 989-993.

Hagan M.T., Demuth H.B., Beale M.H., Jesus O.D., 2014, *Neural Network Design*, Martin Hagan.

Hall R.C., Seborg D.E., 1989, 'Modelling and Self-Tuning Control of a Multivariable pH Neutralization Process Part I: Modelling and Multiloop Control', in *American Control Conference, 1989, Pittsburgh*, pp. 1822-1827.

Han M., Fan J., Han B., 2009, 'An Adaptive Dynamic Evolution Feedforward Neural Network on Modified Particle Swarm Optimization', in *Proceedings of International Joint Conference on Neural Networks, Atlanta*, pp. 1083-1089.

Han M., Fan J., Wang J., 2011, 'Dynamic Feedforward Neural Network based on Gaussian Particle Swarm Optimization and its Application for Predictive Control', *IEEE Transactions on Neural Networks*, 22, pp. 1457-1468.

Harriott P., 1964, *Process Control*, McGraw-Hill Book Company, USA.

Henson M.A., 1998, 'Nonlinear Model Predictive Control: Current Status and Future Directions', *Computers & Chemical Engineering*, 23, pp. 187-202.

Heredia-Molinero M.C., Sánchez-Prieto J., Briongos J.V., Palancar M.C., 2014, 'Feedback PID-like Fuzzy Controller for pH Regulatory Control near the Equivalence Point', *Journal of Process Control*, 24, pp. 1023-1037.

Hermansson A.W., Syafiie S., Noor S.B.M., 2010, 'Multiple Model Predictive Control of Nonlinear pH Neutralization System', in *Industrial Engineering and Engineering Management (IEEM), 2010 IEEE International Conference on, Macao*, pp. 301-304.

Holland J.H., 1992, *Adaptation in natural and artificial systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, MIT Press, Cambridge.

Humphrey A.E., Deindoerfer F.H., 1961, 'Fermentation', *Ind. Eng. Chem.*, 53, pp. 934-946.

Ibrahim R., Murray-Smith J., 2007, 'Design, Implementation and Performance Evaluation of a Fuzzy Control System for a pH Neutralisation Process Pilot Plant', in *Intelligent and Advanced Systems, 2007. ICIAS 2007. International Conference on, Kuala Lumpur*, pp. 1001-1006.

Jang J.-H.R., Sun C.-H., 1995, 'Neuro-Fuzzy Modeling and Control', *Proceedings of the IEEE*, 83, pp. 378-406.

Jia L., Chiu M.-S., Ge S.S., 2005, 'Iterative Identification of Neuro Fuzzy based Hammerstein Model with Global Convergence', *Ind. Eng. Chem. Res.*, 44, pp. 1823-1831.

Jiayu K., Mengxiao W., Zhongjun X., Yan Z., 2009, 'Fuzzy PID Control of the pH in an Anaerobic Wastewater Treatment Process', in *Intelligent Systems and Applications, 2009. ISA 2009. International Workshop on, Wuhan*, pp. 1-4.

Karasakal O., Guzelkaya M., Eksin I., Yesil E., Kumbasar T., 2013, 'Online Tuning of Fuzzy PID Controllers via Rule Weighing based on Normalized Acceleration', *Engineering Applications of Artificial Intelligence*, 26, pp. 184-197.

Karr C.L., Gentry E.J., 1993, 'Fuzzy Control of pH using Genetic Algorithms', *IEEE Transactions on Fuzzy Systems*, 1, pp. 46-53.

Kennedy J., 1997, 'The Particle Swarm: Social Adaptation of Knowledge," in *Evolutionary Computation 1997, IEEE International Conference on, Indianapolis*, pp. 303-308.

Kennedy J., Eberhart R., 1995, 'Particle swarm optimization', in *Neural Networks, 1995. Proceedings., IEEE International Conference on (Volume 4), Perth*, pp. 1942-1948.

Khemliche M., Mokeddem D., Khellaf A., 2002, 'Design of a Fuzzy Controller of pH by the Genetic Algorithms', in *Power Conversion Conference, 2002. PCC-Osaka 2002. Proceedings of the (Volume: 2), Osaka*, pp. 912-916 (Vol. 2).

Kim K.-K.K., Ríos-Patrón E., Braatz R.D., 2012, 'Robust Nonlinear Internal Model Control of Stable Wiener Systems', *Journal of Process Control*, 22, pp. 1468-1477.

Kim S., Mahmood W., Vachtsevanos G., Samad T., 1996, 'An Operator's Model for Control and Optimization of Industrial Processes', in *Control Applications, 1996., Proceedings of the 1996 IEEE International Conference on, Dearborn*, pp. 95-100.

King P.J., Mamdani E.H., 1977, 'The Application of Fuzzy Control Systems to Industrial Processes', *Automatica*, 13, pp. 235-242.

Klatt K.-U., Engell S., 1996, 'Nonlinear Control of Neutralization Processes by Gain-scheduling Trajectory Control', *Ind. Eng. Chem. Res.*, 35, pp. 3511-3518.

Koivisto H., Kimpimaki P., Koivo H., 1991, 'Neural Predictive Control - A Case Study', in *Intelligent Control, 1991., Proceedings of the 1991 IEEE International Symposium on, Arlington*, pp. 405-410.

Krishnapura V.G., Jutan A., 2000, 'A Neural Adaptive Controller', *Chemical Engineering Science*, 55, pp. 3803-3812.

Kuo L.-E., Melsheimer S.S., 1998, 'Wastewater Neutralization Control using a Neural Network based Model Predictive Controller', in *American Control Conference, 1998. Proceedings of the 1998 (Volume: 6), Philadelphia*, pp. 3896-3899 (Vol. 6).

Kwok D.P., Tam P., Zhou K., 1994, 'Control of a pH Neutralization Process using a Modified Elman Neural Net', in *Intelligent Information Systems,1994. Proceedings of the 1994 Second Australian and New Zealand Conference on, Brisbane*, pp. 71-75.

Kwok D.P., Wang P., 1993, 'Enhanced Fuzzy Control of a pH Neutralization Processes', in *Industrial Electronics, Control, and Instrumentation, 1993. Proceedings of the IECON '93., International Conference on, Maui*, pp. 285-288 (Vol. 1).

Lederer P.J., Li L., 1997, 'Pricing, Production, Scheduling, and Delivery-Time Competition', *Operations Research*, 45, pp. 407-420.

Leng G., Prasad G., McGinnity T.M., 2002, 'A New Approach to Generate a Self-Organizing Fuzzy Neural Network Model', in *Systems, Man and Cybernetics, 2002 IEEE International Conference on (Volume: 4), Hammamet*.

Li Q., Qu B., Ge Z., Zhan X., 2006, 'Study of Fuzzy Generalized Predictive Control Algorithm on Nonlinear Systems', in *Innovative Computing, Information and Control, 2006. ICICIC '06. First International Conference on (Volume: 1), Beijing*, pp. 437-440.

Liao Q., Li N., Li S., 2009, 'Type-II T-S Fuzzy Model-based Predictive Control', in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on, Shanghai*, pp. 4193-4198.

Lightbody G., O'Reilly P., Irwin G.W., Kelly K., McCormick J., 1997, 'Neural Modelling of Chemical Plant using MLP and B-Spline Networks', *Control Engineering Practice*, 5, pp. 1501-1515.

Lin J.-Y., Yu C.-C., 1993, 'Automatic Tuning and Gain Scheduling for pH Control', *Chemical Engineering Science*, 48, pp. 3159-3171.

Mahmoodi S., Poshtan J., Jahed-Motlagh M.R., Montazeri A., 2009, 'Nonlinear Model Predictive Control of a pH Neutralization Process based on Wiener-Laguerre Model', *Chemical Engineering Journal*, 146, pp. 328-337.

Mamdani E.H., 1974, 'Application of Fuzzy Algorithms for Control of Simple Dynamic Plant', *Proceedings of the Institution of Electrical Engineers, 121,*pp. 1585-1588.

Mamdani E.H., 1977, 'Application of Fuzzy Logic to Approximate Reasoning Using Linguistic Synthesis', *IEEE Transactions on Computers*, C-26, pp. 1182-1191.

Mamdani E.H., Assilian S., 1975, 'An experiment in linguistic synthesis with a fuzzy logic controller', *International Journal of Man-Machine Studies*, 7, pp. 1-13.

McAvoy T.J., 1972, 'Time Optimal and Ziegler-Nichols Control', *Ind. Eng. Chem. Process Des. Develop.*, 11, pp. 71-78.

McAvoy T.J., Hsu E., Lowenthals S., 1972, 'Dynamics of pH in Controlled Stirred Tank Reactor', *Ind. Eng. Chem. Process Des. Develop.*, 11, pp. 68-70.

Mellichamp D.A., Coughanowr D.R., Koppel L.B., 1966a, 'Characterization and Gain Identification of Time Varying Flow Processes', *AIChE Journal*, 12, pp. 75-82.

Mellichamp D.A., Coughanowr D.R., Koppel L.B., 1966b, 'Identification and Adaptation in Control Loops with Time Varying Gain', *AIChE Journal*, 12, pp. 83-89.

Morari M., Lee J.H., 1999, 'Model Predictive Control: Past Present and Future', *Computers & Chemical Engineering*, 23, pp. 667-682.

Narayanan N.R.L., Krishnaswamy P.R., Rangaiah G.P., 1997, 'An Adaptive Internal Model Control Strategy for pH Neutralization', *Chemical Engineering Science*, 52, pp. 3067-3074.

Norquay S.J., Palazoglu A., Romagnoli J.A., 1997, 'Model Predictive Control based on Wiener Models', *Chemical Engineering Science*, 53, pp. 75-84.

Norquay S.J., Palazoglu A., Romagnoli J.A., 1999, 'Application of Wiener Model Predictive Control (WMPC) to a pH Neutralization Experiment', *IEEE Transactions on Control System Technology*, 7, pp. 437-445.

Nyström R.H., Åkesson B.M., Toivonen H.T., 2002, 'Gain-Scheduling Controllers based on Velocity-Form Linear Parameter-Varying Models Applied to an Example Process', *Ind. Eng. Chem. Res.*, 41, pp. 220-229.

Oblak S., Škrjanc I., 2006, 'Nonlinear Model-Predictive Control of Wiener-type Systems in Continuous-time Domain using a Fuzzy-System Function Approximation', in *Fuzzy Systems, 2006 IEEE International Conference on, Vancouver*, pp. 2203-2208.

Oblak S., Škrjanc I., 2007, 'Continuous-Time Wiener-Model Predictive Control of a pH Process', in *Information Technology Interfaces, 2007. ITI 2007. 29th International Conference on, Cavtat*, pp. 771-776.

Oblak S., Škrjanc I., 2010, 'Continuous-Time Wiener-Model Predictive Control of a pH Process based on a PWL Approximation', *Chemical Engineering Science*, 65, pp. 1720-1728.

Oh S.-K., Pedrycz W., 2002, 'The design of self-organizing Polynomial Neural Networks', *Information Sciences An International Journal*, 141, pp. 237-258.

Oh S.-K., Roh S.-B., 2010, 'The Design of Fuzzy Controller Based on Genetic Optimization and Neurofuzzy Networks', *Journal of Electrical Engineering & Technology*, 5, pp. 653-665.

Oh S.-K., Roh S.-B., Kim H.-K., 2004, 'Fuzzy Controller Design by Means of Genetic Optimization and NFN based Estimation Technique', *International Journal of Control, Automation, and Systems*, 2, pp. 362-373.

Palancar M.C., Aragón J.M., Miguéns J.A., Torrecilla J.S., 1996, 'Application of a Model Reference Adaptive Control System to pH Control. Effects of Lag and Delay Time', *Ind. Eng. Chem. Res.*, 35, pp. 4100-4110.

Palancar M.C., Aragón J.M., Torrecilla J.S., 1998, 'pH Control System based on Artificial Neural Networks', *Ind. Eng. Chem. Res.*, 37, pp. 2729-2740.

Palancar M.C., Martin L., Aragón J.M., Villa J., 2007, 'PD and PID Fuzzy Logic Controllers. Application to Neutralization Processes', in *Proceedings of European Congress of Chemical Engineering (ECCE-6), Copenhagen*.

Parekh M., Desai M., Li H., Rhinehart R.R., 1994, 'In-Line Control of Nonlinear pH Neutralization Based on Fuzzy Logic', *IEEE Transactions on Components, Packaging, and Manufacturing Technology - Part A*, 17, pp. 192-201.

Pandit N.K., 2007, *Introduction to the Pharmaceutical Sciences*, Lippincott Williams & Wilkins.

Park H.C., Sung S.W., Lee J., 2006, 'Modeling of Hammerstein-Wiener Processes with Special Input Test Signals', *Ind. Eng. Chem. Res.*, 45, 1029-1038.

Pishkenari H.N., Mahboobi S.H., Alasty A., 2011, 'Optimum Synthesis of Fuzzy Logic Controller for Trajectory Tracking by Differential Evolution', *Scientia Iranica Transactions B: Mechanical Engineering*, 18, pp. 261–267.

Pottmann M., Seborg D.E., 1997, 'A Nonlinear Predictive Control Strategy based on Radial Basis Function Models', *Computers & Chemical Engineering*, 21, pp. 965-980.

Price K.V., 1996, 'Differential Evolution: A Fast and Simple Numerical Optimizer', in *Fuzzy Information Processing Society, 1996. NAFIPS., 1996 Biennial Conference of the North American, Berkeley*, pp. 524-527.

Procyk T.J., Mamdani E.H., 1979, 'A Linguistic Self-Organizing Process Controller', *Automatica*, 15, pp. 15-30.

Proudfoot C.G., Gawthrop P.J., Jacobs O.L.R., 1983, 'Self-Tuning PI Control of a pH Neutralisation Process', *Control Theory and Applications, IEE Proceedings D*, 130, pp. 267-272.

Public.Resource.Org, Inc., 2012, *Indian Standad: Drinking Water - Specification (Second Revision)*, Available: (https://law.resource.org/pub/in/bis/S06/is.10500.2012.pdf).

Qin S.J., Badgwell T.A., 2003, 'A Survey of Industrial Model Predictive Control Technology', *Control Engineering Practice*, 11, pp. 733-764.

Rivera D.E., Morari M., Skogestad S., 1986, 'Internal Model Control. 4. PID Controller Design', *Ind. Eng. Chem. Process Des. Develop.*, 25, pp. 252-265.

Rodrigues C.S.D., Madeira L.M., Boaventura R.A.R., 2014, 'Decontamination of an Industrial Cotton Dyeing Wastewater by Chemical and Biological Processes', *Ind. Eng. Chem. Res.*, 53, pp. 2412−2421.

Roh S.-B., Pedrycz W., Oh S.-K., 2007, 'Genetic Optimization of Fuzzy Polynomial Neural Networks', *IEEE Transactions on Industrial Electronics*, 54, pp. 2219-2238.

Rumelhart D.E., Hinton G.E., Williams R.J., 1985, *Learning Internal Representations by Error Propagation*, Institute for Cognitive Science Report 8506, University of California.

Saji K.S., Sasi M.K., 2010, 'Fuzzy Sliding Mode Control for a pH Process', in *Communication Control and Computing Technologies (ICCCCT), 2010 IEEE International Conference on, Ramanathapuram*, pp. 276-281.

Salehi S., Shahrokhi M., Nejati A., 2009, 'Adaptive Nonlinear Control of pH Neutralization Processes using Fuzzy Approximators', *Control Engineering Practice*, 17, pp. 1329-1337.

Semple S.J.G., Mattock G., Uncles R., 1962, 'A Buffer Standard for Blood pH Measurements', *The Journal of Biological Chemistry*, 237, pp. 963-967.

Shafiee G., Arefi M.M., Jahed-Motlagh M.R., Jalali A.A., 2006, 'Model Predictive Control of a Highly Nonlinear Process Based on Piecewise Linear Wiener Models', in *E-Learning in Industrial Electronics, 2006 1ST IEEE International Conference on, Hammamet*, pp. 113-118.

Shahraeini Z., Daneshpour N., Motlagh M.R.J., Poshtan J., 2006, 'A Nonlinear Model Predictive Control System Based on Wiener Model', in *Control, Automation, Robotics and Vision, 2006. ICARCV '06. 9th International Conference on, Singapore*, pp. 1-6.

Sharma S.K., Sutton R., Irwin G.W., 2012, 'Dynamic evolution of the genetic search region through fuzzy coding', *Engineering Applications of Artificial Intelligence*, 25, pp. 443-456.

Shi Y., Eberhart R., 1998, 'A Modified Particle Swarm Optimizer', in *Evolutionary Computation Proceedings 1998, IEEE World Congress on Computational Intelligence, The 1998 IEEE International Conference on, Anchorage*, pp. 69-73.

Shinskey F.G., 1979, *Process-Control Systems: Application/Design/Adjustment*, McGraw-Hill, Inc., USA.

Sickel J.H.V., Lee K.Y., Heo J.S., 2007, 'Differential Evolution and its Applications to Power Plant Control', in *Intelligent Systems Applications to Power Systems 2007, ISAP 2007, International Conference on, Toki Messe*, pp. 1-6.

Sing C.H., Postlethwaite B., 1997, 'pH Control Handling Nonlinearity and Deadtime with Fuzzy Relational Model-based Control', *Control Theory and Applications, IEE Proceedings*, 144, pp. 263-268.

Sivaraman E., Arulselvi S., Babu K., 2011, 'Data Driven Fuzzy C-means Clustering Based on Particle Swarm Optimization for pH process', in *Emerging Trends in Electrical and Computer Technology (ICETECT), 2011 International Conference on, Tamil Nadu*, pp. 220-225.

Stock J.T., 1991, 'Early Industrial pH Measurement and Control', *Bulletin for the History of Chemistry*, 10, pp. 31-34.

Storn R., 1996, 'On the Usage of Differential Evolution for Function Optimization', in *Fuzzy Information Processing Society, 1996. NAFIPS., 1996 Biennial Conference of the North American, Berkeley*, pp. 519-523.

Storn R., Price K., 1996, 'Minimizing the Real Functions of the ICEC'96 contest by Differential Evoltion', in *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on, Nagoya*, pp. 842-844.

Syed A.H., Abido M.A., 2013, 'Differential Evolution based Intelligent Control for Speed Regulation of a PMDC Motor', in *Control & Automation (MED) 2013, 21st Mediterranean Conference on, Chania*, pp. 1451-1456.

Takagi T., Sugeno M., 1985, 'Fuzzy identification of systems and its applications to modeling and control', *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15, pp. 116-132.

Tan W.W., Lu F., Loh A.P., Tan K.C., 2005, 'Modeling and Control of a Pilot pH Plant using Genetic Algorithm', *Engineering Applications of Artificial Intelligence*, 18, pp. 485–494.

Tang Y., Qiao L., Guan X., 2010, 'Identification of Wiener Model using Step Signals and Particle Swarm Optimization', *Expert Systems with Applications*, 37, pp. 3398–3404.

Toivonen H.T., Sandström K.V., Nyström R.H., 2003, 'Internal Model Control of Nonlinear Systems described by Velocity-based Linearizations', *Journal of Process Control*, 13, pp. 215-224.

Tzafestas S., Papanikolopoulos N.P., 1990, 'Incremental Fuzzy Expert PID Control', *IEEE Transactions on Industrial Electronics*, 37, pp. 365-371.

Valarmathi K., Devaraj D., Radhakrishnan T.K., 2008, 'A Combined Genetic Algorithm and Sugeno Fuzzy Logic based approach for On-Line Tuning in pH Process', in *Intelligent Systems, 2008. IS '08. 4th International IEEE Conference (Volume: 3), Varna*, pp. 20-2 - 20-7.

Valarmathi K., Kanmani J., Devaraj D., Radhakrishnan T.K., 2007, 'Hybrid GA Fuzzy Controller for pH Process', in *Conference on Computational Intelligence and Multimedia Applications, 2007. International Conference on (Volume: 4), Sivakasi*, pp. 13-18.

Venkateswarlu K.S., 1996, *Water Chemistry: Industrial and Power Station Water Treatment*, New Age International (P) Limited, Publishers.

Venkateswarlu C., Anuradha R., 2004, 'Dynamic Fuzzy Adaptive Controller for pH', *Chemical Engineering Communications*, 191, pp. 1564-1588.

Waller K.V., Mäkilä P.M., 1981, 'Chemical Reaction Invariants and Variants and their use in Reactor Modeling, Simulation, and Control', *Ind. Eng. Chem. Process Des. Develop.*, 20, pp. 1-11.

Walling F.B., Otts L.E., 1967, *Water Requirements of the Iron and Steel Industry*, Available: (http://pubs.usgs.gov/wsp/1330h/report.pdf).

Wan F., Shang H., Wang L.-X., 2006, 'Adaptive Fuzzy Control of a pH Process', in *Fuzzy Systems, 2006 IEEE International Conference on, Vancouver*, pp. 2377-2384.

Wan F., Shang H., Wang L.-X., Sun Y.-X., 2004, 'Neutralization Process Control using an Adaptive Fuzzy Controller', in *American Control Conference, 2004. Proceedings of the 2004 (Volume: 2), Boston*, pp. 1770-1775.

Wan T.W., Kamal D.H., 2006, 'On-Line Learning Rules for Type-2 Fuzzy Controller', in *Fuzzy Systems, 2006 IEEE International Conference on, Vancouver*, pp. 513-520.

Wang P., Kwok D.P., 1992, 'Optimal Fuzzy PID Control based on Genetic Algorithm', in *Industrial Electronics, Control, Instrumentation, and Automation, 1992. Power Electronics and Motion Control., Proceedings of the 1992 International Conference on, San Diego,* pp. 977-981 (Vol. 2).

Wenfeng W., Wei L., Jin H., 2009, 'Nonlinear Model Predictive Control of pH in Rolling Mill Wastewater Treatment', in *Intelligent Systems and Applications, 2009. ISA 2009. International Workshop on, Wuhan*, pp. 1-5.

Williams T.J., 1966, 'Computers and Process Control', *Ind. Eng. Chem.*, 58, pp. 55−70.

Williams T.J., 1970, 'Computers and Process Control - Fundamentals', *Ind. Eng. Chem.*, 62, pp. 94−107.

Wilkins J., 2011, *pH: Are You in Control of a Moving Target*, Available: (http://www.pharmamanufacturing.com/articles/2011/044/?show=all).

Wright R.A., Kravaris C., 1991, 'Nonlinear Control of pH Processes using Strong Acid Equivalent', *Ind. Eng. Chem. Res.*, 30, pp. 1561-1572.

Wright R.A., Soroush M., Kravaris C., 1991, 'Strong Acid Equivalent Control of pH Processes: An Experimental Study', *Ind. Eng. Chem. Res.*, 30, pp. 2437-2444.

Yang B., Chen Y., Zhao Z., 2007, 'A Hybrid Evolutionary Algorithm by Combination of PSO and GA for Unconstrained and Constrained Optimization Problems', in *Control and Automation, 2007. ICCA 2007. IEEE International Conference on, Guangzhou*, pp. 166-170.

Yeo Y.-K., Kwon T.-I., 1999, 'A Neural PID Controller for the pH Neutralization Process', *Ind. Eng. Chem. Res.*, 38, pp. 978-987.

Yeo Y.-K., Kwon T.-I., 2004, 'Control of pH Processes based on the Genetic Algorithm', *Korean Journal of Chemical Engineering*, 21, pp. 6-13.

Ylén J.-P., 1998, 'Improved Performance of Self-Organising Fuzzy Controller (SOC) in pH Control', in *Fuzzy Systems Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on (Volume: 1), Anchorage*, pp. 258-263 (Vol. 1)

Yu X., Huang D., Wang X., Jin Y., 2008, 'DE-based Neural Network Nonlinear Model Predictive Control and its Application for pH Neutralization Process Reactor Control', in *Control and Decision Conference, 2008. CCDC 2008. Chinese, Yantai*, pp. 1597-1602.

Zadeh L.A., 1965, 'Fuzzy sets', *Information and Control*, 8, pp. 338-353.

Zadeh L.A., 1973, 'Outline of a New Approach to the Analysis of Complex Systems and Decision Processes', *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3, pp. 28-44.

Zadeh L.A., 2008, 'Is there a need for Fuzzy Logic?', *Information Sciences An International Journal*, 178, pp. 2751-2779.

Zárate L.E., Resende P., Menezes B., 2001, 'A Fuzzy Logic and Variable Structure based Controller for pH Control', in *Industrial Electronics Society, 2001. IECON '01. The 27th Annual Conference of the IEEE (Volume: 1), Denver*, pp. 37-42 (Vol. 1).

Zeybek Z., Alpbaz M., 2005, 'Fuzzy-Dynamic Matrix pH Control for Treatment of Dye Wastewater Plant', in *Computational Intelligence and Multimedia Applications, 2005. Sixth International Conference on*, pp. 118-123.

Zhang J., 2001, 'A Nonlinear Gain Scheduling Control Strategy based on Neuro-Fuzzy Networks', *Ind. Eng. Chem. Res.*, 40, pp. 3164-3170.

Zheng Z., Wang N., 2002, 'Model-Free Control based on Neural Networks', in *Machine Learning and Cybernetics, 2002. Proceedings. 2002 International Conference on (Volume: 4), Beijing*, pp. 2180-2183 (Vol. 4).

Appendix A1

Armfield Multifunctional Process Control Teaching System Specifications

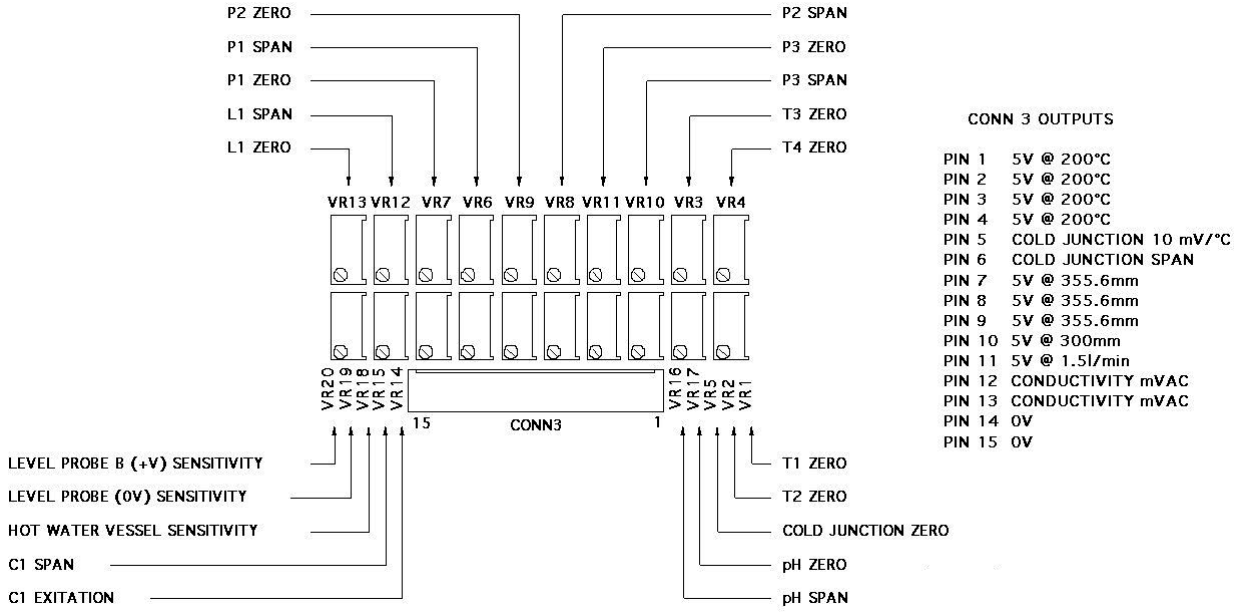
PCT40 Specifications:

Dimensions	1000 × 530 × 725 mm (W × D × H)
Input Voltage	220 V AC, 50 Hz
Fuse Rating	10 A
Temperature Sensors T1 – T4	Type, 0 – 200 °C
Pressure Sensor L1	Piezo, 0 - 300 mm H_2O (gauge)
Pressure Sensors P1 – P3	Piezo, 0 - 355.6 mm (differential)
Turbine Flowmeter	0.2 – 1.5 L/min
Solenoid Valve SOL1	Orifice 2.4 mm diameter
Solenoid Valve SOL2	Orifice 2.4 mm diameter
Solenoid Valve SOL3	Orifice 3.2 mm diameter
Proportioning Solenoid Valve	Orifice 2.4 mm diameter
Peristaltic Pump Flowrate	0 – 1.3 L/min (nominal)
Large Process Vessel Capacity (Full)	6.8 L (nominal)
Large Process Vessel Capacity (Reduced)	4 L (nominal)
Heater Power (Hot Water Vessel)	2 kW (nominal)
Maximum Hot Water Temperature	80 °C (nominal)
Flexible Tube Material	Silicone rubber
Flexible Tube Wall Thickness	1.6 mm
Flexible Tube Internal Diameter	6.4 mm
Peristaltic Pumps Manufacturer	Watson-Marlow
Peristaltic Pumps Code	Watson-Marlow 313D

PCT41 Specifications:

Dimensions	255 × 300 × 450 mm (W × D × H)
Process Vessel Diameter	153 mm
Process Vessel Depth	54 to 108 mm
Process Vessel Operating Volume	1 to 2 L
Flexible Tube Material	Silicone rubber
Flexible Tube Wall Thickness	1.6 mm
Flexible Tube Internal Diameter	3.2 mm

PCT40 Potentiometer Identifications:



PCT40 Input/Output (I/O) Connector Pin Descriptions:

Pin	IFD Function	PCT43 Function	Signal	Unit
1	Channel 0	Temperature T1	0 – 5 V	0 – 200 °C
2	Channel 1	Temperature T2	0 – 5 V	0 – 200 °C
3	Channel 2	Temperature T3	0 – 5 V	0 – 200 °C
4	Channel 3	Temperature T4	0 – 5 V	0 – 200 °C
5	Channel 4	Pressure P1	0 – 5 V	0 – 355.6 mm
6	Channel 5	Pressure P2	0 – 5 V	0 – 355.6 mm
7	Channel 6	Pressure P3	0 – 5 V	0 – 355.6 mm
8	Channel 7	Level L1	0 – 5 V	0 – 300 mm
9	Channel 8	Flowrate F1	0 – 5 V	0 – 1.5 L/min
10	Channel 9	User Input	0 – 5 V	
11	Channel 10	Conductivity	0 – 5 V	0 – 200 mS
12	Channel 11	pH	0 – 5 V	0 – 14 pH
13-15	Channel 12-14	Not Used		
16	+5 V Out	+5 V Supply		
17	Analog Ground	0 V		
18	Amp Lo	0 V		

19	+12 V Out	+12 V Supply		
20	-12 V Out	-12 V Supply		
21	Power Ground	0 V		
22	DAC0 Output	Pump A Speed	0 – 5 V	0 - 100 %
23	DAC0 Ground	0 V		
24	DAC1 Output	Pump B Speed	0 – 5 V	0 - 100 %
25	DAC1 Ground	0 V		
26-27	Digital Ground	0 V		
28	Digital Input Line 0	Not Used		
29	Digital Input Line 1	Not Used		
30	Digital Input Line 2	Hot Water Vessel Low Level		
31	Digital Input Line 3	Hot Water Vessel Over Temperature		
32	Digital Ground	0 V		
33	Digital Input Line 4	Thermostat On/Off		
34	Digital Input Line 5	Level Switch On/Off		
35	Digital Input Line 6	Not Used		
36	Digital Input Line 7	Differential Level Switch On/Off		
37	Digital Ground	0 V		
38-40	Digital O/P Line 0-2	Not Used		
41	Digital O/P Line 3	Solid State Relay Drive		
42	Digital Ground	0 V		
43	Digital O/P Line 4	Solenoid Valve SOL1 On/Off		
44	Digital O/P Line 5	Solenoid Valve SOL2 On/Off		
45	Digital O/P Line 6	Solenoid Valve SOL3 On/Off		
46	Digital O/P Line 7	PCT41 Stirrer On/Off		
47	Digital Ground	0 V		
48	Aux Output 1	USB Control		
49	Aux Output 2	Gear Pump	0 – 5 V	
50	Aux Output 3	Proportional Solenoid Valve Control	0 – 5 V	
51-53		+24 V Supply		
54-53		0 V		
57		+12 V Supply		
58		+5 V Supply		
59		-12 V Supply		
60		0 V		

Appendix A2

Chemical Reagent Specifications

Hydrochloric acid (HCl) Pure:

Manufacturer	Molychem
Molecular weight	36.46 gm
Assay (acidimetric)	35.37%
Wt. per ml at 20 °C	about 1.18 gm
Non-volatile matter	0.01%
Free-chlorine (Cl)	0.0005%
Sulphuric acid (H_2SO_4)	0.02%
Iron (Fe)	0.0005%
Lead (Pb)	0.0005%

Sodium hydroxide pellets (NaOH) Purified:

Manufacturer	Merck
Molecular weight	40 gm
Assay (NaOH)	greater than 97%
Carbonate (as Na_2CO_3)	less than 2%
Chloride (Cl)	less than 0.02%
Sulfate (SO_4)	less than 0.01%
Heavy metals (as Pb)	less than 0.002%
Iron (Fe)	less than 0.005%

Buffer capsules:

Manufacturer	Merck
pH	4 ± 0.05, 7 ± 0.05, 9.2 ± 0.05
Volume of buffer solution	100 ml

Appendix A3

Water Purifier Specification

Aquaguard Reviva:

Manufacturer	Eureka Forbes
Water Purification Process	Reverse Osmosis
Input Water Temperature	10 to 40 °C
Input Water Iron content	0.3 mg/L (max)
Input Water Chlorine content	0.2 mg/L (max)
Input Water Turbidity	15 NTU (max)
Input Water Hardness	600 mg/L (max)
Input Water Pressure	0.6 to 2 kg/cm^2
Input Water TDS	300 to 2000 mg/L
Input Voltage	230 V AC, 50 Hz
Power Rating	40 Watts
Storage Tank	8 L
Net Weight	8.5 kg
Dimensions	320 × 275 × 410 mm (W × D × H)

Appendix A4

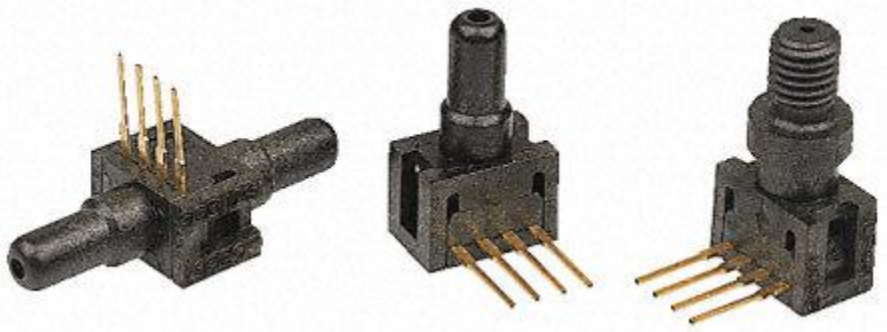
313D Rapid Load Pumphead Specification

Representative image of Watson-Marlow peristaltic pump (313D)

Body Rear	Glass filled polypropylene
Body Front	IXEF
Rotor	Glass filled Nylon
Rollers	MoS2 filled Nylon 6 (Nylatron)
Wall	1.6 mm
Bore	0.5 to 8 mm
Flexible Tube Material	Silicone rubber
Maximum Continuous Speed	400 rpm
Flowrate	0.03 to 5 mL/revolution (0.5 to 8 mm bore), 1 mL/revolution (3.2 mm bore)
Maximum Continuous Flowrate	12 to 2000 mL/min (0.5 to 8 mm bore), 400 mL/min (3.2 mm bore)
Maximum Continuous Pressure	1 to 2 bar (0.5 to 8 mm bore), 1.5 bar (3.2 mm bore)

Appendix A5

24PC Series Pressure Sensor (24PCEFA6D) Specification



Representative images of differential pressure sensors (24PCEFA6D) from Honeywell

Accuracy	0.2 %
Analogue Output	25 - 45 mV
Maximum Operating Temperature	+85 °C
Maximum Pressure Reading	0.5 psi
Media Measured	Nitrogen Gas, Oxygen, Water
Minimum Operating Temperature	-40 °C
Minimum Pressure Reading	0 psi
Output Type	Unamplified
Pressure Reading Type	Differential
Response Time	1 ms
Supply Voltage	10 V dc
Terminal Type	Solder Tag

Appendix A6

Combination pH Electrode (HI 1230B) Specifications

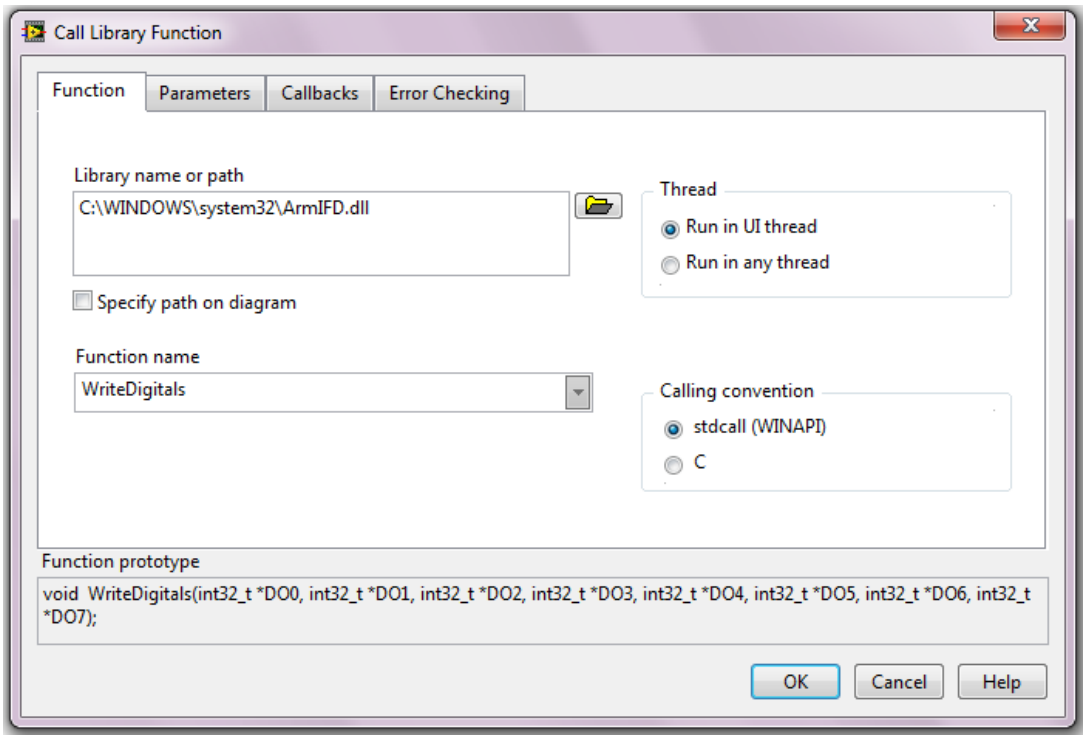
Representative image of combination pH electrode (HI 1230B) from Hanna Instruments

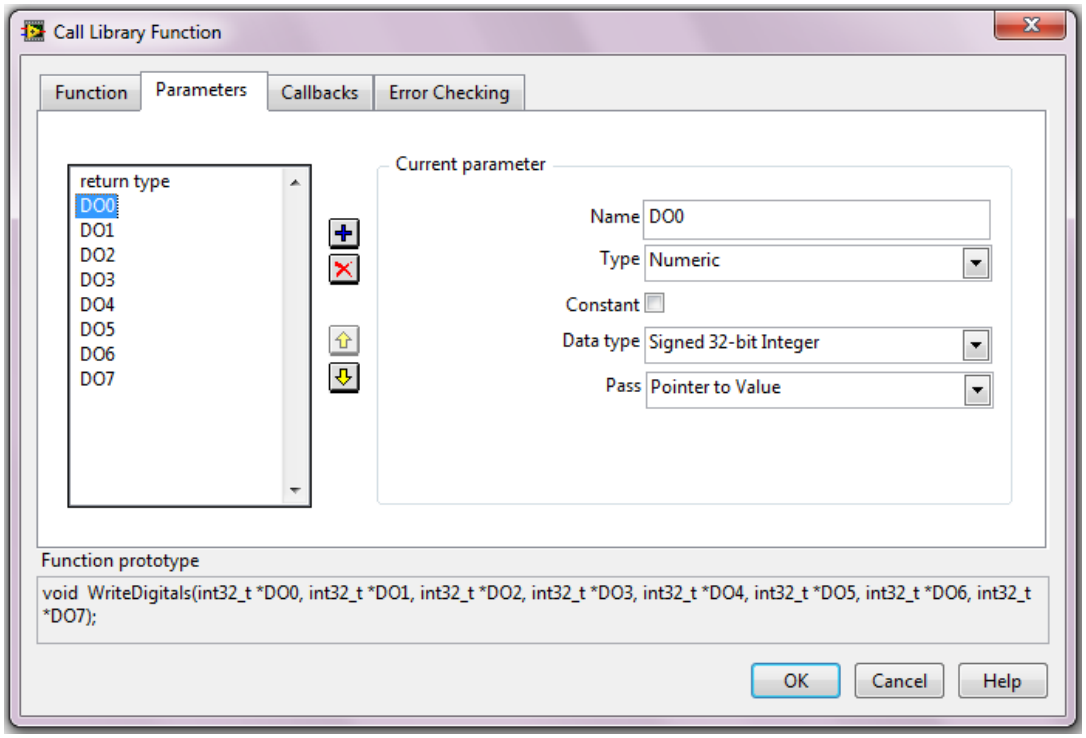
Use	Field applications
Reference	Double, Ag/AgCl
Junction/Flow Rate	Ceramic, single
Electrolyte	Gel
Max Pressure	2 bar
Range	0 to 13 pH; 0 to 80°C (68 to 104°F)
Tip Shape	Sphere (∅7.5 mm)
Temperature Sensor	No
Amplifier	No
Body Material	PEI (Polyetherimide)
Cable	Coaxial; 1 m (3.3')
Connection	BNC

Appendix A7

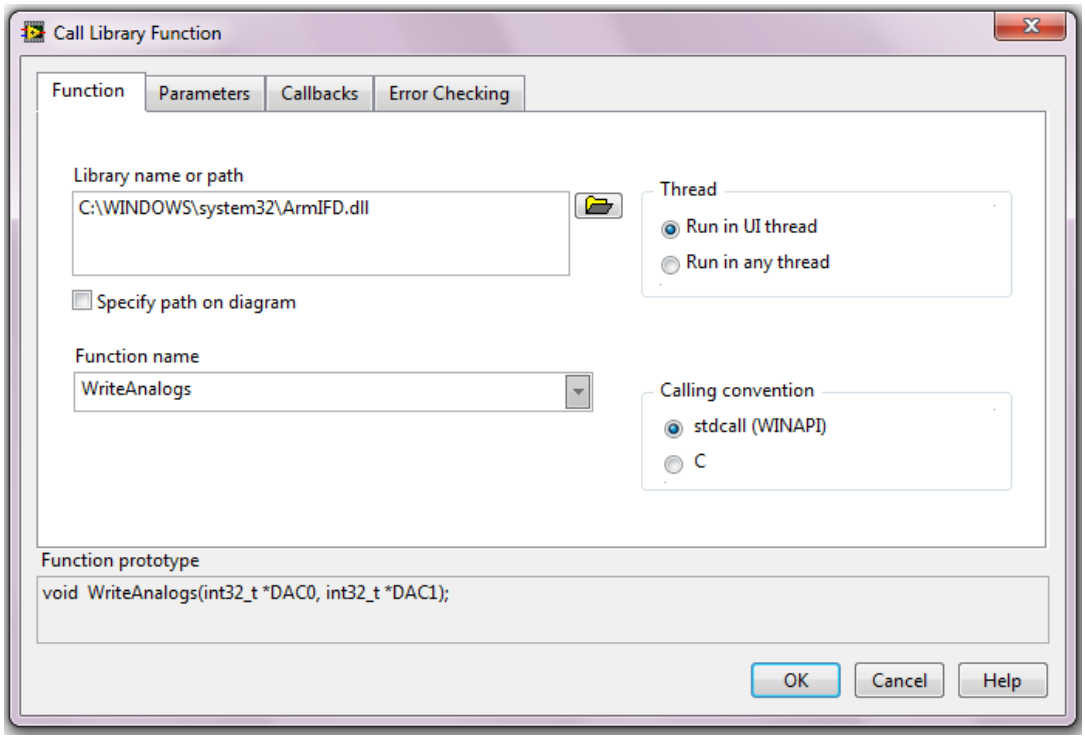
Call Library Function Node Configurations

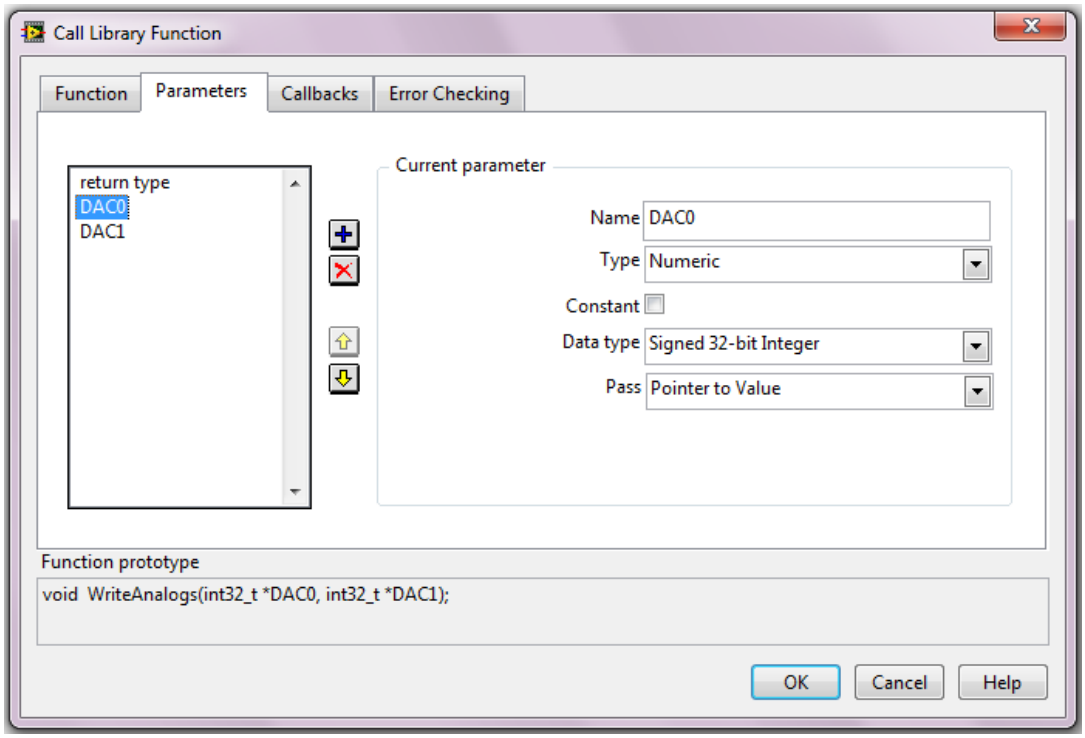
WriteDigitalsArmIFD.DLL:



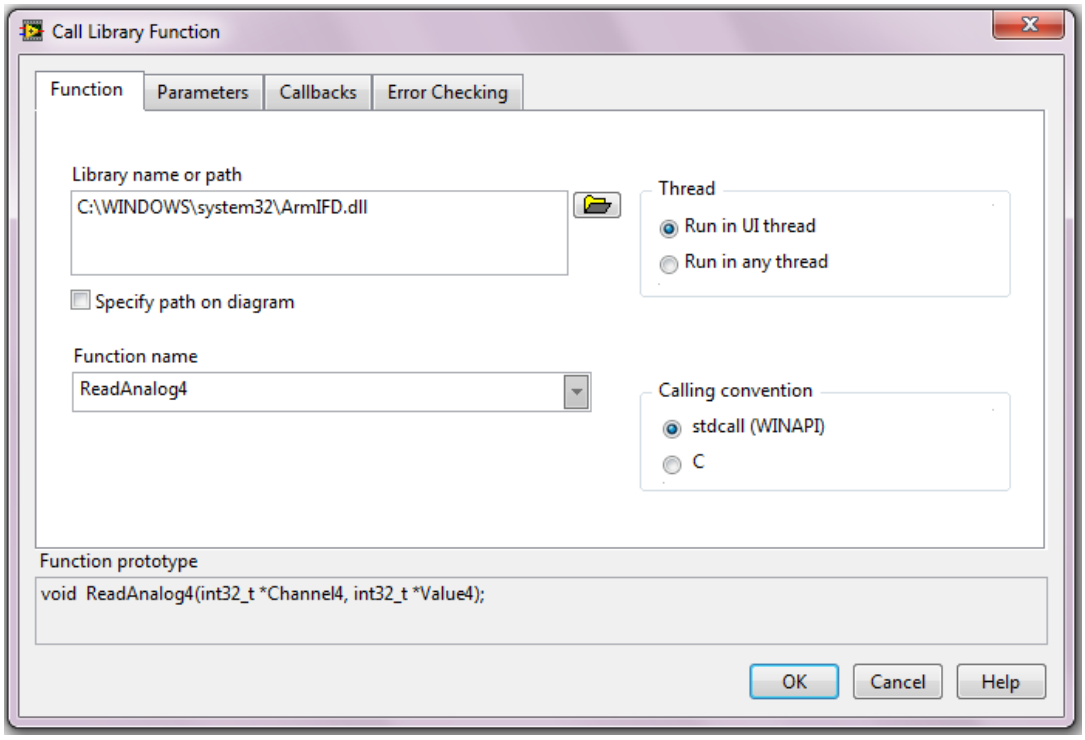


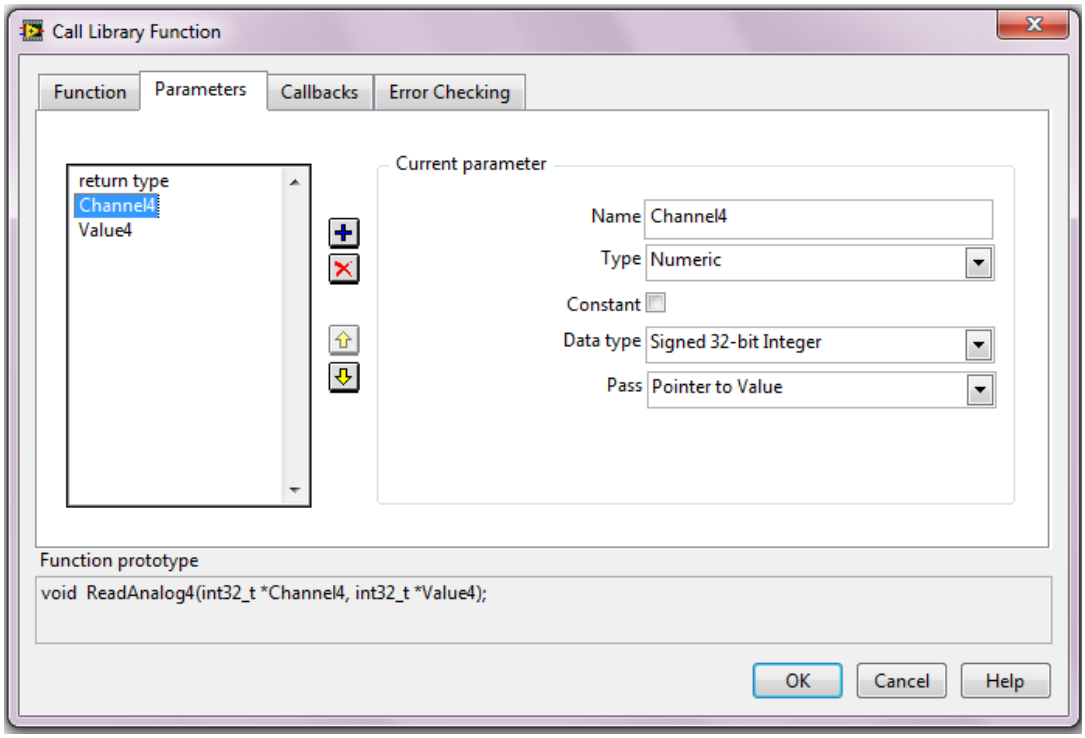
WriteAnalogsArmIFD.DLL:



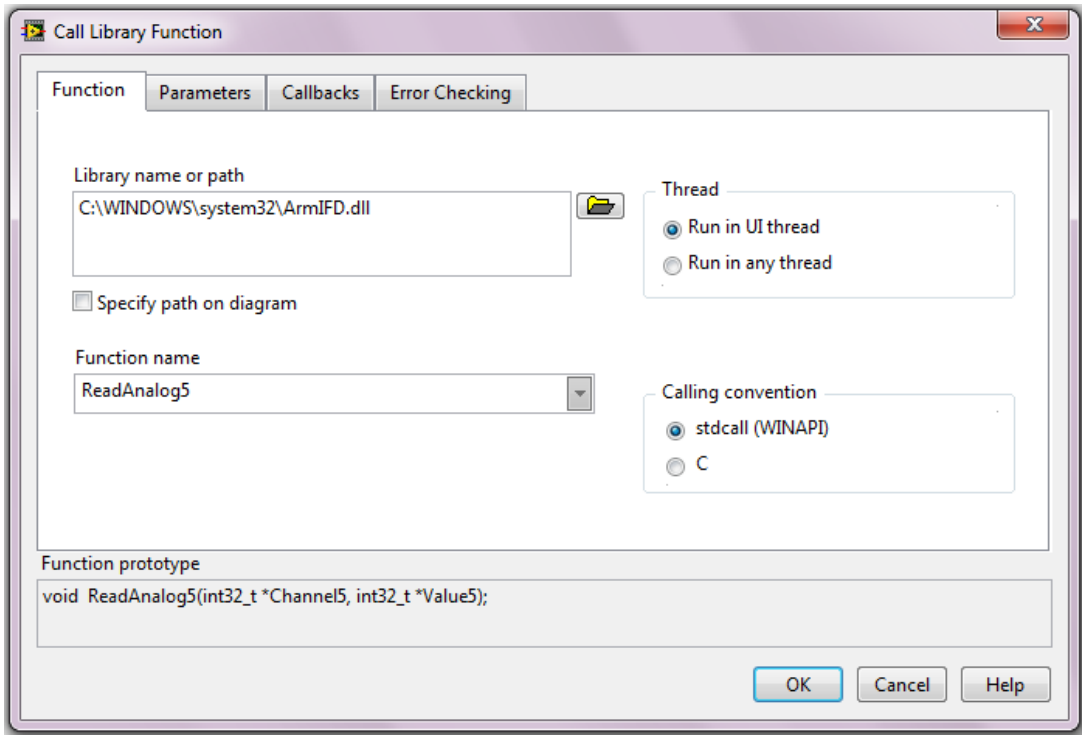


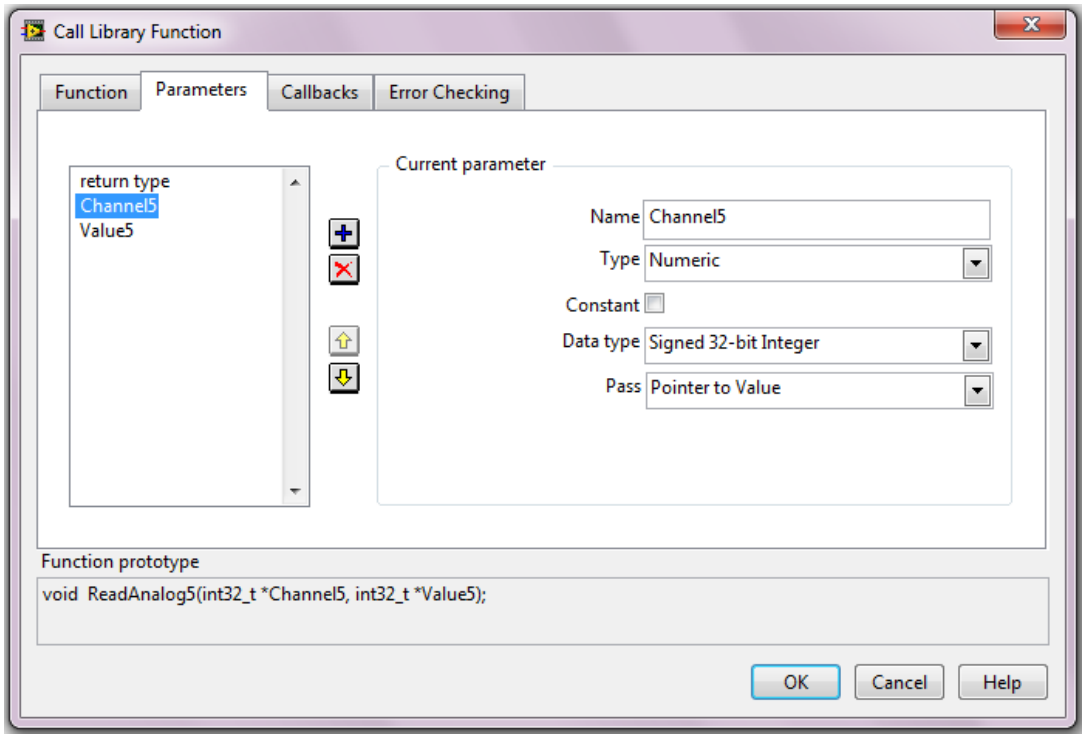
ReadAnalog4ArmIFD.DLL:



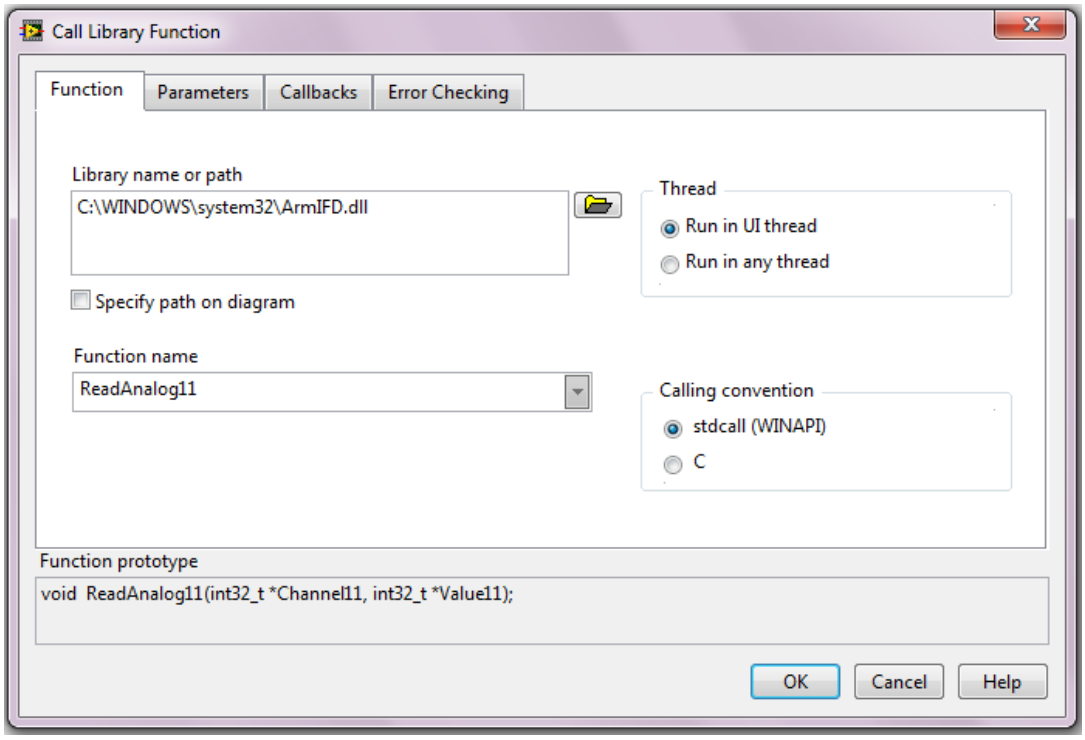


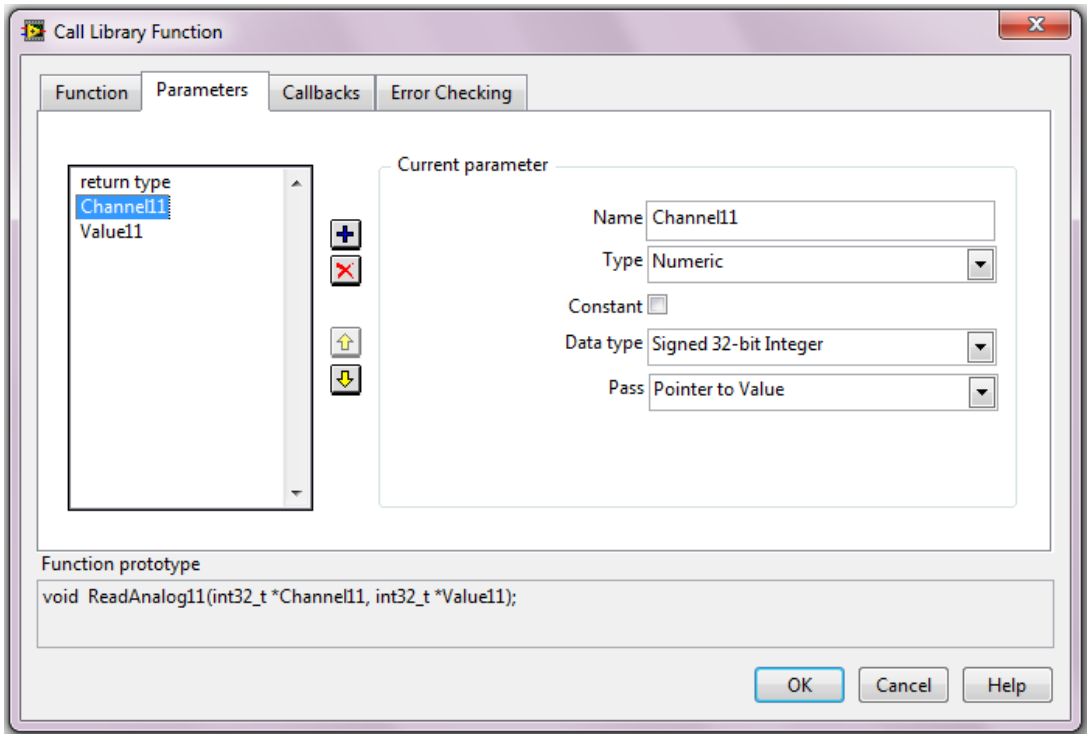
ReadAnalog5ArmIFD.DLL:





ReadAnalog11ArmIFD.DLL:





Appendix A8

Preparation of Aqueous HCl and NaOH Solutions

Aqueous HCl and NaOH solutions, with desired concentration of 0.02 M and volume 20 L each, are prepared using raw water having pH sensor voltage reading as 2.4988 V. Using Eq. (3.5), pH of raw water is calculated as 6.7121. Since the raw water is not pure distilled in nature, actual concentration of HCl and NaOH solutions differ slightly from the desired value, as mentioned in Table 4.1.

Aqueous HCl solution:

Appendix A2.1 shows that pure HCl has assay of 35.37%. It means that, there is 350 mL of HCl per 1000 mL of pure aqueous HCl solution. Since density of HCl is 1.18 gm/mL, we have 1.18×353.7 = 417.366 gm of HCl per 1000 mL of pure aqueous HCl solution. Further, the equivalent mass of HCl is 36.46 gm. Therefore molar concentration of the pure aqueous HCl solution is given as 417.366/36.46 = 11.4472 M. Suppose 'V_{HCl}' mL of pure aqueous HCl solution is added to distilled water to obtain 20 L of diluted aqueous HCl having concentration 0.02 M. Under aqueous equilibrium, we have $V_{HCl} \times 11.4472 = 20000 \times 0.02$, that is $V_{HCl} = 35$ mL (approximately).

Hence if we add 35 mL of 11.4472 M HCl to distilled water in order to make a diluted solution of volume 20 L, then the concentration of diluted HCl solution will be approximately 0.02 M.

Aqueous NaOH solution:

Appendix A2.1 shows that NaOH pellets has assay of greater than 97%. Since 40 gm of NaOH pellets in 1000 mL of distilled water gives approximately 1 M aqueous NaOH solution. Therefore 16 gm of NaOH pellets in 1000 mL of distilled water gives approximately 16/40 = 0.4 M aqueous NaOH solution.

Hence 16 gm of NaOH pellets in 20 L of distilled water gives approximately 0.4/20 = 0.02 M aqueous NaOH solution.

Appendix A9

Pseudocodes for Design of GA based pH Controller

Pseudocode to create initial population (GA01):

% GA01 implementation
Input parameters: population size, L; no. of variables, n; initial population range, $[K_L;K_U]$ = $[K_{1L}, K_{2L}, K_{3L};K_{1U}, K_{2U}, K_{3U}]$; uniformly distributed random number, rand $\in$ [0, 1]
Output parameter: initial population, K = $[K_1, K_2, K_3]$
If K is given by user
  Break
Else
  For i = 1:L
    For j = 1:n
      Obtain individual member $K(i,j) = K_L(1,j) + ((K_U(1,j)- K_L(1,j)) \times rand(1,1)$
    End
  End
End

Pseudocode to evaluate fitness function for offline operation (pH00):

% pH00 implementation
Input parameters: Initial pH values, [pH(1), pH(2), pH(3)]; initial acid flowrate values, $[S_a(1), S_a(2), S_a(3)]$; initial base flowrate values, $[S_b(1), S_b(2), S_b(3)]$; base flowrate saturation limiter, $[MV_{LB}, MV_{UB}]$; pH setpoint variations, pH_{SP}; acid flowrate variations, S_a; current generation population, Pop; L; Duration for servo and regulatory operation, T
Output parameter: Fitness function values in current generation, ISE
For i = 1:L
  Calculate initial errors e(1), e(2), e(3)
  Assign initial change in error ce(1) = 0, and calculate ce(2), ce(3)
  Calculate initial ISE values $ISE_i(1)$, $ISE_i(2)$, $ISE_i(3)$
  For k = 4:T
    Read $pH_{SP}(k)$ and $S_a(k)$
    Calculate $e(k) = pH_{SP}(k) - pH(k-1)$, $ce(k) = e(k) - e(k-1)$
    Estimate change in output co(k) using either PID or FLC schemes
    Update base flowrate $S_b(k) = S_b(k-1) + co(k)$
    Limit base flowrate such that $MV_{LB} \leq S_b(k) \leq MV_{UB}$
    Estimate pH(k) using dynamic ANN model
    Obtain fitness function value $ISE_i(k) = ISE_i(k-1) + e(k) \times e(k)$
    Assign $ISE_i(k)$ to ISE_i
  End
End
Obtain ISE = $[ISE_1; ISE_2; ISE_3; ...; ISE_L]$

Pseudocode to evaluate fitness function for online operation (pH01):

% pH01 implementation
Input parameters: Initial pH range, [pH_{LB}, pH_{UB}]; base flowrate saturation limiter, [MV_{LB}, MV_{UB}]; nominal acid flowrate, DV0; nominal base flowrate, MV0; pH setpoint variations, pH_{SP}; acid flowrate variations, DV; base flowrate variations, MV; current generation population, Pop; L; Duration for servo and regulatory operation, T
Output parameter: Fitness function values in current generation, ISE
For i = 1:L
  Write in DLL to start the stirrer
  Read from DLL to obtain pH sensor voltage
  Estimate pH using calibration equation (3.5)
  While initial pH is not within range [pH_{LB} pH_{UB}] ***% Start pH process initialization***
    If pH < pH_{LB} = $(pH_{SP})_{initial}$ - 0.1
      Write in DLL to set S_a = DV0 = 35, S_b = 35+5
    End
    If pH > pH_{UB} = $(pH_{SP})_{initial}$ + 0.1
      Write in DLL to set S_a = DV0 = 35, S_b = 35+0
    End
  End ***% End pH process initialization***
  Initialize S_a = DV0, S_b = MV0, ISE = 0, pH_{SP}, DV
  For m = 1 to Number of Set points ***% Begin pH control***
    For k = 1:T ***% For each servo and regulatory operations***
      Estimate pH(k), e(k), and ce(k)
      Estimate change in output co(k) using either PID or FLC schemes
      Update base flowrate $S_b(k) = S_b(k-1) + co(k)$
      Limit base flowrate such that $MV_{LB} \leq S_b(k) \leq MV_{UB}$
      Write in DLL to update S_a and S_b
      Update fitness function value $ISE_i(k) = ISE_i(k-1) + (e(k))^2$
      Assign $ISE_i(k)$ to ISE_i
    End ***% For each servo and regulatory operations***
  End ***% End pH control***
End
Obtain ISE = [ISE_1; ISE_2; ISE_3; ...; ISE_L]

Pseudocode to rank and scale fitness function values and determine elite kids (GA02):

```
% GA02 implementation
Input parameters: ISE; Pop; L; n; elite count, EC
Output parameters: normalized expectation, EN; minimum ISE score in current generation,
                   ISE1; elite kids in current generation, EK
Obtain 'Rank' of 'Pop' members based on sorted ISE scores in ascending order
Assign reciprocal of square root of 'Rank' of individual 'Pop' members as scaled values
Calculate summation of scaled values 'Sum E'
For i = 1:L
  Obtain normalized expectation for each 'Pop' member EN(Rank(i)) = (1/(i)^0.5)/(Sum E)
End
For i = 1:EC
  For j = 1:n
    Obtain individual elite kid EK(i,j) = Pop(Rank(i),j)
  End
End
Obtain ISE1
```

Pseudocode to select parents for crossover and mutation (GA03):

```
% GA03 implementation
Input parameters: EN; no. of crossover plus mutation parents, NCMP; rand
Output parameter: parents index for crossover and mutation, ICM
% NCMP depends upon: L; EC; crossover rate, CR; no. of mutation kids, NMK
% NCMP = 2 × (round((L - EC) × CR)) + NMK
Roulette-wheel step size RWSS = 1/NCMP
Assign EN(1) to roulette-wheel slot RW(1) where RW indicates cumulative sum of EN
For i = 2:L
  RW(i) = EN(i) + RW(i-1)
End
Random initial pointer position P = RWSS × rand(1,1)
Initialize roulette wheel slot number, say SN = 1
For i = 1: NCMP
  For j = SN:NCMP
    If P < RW(j)
      Selected parent index SP2(i) = j
      Reinitialize slot number SN = j
      Break
    End
  End
  Reinitialize the pointer position P = P + RWSS
End
Randomly permute SP2 to obtain ICM
```

Pseudocode to create crossover kids (GA04):

% GA04 implementation
Input parameters: n; Pop; ICM; rand; no. of crossover kids, NCK
Output parameter: Crossover kids, CK; starting parents index for mutation, IM
Initialize index r = 1
For p = 1:NCK
  First parent index m_1 = ICM(r)
  Second parent index m_2 = ICM(r+1)
  For q = 1:n
    If rand(1,1) > 0.5
      Obtain scattered crossover kid CK(p,q) = Pop(m_1,q)
    Else
      Obtain scattered crossover kid CK(p,q) = Pop(m_2,q)
    End
  End
Reinitialize index r = r + 2
End
Obtain IM = (2×NCK+1)

Pseudocode to create mutation kids (GA05):

% GA05 implementation
Input parameters: n; Pop; ICM; IM; rand; [K_{1L}, K_{2L}, K_{3L};K_{1U}, K_{2U}, K_{3U}]; total number of generation, G; current generation, i; no. of mutation kids, NMK; Gaussian mutation scale, MSC; Gaussian mutation shrink, MSH
Output parameter: Temporary mutation kids, MKT
Update mutation scale MSC1 = MSC×[1-(MSH× i/G)] × ([K_{1L}, K_{2L}, K_{3L}] - [K_{1U}, K_{2U}, K_{3U}])
For s = 1: NMK
  For v = 1:n
    Gaussian mutation kid MKT(s,v) = Pop(ICM(IM(s)),v) + MSC1(1,v)×(rand(1,1) - 0.5)×2
  End
End

Pseudocode to check boundary conditions of mutation kids (GA06):

% GA06 implementation
Input parameters: $[K_L;K_U] = [K_{1L}, K_{2L}, K_{3L};K_{1U}, K_{2U}, K_{3U}]$; rand; MKT
Output parameter: Mutation kids with boundary check, MK; Temporary mutation kids with out of boundary cases, Boundary

```
For s = 1:NMK
  For v = 1:n
    If MKT(s,v) < K_L(1,v)
      Mutation kid is less than lower boundary value so set Boundary (s,v) = 'True'
      Create new element as MK(s,v) = K_L(1,v) + ((K_U(1,v) - K_L(1,v)) × rand(1,1)
    Elseif MKT(s,v) ≥ K_U(1,v)
      Mutation kid is greater than upper boundary value so set Boundary (s,v) = 'True'
      Create new element as MK(s,v) = K_L(1,v) + ((K_U(1,v) - K_L(1,v)) × rand(1,1)
    Else
      Mutation kid is within boundary value so set Boundary (s,v) = 'False'
      Set MK(s,v) = MKT(s,v)
    End
  End
End
```

Pseudocode to check termination criteria (GA07):

% GA07 implementation
Input parameters: G; ISE1; Current generation, i; Desired minimum fitness function values, ISE1L; Desired difference between two consecutive minimum fitness function values, DISE1L
Output parameter: Stop signal, Stop

```
Calculate stop generation limit as half of total generation i.e. SGL = G/2
Initialize difference between two consecutive minimum fitness values, DISE1 = 0
If i > 2
  Calculate DISE1(i) = ISE1(i) - ISE1(i-1)
End
If i ≤ SGL
  Keep optimization process running with Stop = 'False'
Elseif (i > SGL & (ISE1 < ISE1L)) | (i > SGL & (DISE1 < DISE1L))
  Terminate optimization process with Stop = 'True'
Elseif i = G
  Terminate optimization process with Stop = 'True'
End
```

Appendix A10

Pseudocodes for Design of DE based pH Controller

Pseudocode to select competitive population members for current generation (DE02):

```
% DE02 implementation
Input parameters: Initial population, Pop0; temporary population obtained using DE, PopT;
                  temporary fitness function values, ISET
Output parameter: Acceptable population after performance comparison, Pop; acceptable
                  fitness function values after performance comparison, ISE
If current generation = 1
  Obtain fitness function values for first generation population Pop0 as ISE0 = ISET
  Assign ISE0 to ISE and Pop0 to Pop
Elseif current generation > 1
  For i = 1:L
    If ISET(i) < ISE0(i)
      Accept better performed member as Pop(i) = PopT(i) and Update ISE(i) = ISET(i)
    Elseif ISET(i) ≥ ISE0(i)
      Reject new member and keep original values as Pop(i) = Pop0(i) and ISE(i) = ISE0(i)
    End
  End
End
```

Pseudocode to subject population members with random shuffling (DE03):

```
% DE03 implementation
Input parameters: L; Pop; no. of shuffling, N_S
Output parameter: Shuffled populations, Pop1, Pop2, Pop3
Generate random permuted vector of length N_S, and integer values between 1 to N_S
Generate integer values from 1 to L
Generate random permuted vector of length L and integer values 1 to L
Using above generated values, obtain random indices RI1, RI2, RI3
Shuffle Pop with RI1, RI2 and RI3 to obtain Pop1, Pop2 and Pop3 respectively
```

Pseudocode to create trial population with differential mutation and crossover (DE04):

```
% DE04 implementation
Input parameters: n; L; rand; Pop; Pop1; Pop2; Pop3; Weight factor, Weight; Crossover rate,
                  CR
Output parameter: Shuffled original population, PopOrg; DE based population, PopDE
For i = 1:L
  For j = 1:n
    If rand(1,1) < CR
      Assign crossover masks inverse values with MO1(i,j) = 1 and MO2(i,j) = 0
    Else
      Assign crossover masks inverse values with MO1(i,j) = 0 and MO2(i,j) = 1
    End
  End
End
Apply DE to generate PopDE = (Pop3 + Weight × (Pop1 - Pop2)) × MO1 + Pop × MO2
Assign Pop3 to PopOrg = Pop3
```

Pseudocode to check boundary conditions for trial population (DE05):

```
% DE05 implementation
Input parameters: [K_L;K_U] = [K_1L, K_2L, K_3L;K_1U, K_2U, K_3U]; rand; PopDE; PopOrg
Output parameter: PopT; Boundary
For s = 1:L
  For v = 1:n
    If PopDE(s,v) > K_U(1,v)
      PopDE member is greater than upper boundary value so set Boundary (s,v) = 'True'
      Create new element as PopT(s,v) = K_U(1,v) + ((PopOrg(1,v) - K_U(1,v)) × rand(1,1)
    Elseif PopDE(s,v) < K_L(1,v)
      PopDE member is less than lower boundary value so set Boundary (s,v) = 'True'
      Create new element as PopT(s,v) = K_L(1,v) + ((PopOrg(1,v) - K_L(1,v)) × rand(1,1)
    Else
      PopDE member is within boundary value so set Boundary (s,v) = 'False'
      Set PopT(s,v) = PopDE(s,v)
    End
  End
End
```

Appendix A11

Pseudocodes for Design of PSO based pH Controller

Pseudocode to determine global and local best particles positions and fitness function values (PS02):

% PS02 implementation
Input parameters: L; G; Pop; ISE; local best position of last generation, LBX; local best ISE of last generation, LBISE; global best value, GBISE1; array of global best values, GBISE; global best member, GBK; current generation, i
Output parameter: ISE1; updated LBX; updated LBISE; updated GBISE1; updated GBISE; updated GBK

```
For i = 1:G
  For s = 1:L
    If ISE(s) < LBISE(s)
      Update LBISE(s) = ISE(s) and LBX(s) = Pop(s)
    End
  End
  Calculate minimum fitness value ISE1, and ISE1 index
  If ISE1 < GBISE1
    Update GBISE1 = ISE1, GBISE(g) = ISE1, and GBK = Pop(ISE1 index)
  End
End
```

Pseudocode to update particles velocity, position and inertia (PS03):

% PS03 implementation
Input parameters: n; L; G; Pop; LBX; GBK; rand; lower and upper inertia limit, [C_{LB}, C_{UB}]; particle inertia, C; particle velocity, V; cognitive attraction, C_1; social attraction, C_2; current generation, i
Output parameter: C; V; Pop

Generate L× n random arrays R_1 and R_2
Repeat L times GBK to generate L× n matrix of global best members, say GBKL
Update particle velocity as $V = C \times V + (LBX - Pop) \times C_1 \times R_1 + (GBKL - Pop) \times C_2 \times R_2$
Update particle position as Pop = Pop + V
Update particle inertia as $C = C_{UB} - (C_{LB} \times i / G)$

Brief Biography of Candidate

Parikshit Kishor Singh was born in Kolkata, India in 1977. He received the B.E. (Electronics Engineering) and M.Tech. (Electronics and Instrumentation) degrees from Sardar Vallabhbhai National Institute of Technology (SVNIT) Surat and National Institute of Technology (NIT) Warangal, in 2003 and 2005, respectively. In 2005, he joined the Vignan's Institute of Information Technology (VIIT) Vishakhapatnam as Assistant Professor, under Early Faculty Induction Programme (EFIP) of All India Council of Technical Education (AICTE). In 2007, he joined BITS Pilani as Assistant Lecture. From 2009 onwards, he is designated as Lecturer at BITS Pilani. From 2008 onwards, he is pursuing his doctoral research in the area of applications of computational intelligence in process control.

Brief Biography of Supervisor

Surekha Bhanot was born in Patiala, India in 1957. She received the B.E. (Mechanical Engineering) and M.Phil. (Instrumentation) degrees from BITS Pilani, and Ph.D. degree from Indian Institute of Technology (IIT) Roorkee, in 1979, 1983 and 1995, respectively. In 1979, she joined BITS Pilani as Teaching Assistant. In 1983, she joined Thapar Institute of Engineering and Technology (TIET) Patiala. In 2002, she joined BITS Pilani as Associate Professor. From 2006 onwards, she is designated as Professor at BITS Pilani. Her research area of interest includes artificial intelligence applications in process modeling and control, biosensor design and applications, and biomedical signal processing and related instrumentation.

Brief Biography of Co-Supervisor

Hare Krishna Mohanta was born in Odisha, India in 1972. He received the B.E. (Chemical Engineering), M.Tech. (Chemical Engineering) and Ph.D. degrees from NIT Rourkela, IIT Kanpur and BITS Pilani, in 1995, 1998 and 2006, respectively. In 1995, he joined as a graduate engineer trainee in Indian Rare Earths Limited, Chhatrapur and worked for around one year before he left for M.Tech. in IIT Kanpur. After completing M.Tech., he joined BITS Pilani as an Assistant Lecturer in 1998, became Lecturer in 2000 and Assistant Professor in 2006. He has over 16 years of teaching and 1 year industrial experience. His research area of interest includes advanced process control, process monitoring and control, applied wavelet analysis and modeling & Simulation.