Design of Intelligent FOPID/PID controllers Using Meta-heuristics and NARXnets

Submitted in partial fulfilment of the requirements

for the award of the degree of

Doctor of Philosophy

by

Vijaya Kumar Munagala

(Roll No: 719071)

Under the supervision of

Dr. J. Ravi Kumar

Professor

Department of Electronics & Communication Engineering

National Institute of Technology Warangal

Telangana, India - 506004

2023

Dedicated

To

My Family,

Gurus & Friends

Declaration

This is to certify that the work presented in this thesis entitled **Design of Intelligent FOPID/PID controllers Using Meta-heuristics and NARXnets** is a bonafied work done by me under the supervision of **Prof. J. Ravi kumar** and was not submitted elsewhere for the award of any degree.

I declare that this written submission represents my own ideas and even considered others ideas which are adequately cited and further referenced the original sources. I understand that any violation of the above will cause disciplinary action by the institute and can also evoke panel action from the sources or from whom proper permission has not been taken when needed. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea or data or fact or source in my submission.

Place:

Date:

Vijaya Kumar Munagala
Research Scholar
Roll No.: 719071

NATIONAL INSTITUTE OF TECHNOLOGY

WARANGAL, INDIA-506004

Department of Electronics & Communication Engineering

CERTIFICATE

This is to certify that the thesis work entitled **Design of Intelligent FOPID/PID controllers Using Meta-heuristics and NARXnets** is a bonafide record of work carried out by **Mr. Vijaya Kumar Munagala (Roll No.719071)** submitted to the faculty of **Electronics & Communication Engineering** department, in partial fulfilment of the requirements for the award of the degree of **Doctor of Philosophy** in **Electronics and Communication Engineering, National Institute of Technology Warangal, India-506004**. The contributions embodied in this thesis have not been submitted to any other university or institute for the award of any degree.

Prof. J. Ravi Kumar

Research Supervisor

Professor

Department of ECE

NIT Warangal, India-506 004.

Place:

Date:

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor Prof. J. Ravi Kumar for the continuous support of my Ph.D. study and related research, for his patience, motivation, and guidance with his moral values. My sincere thanks to him for providing me an opportunity to join the institute as a Ph.D. research scholar and giving me access to the research facilities. I find no words inadequate to thank him for enabling me to complete this work in spite of all obstacles. The thesis would not have seen the light of the day without his insistent support and cooperation. I felt very happy working with you over the past few years.

My special words of thanks should also go to Prof.Patri Sreehari Rao, Head of the department, Electronics and Communication Engineering Department, NIT Warangal for his valuable suggestions and support that he shared during my research tenure.

Here, I also would like to take this privilege to thankful my DSC members, Prof. Patri Sreehari Rao, Department of Electronics and Communication Engineering, NIT Warangal, Prof. V. T. Somasekhar and Prof. N. V. Srikanth Department of Electrical and Electronics Engineering, NIT Warangal, Dr. P. Muralidhar, Associate Professor and Dr. B. K. N. srinivasa Rao, Assistant Professor, Department of Electronics and Communication Engineering, NIT Warangal for their continuous support, suggestions and advice during my research period whenever required.

I also appreciate the encouragement from teaching staff, non-teaching members and fraternity of Dept. of E.C.E. in N.I.T. Warangal. They have always been encouraging and supportive towards my research.

I thank my fellow labmates (Dr. Jailsingh Bhookya, Dr. Kishor Ingle, Dr. Prathap soma, Mr. A. B. Ahadit, Mr.D. Laxmi Kantham, Smt. K. Sri vidya and Mr. M. Naresh)

for the stimulating discussions, we were working together before deadlines, and for all the fun we have had in the last few years.

It is my pleasure to show my indebtedness to my co-scholars at NITW like Mr. P. Raveendra and friends for their help during the course of this work. Their timely help and friendship shall always be remembered during my stay.

My heartfelt special thanks goes to my father Mr. M. Jagadeeswara Rao and mother Smt. M. Seeta Kumari for supporting me throughout my life and always a great inspiration. Their unconditional love, support, self-sacrifices and patience has always been the source of my strength. I shall be grateful forever because of their kindness and love.

Moreover, I find no words inadequate to express any form of acknowledgment to my sister's family Mr. N.Balaji, smt. N. Madhuri and my liitle lassie Devanshi & for their love, help and moral support in any situation, I shall be grateful forever to them.

Last but not least, I want to mention my sincere thanks to DST-ICPS, New Delhi, India for their financial support, which has supported me so much during this research.

Finally, I thank God, for filling me every day with new hopes, strength, purpose and faith.

Vijaya Kumar Munagala

Abstract

Intelligent control systems is a branch of engineering which deals with solving control system problems using non-traditional computing techniques such as nature- inspired algorithms and neural networks. The classical computing techniques can give the satisfactory performance or acceptable solutions, for complex systems identification and control is a laborious task. Therefore to efficiently solve identification as well as controller design, tuning corresponding parameters with meta-heuristics and neural networks are investigated in the work.

In recent times, the significance of neural networks in control systems has increased because of their learning and universal approximation capabilities. Particularly, system identification and corresponding controller design becomes difficult, if the system dynamics are complex. A novel method was proposed for identifying system dynamics and design of neural network based FOPID controller. A class of RNNs(Recurrent Neural Networks) called NARXnets (Nonlinear Auto Regressive with eXogenous input) have been used to recognize the system dynamics. Later, this neural network was used to identify the best suitable neural controller from FOPID controller data. To verify the proposed method, a separately excited DC motor is considered as plant and HHO(Harris Hawks Optimization) algorithm-tuned FOPID system as the reference controller. The motor and controller dynamics were captured using the designed NARXnet. The simulation results show that the proposed controller is performing superior to the conventional FOPID / PID controllers. In addition, the proposed method can also be used as an alternative technique to approximate FOPID controllers using neural networks.

A novel architecture based on neural networks is proposed for the FOPID controller. A new optimization algorithm using chaotic maps, called Chaotic Political Optimization (CHPO) is developed to find the optimal weights of the neural network. To verify the efficiency of the proposed single neuron FOPID (SNFOPID) controller, a separately excited

DC (direct current) motor is considered as plant and its response is optimized. A new cost function is defined based on the performance metrics of the plant and overall system error. The simulation results show that the cost function can efficiently optimize the system response. The performance metrics of the proposed controller are compared with different existing controllers. The results show that the SNFOPID controller has produced better rise time and settling time than the existing techniques. The controller is tested in different scenarios like load changes, sudden disturbances, and sinusoidal set-point variations. In all the cases, the proposed controller shown its superiority.

Optimal tuning of fractional order proportional integral derivative (FOPID) controller parameters for automatic voltage regulator (AVR) system is a complex problem that requires solution of real order integral and differential equations. Therefore, a novel optimization technique called Chaotic Black Widow Optimization (ChBWO) algorithm is proposed to tune the parameters of the FOPID controller. A new cost function is defined with a combination of Zwe-Lee Gaing's (ZLG) and integral of time multiplied absolute error (ITAE) objective functions. The proposed controller performance (rise time, settling time, and overshoot) is compared with the existing state-of-the-art techniques. To verify the robustness of the proposed controller, the plant parameters deviated from -50% to 50%. The simulation results show that the controller is robust to deviations in plant parameters. Disturbance analysis is also carried out by incorporating sudden changes in the reference signal. The simulation results demonstrate that the controller is stable, robust, and sustains sudden disturbances.

Identification of optimal controller parameters for a system requires a model of the system and complex tuning procedures. Since neural networks have excellent approximation abilities, they can be used for system model identification and controller design. A new methodology was proposed using neural networks for automatic identification of controller. A class of recurrent neural networks (RNN) called NARXnets (Non-linear AutoRegressive with eXogenous input networks) are used for system identification and controller design. Initially, the plant dynamics are identified using plant excitation and response data. Later, the identified plant was used to design the NARXnet-based neural network controller. To train the NARXnets, Bayesian regularization (BR) backpropagation algorithm is used. The algorithm avoids over-fitting and reduces the number of

training parameters using its inherent procedures. To verify the proposed method, the automatic voltage regulator (AVR) system is considered as plant. The system response is optimized by properly training the controller neural network. The response of the proposed controller is compared with the state-of-the-art methodologies. The simulation results showed that the proposed controller can track the reference signal efficiently. The disturbance and load response analysis show that the overall system is stable.

To optimize AVR system response, a novel tuning method was proposed for sigmoid proportional, integral, and derivative controller (SPID). The method uses the jellyfish search optimization (JSO) algorithm to identify the optimal parameter for the SPID controller. The controller parameters are found using the ITAE objective criteria. The performance of the SPID controller is compared with the existing state-of-the-art PID controllers. The analysis of simulated results indicated that the proposed SPID controller can considerably reduce the overshoot compared to the traditional PID controllers. The proposed JSO-SPID controller can stabilize the overall AVR system and optimize the total system response. The disturbance analysis and reference signal tracking showed that the proposed controller can be used under different operating conditions. Robust analysis was also carried out to study the controller response to uncertain variations in the AVR system parameters. The simulated results indicated that the JSO-SPID controller is stable and produces robust performance for variations in plant parameters.

Finally, the HHO-FOPID controller was realized for DC motor speed control using DSPACE tools. During the implementation, different objective functions are considered to find the optimal controller for the DC motor. The results showed that the fractional order controller can control the speed of the DC motor successfully.

Contents

List of Figures

List of Tables

List of Abbreviations

ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
ANN	Artificial Neural Network
ASO	Atom Search Optimization
AVR	Automatic Voltage Regulator
BBO	Biogeography-Based Optimization
BF	Bacterial Foraging
BR	Bayesian Regularization
BWO	Black Widow Optimization
C-ABC	Cauchy Mutated Arti cial Bee Colony
CGGSA	Caushy and Gaussian mutated Gravitational Search Algorithm
ChASO	Chaotic Atom Search Optimization
ChBWO	Chaotic Black Widow Optimization
ChMO	Chaotic Multi Objective
CHPO	Chaotic Political Optimization
CNC-ABC	Cyclic exchange Neighborhood with Chaos-Artificial Bee Colony
CR	Cannibalism Rate
CRONE	Commande Robuste dordre Non Entier
CRPSO	Crazyness Particle Swarm Optimization
CS	Cuckoo Search
DC	Direct Current
DE	Differential Evolution
FFNN	Feed Forward Neural Network
FOA	Fruit fly Optimization Algorithm

FOMCON	Fractional Order Modeling and Control
FOMRAC	Fractional Order Model Reference Adaptive Controller
FOPID	Fractional Order Proportional Integral Derivative
FPSOMA	Fractional Particle Swarm Optimization based Memetic Algorithm
GA	Genetic algorithm
GBMO	Gases Brownian Motion Optimization
GOA	Grasshopper Optimization Algorithm
GWO	Grey Wolf Optimizer
HAS	Harmony Search Algorithm
HGSO	Henry Gas Solubility Optimization
HHO	Harris Hawks Optimization
IAE	Integral of Absolute Error
IEMGA	Improved Electro-Magnetism Algorithm and Genetic Algorithm
IGBT	Insulated Gate BipolarTransistor
IOPID	Integer Order Proportional Integral Derivative
ISE	Integral of Squared Error
ITAE	Integral of Time multiplied Absolute Error
ITSE	Integral of Time multiplied Squared Error
JOA	Jaya Optimization Algorithm
JSO	Jellyfish Search Optimization
KIA	Kidney Inspired Algorithm
LM	Levenberg Marquardt
LUS	Local Unimodal Sampling
MGWO	Modifed Grey Wolf Optimizer
MLP	Multi-Layer Perceptron
MOA	Moth-flame Optimization Algorithm
MR	Mutation Rate
MRAC	Model Reference Adaptive Control
MRE	Mean Relative Error
MRFO-SA	Manta Ray Foraging Optimization and Simulated Annealing
MSE	Mean Square Error
NARMA-L2	Nonlinear Auto Regressive Moving Average-L2

NARXnets	Nonlinear Auto Regressive with eXogenous input networks
NSCA	Non-linear Sine Cosine Algorithm
NSGA-II	Non-dominated Sorting Genetic Algorithm II
OBL	Opposition Based Learning
PI	Proportional Integral
PI+DF	Proportional Integral with Differential Filter
PID	Proportional Integral Derivative
PIDA	Proportional Integral Derivative Acceleration
PIDD2	Proportional Integral Derivative Double Derivative
PNN	Probabilistic Neural Network
PO	Political Optimization
PR	Procreation Rate
PSE	Power Series Expansion
PSO	Particle Swarm Optimization
RAM	Random Access Memory
RMSE	Root Mean Square Error
SCA	Sine Cosine Algorithm
SFS	Stochastic Fractal Search
SNFOPID	Single Neuron Fractional Order Proportional Integral Derivative
SOMA	Self Organizing Migrating Algorithm
SPID	Sigmoid Proportional Integral Derivative
SQP	Sequential Quadratic Programming
SSE	Sum of Squared Error
SSO	Salp-Swarm Optimization
SSW	Sum of Squared Weights
TID	Tilt Integral Derivative
TLBO	Teaching Learning Based Optimization
VBR	Variable Bit Rate
VURPSO	Velocity Update Relaxation Particle Swarm Optimization
WCO	World Cup Optimization
WOA	Whale Optimization Algorithm
ZLG	Zwee Lee Gaing

ZN	Ziegler Nicholas

Chapter 1

INTRODUCTION

A control system is a branch of engineering in which the plant/system operation is adjusted according to the user commands. The plant/system vary from simple temperature control, liquid level inside a tank to complex power plant, and industrial boilers. When designing the controller for a given plant engineers often come across various problems starting from obtaining plant model, external disturbances, non-linearity within system modules etc. Developing an ideal controller for a system is a challenging task and requires intricate mathematical procedures. Based on the procedure used for controller design different control strategies were evolved such as classical, modern, robust, optimal, and intelligent control etc. However, irrespective of the strategy used, the goal of designer is to get the desired response as good as ideal response under external disturbances and variations in the system operation. The purpose of any control system is to provide better industrial and automated systems for the benefit of the society.

The control theory [3] discusses about making the systems whose response is predictable or making them to exhibit desired behavior. The systems may be mechanical, electrical, and electronic or a combination. As a result, the control theory has become interdisciplinary subject and frequently used in various branches of engineering such as electrical, mechanical, robotics, chemical, and aerospace. The foundation upon which the linear system theory establishes a cause-and-effect relationship for various system components serves as the framework for the analysis of the system. The components that make up the system configuration are connected through the control system [4], which produces the required system response. Consequently, the block in Figure 1.1 can be used

to represent a part of a process system that has to be regulated. This shows the input-output process's cause-and-effect relationship. The input signal is processed to produce an output signal, frequently with power amplification.

The control systems are broadly classified into two types known as open loop and closed loop. Each of the type has its own advantage. The open loop methods require very less components to build and are stable. But suffers when external disturbances or noise effects the system operation. Examples for open loop systems are timer based systems like washing machines , dryers, volume and brightness control in television etc. Figure 1.2 depicts an open-loop control system where the controller or control actuator is employed to get the desired response.

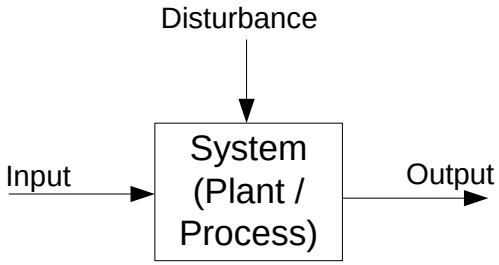

Figure 1.1 System to be controlled

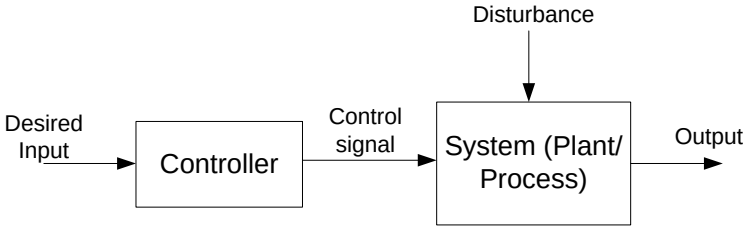

Figure 1.2 Open loop control system

The closed loop control system produces the control signal that drives the actual system using the error signal that is generated by comparing the desired output with the actual/measured output. These systems involves the concept of feedback which creates a closed loop between input and output. This makes the closed loop systems more robust and prone to external disturbances and noise. Examples for closed loop control systems are robotic arms, air conditioners, microwave ovens, and water level controllers for a tank etc.

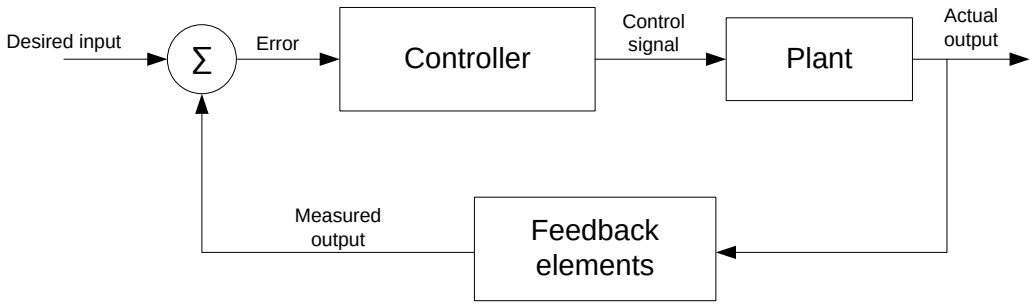

Figure 1.3 Closed loop control system

Figure 1.3 shows basic block diagram of closed-loop/feedback control system. The feedback makes the overall system stable and less prone to external disturbances at the cost of increased complexity in controller architecture. This type of relationship between the actual output and desired or reference input is frequently used in feedback control [5] to get the desired response from the system. The importance of control system has grown multitude because of increased complexity in the systems and new challenges faced by the designers to get the optimum performance from the system.

1.1 Literature survey

Nowadays, researchers are trying to improve the performance of proportional integral derivative (PID) controllers using advanced mathematical concepts. The traditional PID controllers use integer order calculus for the controller design. This gives the controller designers only three degrees of freedom. On the other hand, designing PID controllers using fractional calculus provides two extra degrees of freedom (integral and differential order). Such controllers are called fractional order proportional integral derivative (FOPID) controllers [15]. The FOPID controllers comprise five tuning parameters known as proportional gain (K_p), integral gain (K_i), derivative gain (K_d), order of integration (λ), and order of differentiation (μ). Because of additional tuning parameters (λ and μ), FOPID controllers can produce better response over the existing PID techniques [16].

The equations governing the operation of PID and FOPID controllers in frequency domain are given by

$$c(s) = K_p * e(s) + \frac{K_i}{s} * e(s) + K_d * s * e(s) \tag{1.1}$$

$$c(s) = K_p * e(s) + \frac{K_i}{s^\lambda} * e(s) + K_d * s^\mu * e(s) \tag{1.2}$$

where c(s) is the output of controller, e(s) represents error signal. To efficiently tune the PID and FOPID controller parameters($K_p, K_i, K_d, \lambda,$ and μ) various techniques have been used in the literature. In the following sections, a brief overview of methodologies used in the literature to design PID, FOPID, and neural network based controllers is discussed.

1.1.1 Fractional order systems and control

The fractional order calculus is used to model the physical phenomena more accurately than integer calculus [17, 18]. Examples of physical phenomena which require fractional calculus are heat flow inside a solid metal [15], electromagnetic waves inside lossy transmission lines [19] etc. The importance for fractional order systems has been increased because of availability of various computing methods [20–22]. A brief theory related to fractional calculus and fractional order systems is given in appendix A.

To design the fractional order controllers for different systems various analytical techniques were used in the literature. There are three types of design methods are found from the literature. A tilt integral derivative (TID) controller was proposed in [23] where the proportional component of the controller is replaced by a component having transfer function $'s'$ raised to power of $-\frac{1}{n}$ Where $'s'$ represents complex frequency of system and $n \, \epsilon \, \mathbb{R}$. The controller resulted simpler tuning than PID. Moreover, the TID produced better closed loop response in terms of disturbance rejection and averse to parameter variations in the plant. Oustaloop and group extensively studied the fractional order systems and experimented on control strategies. As a consolidation of their research they developed CRONE (Commande Robuste dordre Non Entier translated in English to Non-integer-order Robust Control) [24]. Different variations of CRONE controllers can be found in [25,26] which are based on frequency domain representation of fractional systems. In the thesis, to approximate the fractional order systems Oustaloop technique was used. A frequency domain approach to design fractional $PI^\lambda D^\mu$ was proposed in [27, 28]. The

traditional lead-lag compensator is extended to fractional lead-lag compensator in [29] and their auto-tuning techniques are mentioned in [30]. A brief review of different tuning techniques used for fractional order controllers was mentioned in [31].

The procedures mentioned in the literature also showed that the fractional order PID controllers produce better response than the classical PID controllers. Most of the analytical methods mentioned in the literature gives us an approximate solution to the fractional controller design problem. Since fractional systems are infinite order systems getting an optimal solution using analytical methods is a challenging task. Alternative methods such as soft computing techniques can be used to get optimum solution.

In the literature, frequency domain based design techniques are mentioned for the design of fractional order controllers. Since the system behavior can be better represented in the frequency domain most of the classical methodologies used this representation. Therefore, time domain based methodologies can be investigated in the design of fractional order controllers.

1.1.2 Optimization of fractional order controllers using meta-heuristic algorithms

Different types of algorithms are used in the literature to perform the optimization of controller parameters. These techniques include traditional approaches like gradient based optimization algorithms which require calculation of gradient of objective function. The problem with these methods is if the objective/cost function is discontinuous then the gradient based methods won't work properly. Moreover if the objective function is multi-modal in nature then the these algorithm may trap in local minima.

On the other side, evolutionary and meta-heuristic algorithms perform better than traditional gradient based methods for the wide variety of objective functions. These algorithms are generalized and can be applied to solve any global optimization problem. Since the design of fractional order controller complex problem, it can be formed as a global optimization problem and solve using evolutionary computing and met-heuristic algorithms.

Evolutionary computing algorithms were used in the design of PID/FOPID controllers for different types of systems such as chemical process control, motor control

power regulator systems, networked control systems etc. In the thesis two plants (DC motor and AVR (Automatic Voltage Regulator) system) are considered to test the implemented methods. Since DC motors and AVR systems are well studied in the literature and moreover different controller mechanisms are applied on these plants. Therefore, it is convenient to compare the proposed techniques with existing state of the art techniques on common system.

Genetic algorithm (GA) was used to identify the optimum parameters of PID controller which are used in chemical processes [32, 33]. A novel optimization algorithm is proposed by hybridizing improved electromagnetism algorithm and genetic algorithm (IEMGA) and it is used for the tuning of FOPID controller for a second order system [34]. The authors compared the performances of PID controller and FOPID controller and showed FOPID controller produced better response. Self organizing migrating algorithm (SOMA) is used to optimize the fractional order controller parameters [35]. In this paper the authors compared the performance of classical optimization scheme and meta-heuristic algorithm. Interestingly, the results showed that both methods produced similar optimum response for the system. Particle swarm optimization (PSO) algorithm was used to tune the FOPID controller parameters [36] for two different processes. The results indicated that the FOPID controllers significantly reduced the large overshoot of the systems. The authors in the paper [37] used modified invasive weed optimization algortihm to tune the controller parameters. A fractional order vehicle suspension system controller was developed using an evolutionary algorithm [37].

A delayed first order system is controlled using fractional PID controller [38], in this paper the authors used genetic algorithm to optimize the controller response. Cauchy Mutated Artificial Bee Colony (C-ABC) algorithm was used to identify fractional $PI^{\lambda}d^{\mu}$ controller [39]. A novel hybrid control methodology which is a combination of fuzzy logic and fractional PID [40]. Fruit fly optimization algorithm (FOA) was used to identify the optimum controller parameters. The authors mentioned that fractional fuzzy control given better results than the traditional fuzzy PID control. An imperialist competitive algorithm (ICA) was used in [41] to control the inter connected power systems in three locations. The paper indicated that use of FOPID controller in the system produced smoother response with reduced oscillations in the steady state. The fractional calculus

introduced in the particle swarm optimization algorithm to update the velocity os particles and named it as fractional particle swarm optimization based memetic algorithm (FPSOMA) [42]. The paper used FPSOMA algorithm for the design of FOPID controller for trajectory control applications. The PSO algorithm is improved with adaptive weight adjustment strategy and used for FOPID controller response optimization [43]. Gases Brownian Motion Optimization (GBMO) algorithm is used to optimize the response of hybrid fuzzy-fopid controller for load frequency control in power systems [44]. To control a pump storage unit an FOPID controller is proposed and its parameters are optimized using Caushy and Gaussian mutated gravitational search algorithm (CGGSA). A novel model refence adaptive control (MRAC) based fractional order PID controller was developed for different systems [45] whose parameters are optimized using Moth-flame optimization algorithm (MOA). Modifed grey wolf optimizer (MGWO) algorithm is used for the fractional order PID controller with derivative filter (FOPIDD) design for automatic Gain Control (AGC) in electric vehicle power systems [46]. To control the movement of robotic arm FOPID controllers are used whose parameters are optimized using colliding bodies optimization algorithm [47].

1.1.3 Optimization of fractional order PID controllers for DC motor and AVR system

The fractional order controllers can be applied to any system where the classical PID controller fit in. FOPID controllers are extensively used in various electrical circuits and systems such as DC-DC converters, synchronous and asynchronous motor drivers, DC motor controllers, load-frequency controller circuits, signal filters, Magnetic levitation systems etc. Even though the number of applications are innumerable, two types of plants are commonly found in the literature, one is electrical motor and the other is AVR system. Since these systems are used as benchmark systems to evaluate the fractional order controllers, the thesis also used the same systems. Moreover, it gives the additional advantage that the proposed methodologies can be compared against wide range of techniques applied on these benchmark systems. This section briefly discusses the state of the art literature related to these plants.

Several fractional order PID controllers for managing the speed of a separately ex-

cited DC motor have been proposed in the literature. optimization of PID controller variables using a genetic algorithm (GA) was studied in [48].The Grey Wolf Optimization (GWO) technique is used to create the best controller for a DC motor [49]. Correspondingly robust analysis for controlling DC motor speed based on GWO was presented in [50]. PID controller parameter identification using IWO (invasive weed optimization) and stochastic fractal search (SFS) algorithm was discussed in [51, 52]. In Recent times, several fractional order PID controllers were proposed using multitude of optimization algorithms for DC motor. The algorithms include Artificial bee colony (ABC) algorithm [53], jaya optimization algorithm(JOA) [54], salp swarm optimization algorithm (SSA) [55]. A fractional order PID controller parameter tuning method using PSO and constrained PSO to regulate the speed of DC motor [56]. In the paper the authors showed that FOPID controllers produced better response and reduced overshoot compared to the traditional PID controller. A GWO tuned FOPID controller for adjusting DC motor speed was suggested, coupled with rigorous analysis in [1]. Recently, based on chaotic theory, ChASO (chaotic atom search optimization) and ASO based FOPID/PID controller design was discussed [2]. The study showed that introduction of chaotic maps improved the original ASO algorithm. Hybridization of Manta ray foraging optimization and simulated annealing (MRFO-SA) techniques incorporated with OBL to find the optimum parameters of FOPID controller for DC motor speed control [57]. The hybrid algorithm produced better response for DC motor. Opposition based henry gas solubility optimization(HGSO) [58] and harris hawks optimization (HHO) [59] algorithms were used to find the optimum parameters of PID controller. Correspondingly fractional PID controller developed in [60] for DC motor using the HHO algorithm and improved the existing performances. An offline optimization technique to PI controller for DC motor speed control using symbiotic organisms search algorithm was mentioned in [61].The authors in [62] optimized PI+DF (proportional integral with differential filter) controller using SFS algorithm. In this paper, the authors improved performance of DC servo system by incorporating anti-wind up mechanism and derivative filter into the traditional PI controller.

Tuning fractional controllers for AVR system involve solving under-defined fractional differential equations. Therefore, it can be posed as an optimization problem and solved with various meta-heuristic algorithms. In case of controller optimization for AVR

system, the cost function plays an important role. The authors in [63] used a complex cost function that involves time domain and frequency domain parameters of the system along with PSO algorithm to optimize the FOPID controller parameters. The proposed method reduced overshoot to zero, but large values of rise time and settling time are observed. Different schemes are used to tune the parameters of the fractional order model reference adaptive controller(FOMRAC), FOPID, and PID controller variables in [64]. The authors used sequential quadratic programming (SQP), particle swarm optimization (PSO) and genetic algorithm (GA) to identify the controller parameters and showed that the fractional order PID controller performed better than all other techniques. Chaotic maps influenced multi-objective optimization problem using a non-dominated sorting genetic algorithm II (NSGA-II) was utilized in [65] to tune the FOPID controller for AVR system. The authors minimized multiple cost functions to identify best values for the controller and showed that the fractional order controllers can produce better rise time and settling time.

Besides these algorithms, various meta-heuristic optimization algorithms are used in the literature. To improve the dynamic response and stability of the FOPID controllers, salp-swarm optimization (SSO) based tuning method was presented in [66]. A multi-objective method based on extremal optimization algorithm was proposed in [67] and showed that the algorithm produces better tuning parameters than NSGA-II for AVR system. The ant swarm algorithm is hybridized with chaotic maps to optimize FOPID controller for AVR system [68]. The authors showed that the chaotic maps identified better FOPID controller parameters than the previous methods. In [69], the authors tuned the PID controller using cuckoo search (CS) algorithm. Also, a new objective function was defined with a weighted combination of integral of time multiplied absolute error (ITAE), overshoot, rise time, and steady state error. FOPID controller tuning using cyclic exchange neighborhood with chaos-artificial bee colony (CNC-ABC) [70] achieved better performance of AVR system than the existing PID and FOPID controllers. Comparison of AVR system controllers using teaching learning based optimization (TLBO), harmony search algorithm (HSA), and local uni-modal sampling (LUS) method was presented in [71]. Sine?cosine algorithm (SCA) based FOPID controller design discussed in [72] and improved the FOPID controller performance. The authors in [73] used chaotic yellow saddle goatfish algorithm (CYSGA) to find optimum parameters for FOPID con-

troller. In the paper, the authors used a new objective function with a combination of ITAE, overshoot, steady state error, and settling time. The chaotic maps are used in the literature to improve the search capability of existing algorithms. To improve the local search capability of differential evolution algorithm, chaotic maps are used in [74]. A combination of two meta-heuristic algorithms simulated annealing (SA) and manta ray foraging optimization (MRFO) were used in [75] to tune PID, proportional integral derivative double derivative (PIDD2), and FOPID controllers. The authors confirmed that FOPID controllers generate better rise time and settling time values, whereas PID controllers are good for set point tracking.

Various methods are proposed in the literature to tune PI/PID controllers for AVR system. PSO algorithm is used in [76] to optimize the PID controller variables for AVR system. In the paper the authors defined a new objective function based on figure of demerit for efficient tuning. Chaotic map based algorithms such as CAS algorithm and chaotic optimization approach are use in [77, 78] for AVR system. An online tuning method for PID controller using two different versions of PSO algorithm called velocity update relaxation (VURPSO) and novel position, velocity updating strategy and craziness (CRPSO) was mentioned in [79]. A comparative analysis performed on PSO, ABC and DE (differential evolution) algorithms [80] for the design of PID controller to AVR system. The authors showed among the algorithms ABC performed better. To efficiently tune the PID controller for AVR system different meta-heuristic algorithms such as simplified particle swarm optimization [81], local uni-modal sampling [82], symbiotic organisms search algorithm [83], cuckoo search algorithm [84], whale optimization algorithm [85], Improved kidney-inspired algorithm [86], grasshopper optimization algorithm [87]. Various Hybrid meta-heuristic optimization algorithms are also used such as hybrid GA-BF (Bacterial Foraging) optimization [88], Taguchi combined genetic algorithm [89], and Ant Colony Optimization with constrained Nelder-Mead algorithm [90] for PID controller response optimization for AVR system.

A detailed analysis of literature survey for the DC motor and AVR system finds some observations. All the methods mentioned in the literature uses objective function as minimization criteria to find the optimum parameters for the controller. Therefore, a better objective function can generate good controller parameters.

1.1.4 Neural networks for intelligent control

In the literature, various systems were identified using neural networks. The authors in [91] used NARXnets (Nonlinear Auto Regressive with eXogenous input networks) to forecast daily solar direct radiation for the photo voltaic management systems. The developed solution forecasts the direct solar radiation for a surface. The authors in [92] presented an overview of properties and computational capabilities of the NARX neural networks along with mathematical modeling. The paper [93] used NARXnets for multi step time series prediction for time delay systems for real-time VBR(variable bitrate) video traffic time series. Different types of training algorithms used to train these neural networks were discussed briefly in [94]. The paper [95] proposed an online adaptive control strategy for controlling the speed of separately excited DC motor with Artificial Neural Networks (ANNs). Recently a comparison between Proportional Integral Derivative (PID) controller and ANN based controller for controlling speed of DC motor was mentioned by [96] and the author shows that the ANN-based controller produces less overshoot and settling time compared to the PID controller. For better estimation of DC motor position and velocity values, ANN-based design was proposed in [97]. In this paper, the authors proved that the ANN-based estimator provides better values of velocity and position of DC motor than the traditional state observer based estimator. [98] designed an IMC (internal model control) based adaptive neural network controller for controlling the speed of DC motor in real-time. [99] used artificial neural networks for estimating the power generation efficiency of permanent magnet synchronous generator. The authors proved that the neural networks can be used for system identification and data generation.

A two stage artificial neural network based controller was designed for DC motor speed controller [100]. In the paper, the authors used two neural networks one for estimation of motor speed and another for generation of control signal generation. NARMA-L2 (Non-linear Auto Regressive Moving Average-L2) based neural network controller for separately excited DC motor is mentioned in [101]. A comparative analysis of neuro-fuzzy system, Genetic algorithm and ZN-Tuning based PID controllers was performed in [102]. In the paper, the authors showed that genetic algorithm and neuro fuzzy based systems performed better than the traditional PID controller. A detailed research report on artificial neural networks for DC motor speed control was given in [103]. To control the

angular position of the DC motor a neural network based approach was given in [104]. PI controller parameter tuning technique using neural networks was given in [105].

To tune the parameters of FOPID controller for AVR system neural networks are used in [106]. The authors showed that neural networks are able to optimize controller response better than the existing optimization algorithms. Probabilistic neural networks(PNN) are used in [107] to control the AVR system response. The authors compared performances of PI, PID, and PNN and showed that PNN produced better performance metrics. A neural network predictive control scheme along with imperialist competitive algorithm (ICA) is used for better optimization of AVR system response [108]. To improve the power system stability feed forward neural networks(FFNN) are used in AVR system [109, 110].

From the literature survey it is found that different types of neural networks were used to tune the parameters of PID/FOPID controllers. In addition feed forward neural networks, probabilistic neural networks, and neuro-fuzzy systems were also used to improve the performance of existing controllers. Even though the authors mentioned that neural networks performance is good training neural networks requires model of the system and controller training data. Moreover there are very few studies related to hybrid architectures to mitigate some of these problems. There is a need to investigate whether a single neural network can be used for identification as well as control.

1.2 Gaps identified from the literature survey

To design optimum controller, various methodologies have been discussed in the literature such as proportional integral derivative(PID), fractioanl order PID(FOPID), neural networks, fuzzy, and neuro-fuzzy. Among these techniques PID controllers are commonly used in industrial applications because of its simplicity and ease of implementation. As an extention to the PID controllers FOPID controllers are proposed in the literature to extend the capabilities of PID controllers such as additional degrees of freedom in tuning parameters, increased robustness. Most of the methodologies mentioned in the literature used nature inspired meta-heuristic algorithms to identify the optimum parameters of FOPID/PID controllers. Neural networks and fuzzy based controllers pro-

posed as alternative to traditional controllers where in additional customization is possible related to control strategies.

During the literature survey short comings of different methods are identified and summarized as follows. These gaps form the foundation for identifying research objectives.

- There are no particular rules defined for tuning FOPID/PID controller parameters. It is due to the fact that the requirements vary for different systems. Therefore, the objective functions used for controller optimization should be defined carefully according to the plant/system under operation.
- meta-heuristic algorithms used for the optimization of controllers can be further improved or new algorithms can be proposed to enhance the controller performance.
- In traditional control theory, system identification and controller design are viewed as two separate problems. Using data driven approaches both problems can be solved using single algorithm.
- There is a need to study how neural networks can be used to solve system identification as well as controller design.
- Hybrid controllers such as combination of FOPID and neural networks need to be investigated. These controllers have the advantages of both FOPID and neural network controllers.

1.3 Motivation

Over the past century, the systems have grown multi-fold and involved various components related to electrical, mechanical and electronics. For such systems designing a controller that is robust, ambiguity free, and with superior performance is a challenging task. Design of a good controller requires the knowledge of entire system operation and accurate mathematical models. As the system complexity increases developing the accurate model becomes difficult and in certain cases it is near to impossible because of huge number of system parameters. The traditional controller design techniques are particular and analytical in nature and produced satisfactory results for certain class of systems [6].

With the advances in the computing technologies we can automate the system modeling and subsequent controller design tasks. This reduces the burden on the system designers and they can concentrate more on the system performance related aspects.

Intelligent control is a class of control techniques that use various artificial intelligence computing approaches like neural networks, fuzzy logic, machine learning, and evolutionary computation [7]. As the importance for machine learning is drastically increasing in several fields of engineering we can use the advantage of availability computing power for the design of optimum controllers. The traditional PID controller tuning procedures [8–11] produces semi-optimum parameters. Therefore in order to find optimum or at least near optimum parameters we can use meta-heuristic algorithms and neural networks [12–14]. Studying the combination of traditional FOPID/PID controllers and neural networks can lead to new type of hybrid controllers. Figure1.4 shows the block diagram for the design procedure for intelligent control.

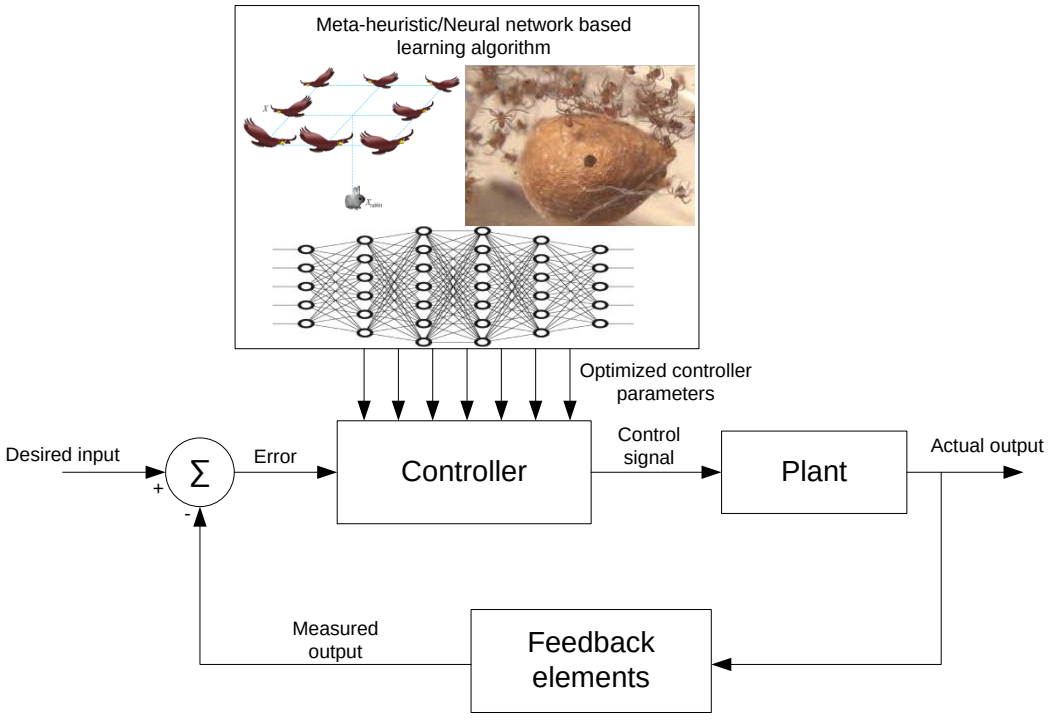

Figure 1.4 Block diagram for intelligent control system

Inspired from the optimization capabilities of meta-heuristic algorithms and universal approximation nature of the neural networks, the primary motivation is to develop procedures to identify optimal parameters for the fractional order proportional integral

and derivative (FOPID)/PID controllers using meta-heuristic algorithms and design hybrid neural network based controllers.

1.4 Problem statement

System identification and controller design is an interesting area where we need to develop the mathematical models of the system by studying the behavior. The controller design for a system requires knowledge of system as well as designing of objective function to design the controller parameters which gives the best response from the system. The most commonly used controller is the proportional, integral and derivative (PID) controller. Extending the concept of PID controllers to fractional calculus resulted in a new type of controllers called fractional order PID (FOPID) controllers. But identification of best parameters for these controllers is a complex problem and it is far more difficult than PID controllers.

The classical control techniques use frequency domain representation of system and based on desired response a suitable compensator is employed. But these techniques are not suitable for multivariate systems. To solve this problem modern control theory uses state space representation of the system. But the design of controller requires solution of set of algebraic equations. The traditional gradient based techniques like simplex and interior point methods can be employed. For these methods to work the system must be continuous or differentiable. If the system is corrupted by noise or external disturbances then these methods may not work. Another way which does not require calculation of derivatives is use of meta-heuristic algorithms and neural networks.

Therefore in the proposed methods meta-heuristic and neural network based techniques are used to identify the optimum controller parameters for a given system. Based on the idea, the problem statement is formulated as mentioned below.

"Development of controller and tuning techniques using meta-heuristic algorithms and neural networks"

1.5 Research Objectives

The research objectives are formed based on the gaps identified in the literature. The summary of the research objectives are mentioned as follows.

- Devise methods to efficiently tune the FOPID/PID controller parameters using meta-heuristic optimization algorithms
- To improve performance of existing meta-heuristic algorithms by incorporating chaotic maps for better tuning of controller parameters
- Design neural network based controllers for identification and control of different systems
- Design hybrid architectures for the controllers using neural networks
- Analyze the performance of different types of controllers for a given system

1.6 Thesis Contributions

The research work present in ensuing chapters of this thesis makes original contribution to the field of fractional PID controller tuning using meta-heuristics and hybrid controller architectures using neural networks. The summary of the contributions are given as follows.

- A type of recurrent neural networks called NARXnets are used to design a Hybrid controller for the speed control of DC motor. The controller is designed by approximating HHO (Harris Hawks Optimization) algorithm tuned FOPID controller using neural networks.
- To verify the operation of FOPID controller, it is practically implemented to control the DC motor speed. For the realization, Dspace 1104 hardware is used along with control desk and MATLAB software.
- Novel architecture based on neural network is proposed for fractional order proportional integral derivative (FOPID) controller. A new optimization algorithm using

chaotic maps, called Chaotic Political Optimization (CHPO) is developed to identify the optimal weights of the neural network. To verify the efficiency of the proposed single neuron FOPID (SNFOPID) controller, a separately excited DC (direct current) motor is considered as plant and its response is optimized. A new cost function is defined based on the performance metrics of the plant and overall system error.

- Optimization of FOPID controller parameters for AVR system using BWO (Black Widow optimization algorithm) was presented. In the second part to further improve the performance of the BWO algorithm, chaotic maps are introduced and applied the algorithm to optimize the AVR system response.

- A procedure for Automatic voltage regulator(AVR) system identification and corresponding controller design procedure was developed using Bayesiaan Regularization NARXnets.

- Novel tuning method proposed for sigmoid proportional, integral, and derivative controller (SPID). The method uses the jellyfish search optimization (JSO) algorithm to identify the optimal parameter for the SPID controller for AVR system.

1.7 Thesis Organization

The rest of the thesis is structured as follows:

Chapter 1 An overview of state of the art controller design techniques, various metheuristic algorithms and neural network based approaches for the design of controllers are discussed. Later, the gaps identified from the literature study, motivation, problem statement, research objectives and thesis contributions are mentioned.

Chapter 2 This chapter mentions design procedure for NARXnet controller for speed control of DC motor using harris hawks optimization algorithm (HHO) tuned FOPID controller as a reference controller. The chapter also includes hardware implementation of FOPID controller for DC motor speed control.

Chapter 3 This chapter proposes a novel architecture for FOPID controller inspired from the architecture of artificial neuron. The parameters of the controller are optimized using chaotic political optimization algorithm.

Chapter 4 This chapter discusses the optimal tuning of FOPID controller parameters for AVR system using black widow optimization (BWO) algorithm and chaotic black widow optimization (ChBWO).

Chapter 5 This chapter discusses a neural network based identification and control scheme for stabilization and control of automatic voltage regulator system. Here, the weights of the neural network are optimized using Bayesian regularization algorithm.

Chapter 6 This chapter presents optimization of sigmoid PID controller parameters for AVR system Jellyfish search optimization (JSO) algorithm.

Chapter 7 Finally, this chapter summarizes the thesis conclusions from the contributions and provides a brief discussion on the direction for future work.

Chapter 2

Design of NARXnet based Fractional-Order PID/PID Controller for Speed Control of DC Motor

In the first part of the chapter, tuning of FOPID controller for speed control of DC motor using HHO (Harris Hawks Optimization) algorithm was presented. In the second part, a type of recurrent neural networks called NARXnets are used to design a hybrid controller for the speed control of DC motor. The controller is designed by approximating HHO algorithm tuned FOPID controller using neural networks. To validate the FOPID controller real-time implementation is discussed at the end of the chapter.

2.1 Introduction

To accurately model the dynamics of a system, it requires good understanding of behavior and important parameters of the system. In traditional methods, the key parameters are identified in the initial phase. Later, a mathematical model is developed based on these parameters. Often the developed model may not accurately predict the dynamics of the system. It happens due to two reasons, primarily the effect of external signals on the system dynamics, the second factor is amount of detail, which is limited by nature of the system. On the other side, artificial neural networks were successfully applied for dynamic system identification and system approximation problems. The advantage of neural networks is, they can model any non-linear function by proper training.

Despite various techniques available for the design of FOPID controller, there is

still scope for research to improve these controllers. Direct implementation of FOPID controllers involves realization of real order integral and differential equations and the realization requires infinite order filters. Therefore, to overcome this problem, we propose a neural network based technique for the design of FOPID controller. The dynamics of FOPID controller can be identified using neural networks. This allows us to accurately model the behavior of these controllers. Since neural networks are adaptive in nature it will be easy for the designer to adjust the weights according to the required controlling action. Another advantage is neural networks perform better under noisy environments than the traditional controllers. This increases the reliability and robustness of the controller.

2.2 Overview of Harris Hawks Optimization (HHO) algorithm

HHO [111] is a meta-heuristic, population-based, gradient-free optimization algorithm developed based on the hunting behavior of hawks. At the beginning, the hawks perch the search area for prey. Once they identify the prey, the hawks confuse the prey and then make it exhausted. The perching behavior of hawks is like exploration and hunting the exhausted prey is like exploitation. Let p be a random number, if $p > 0.5$ then the perching is based on family members else it is based on the position of the prey. This is mathematically modeled using equation (2.1).

$$Z(t+1) = \begin{cases} Z_{rnd}(t).p_1|Z_{rnd}(t) - 2.p_2.Z(t)\ if\ p_{1,2} \geq 0.5 \\ (Z_{rbt}(t) - Z_m(t)).p_3.(L + p_4.(U - L))\ if\ p_{1,2} < 0.5 \end{cases} \tag{2.1}$$

where $Z(t+1)$ represents location of hawks in (t+1) iteration, $Z_{rnd}(t)$ indicates random hawk location, $Z_{rbt}(t)$ is the rabbit location, $Z(t)$ is the present position vector of hawks, $p_1, p_2, p_3, p_4 \epsilon (0, 1)$ represents random variables, the upper and lower bounds are indicated by U and L, and $Z_m(t)$ indicates average position for current population((2.2)).

$$Z_m(t) = \frac{1}{K} \sum_{i=1}^{K} Z_i(t) \tag{2.2}$$

2.2.1 Condition to interchange between exploitation and exploration

The algorithm changes between exploration and exploitation phases based on prey (rabbit) energy. The prey gradually loses energy during the hunting process and this

behavior represented in equation (2.3).

$$N = 2N_0(1 - \frac{t}{T}) \tag{2.3}$$

Here N represents the energy of prey, T is the value of final iteration, and $N_0 \epsilon (-1, 1)$ indicates initial energy of prey. If prey energy $N \geq 1$ then hawks enter into the exploration stage, and when $N < 1$ the hawks enter into the exploitation stage.

2.2.2 Exploitation phase

In this process, Harris hawks perform surprise attacks by targeting the intended prey. Let p represents the probability of escaping for prey then $p < 0.5$ indicates fruitful escape and $p \geq 0.5$ indicates failed to escape before a surprise attack.

Soft besiege

If $p \geq 0.5$ and $|N| \geq 0.5$, the prey has enough energy and it tries to escape by following tricky movements. These moves are modeled with equations (2.4) and (2.5).

$$Z(t+1) = \delta Z(t) - N|Q * Z_{rbt}(t) - Z(t)| \tag{2.4}$$

$$\delta Z(t) = Z_{rbt}(t) - Z(t) \tag{2.5}$$

Where $Q = 2 * (1 - p_5)$ represents the length of rabbit movement.

Hard besiege

If $p \geq 0.5$ and $|N| < 0.5$, the rabbit is tired and it has low N value. The positions of hawks are updated with equation (2.6).

$$Z(t+1) = Z_{rbt}(t) - N|\delta Z(t)| \tag{2.6}$$

2.2.3 Soft besiege using advanced moves

If $|N| \geq 0.5$ but $p < 0.5$, indicates prey has sufficient energy to escape. Therefore the hawks follow soft besiege. In this case the positions of the hawks are updated using the equation shown in (2.7)

$$A = Z_{rbt}(t) - N|Q * Z_{rbt}(t) - Z(t)| \tag{2.7}$$

The hawks will move according to L-based patterns using the rule given in (2.8)

$$B = A + D \times L(C) \tag{2.8}$$

Where C indicates problem dimension and D represents a random vector of size $1 \times C$. Correspondingly, the Leavy function L can be obtained from equation (2.9)

$$L(z) = 0.01 \times \frac{\mu \times \rho}{|u|^{\frac{1}{\gamma}}}, \rho = \left(\frac{\Gamma(1+\gamma) \times sin(\frac{\pi\gamma}{2})}{\Gamma(\frac{1+\gamma}{2}) \times \gamma \times 2^{(\frac{\gamma-1}{2})}} \right)^{\frac{1}{\gamma}} \tag{2.9}$$

The random values $u, \mu\epsilon(0,1)$ and $\gamma = 1.5$. The final equation to update the positions of hawk during soft besiege is given in (2.10).

$$Z(t+1) = \begin{cases} A, \, if \; F(A) < F(Z(t)) \\ B, \, if \; F(B) < F(Z(t)) \end{cases} \tag{2.10}$$

Where $F(.)$ represents the objective or cost function.

2.2.4 Hard besiege using advanced moves

Hard besiege will be performed if $|N| < 0.5$ and $p < 0.5$. In this case, the prey may not contain sufficient energy to escape. This is modeled by using equation (2.10) Where

$$A = Z_{rbt}(t) - N|Q * Z_{rbt}(t) - Z_m(t)| \text{ and } B = A + D \times L(C) \tag{2.11}$$

Various working stages of HHO algorithm is represented in the form of block diagram as shown in the figure 2.1.

2.3 Modeling of DC motor

An externally excited DC motor model was considered for speed control through armature voltage. The equivalent circuit of the DC motor is shown in figure 2.2. The parameters identified for modeling the DC motor were mentioned below.

R_a : Armature resistance(Ω), L_a :Inductance of armature (H),

i_a : Current in armature(A), i_f :Current due to field(A),

e_a : Applied voltage(V), e_b : Back e.m.f(V),

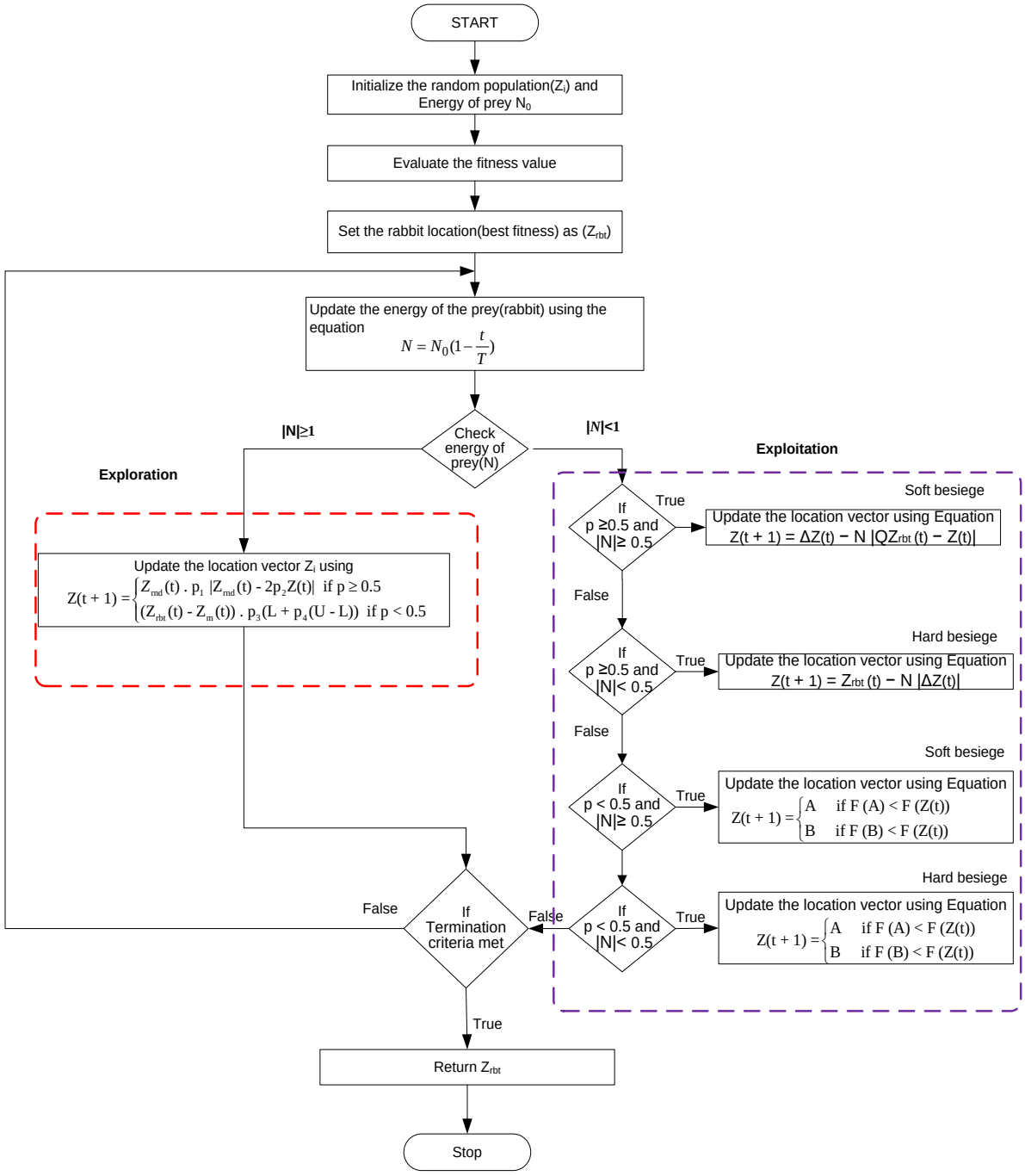

Figure 2.1 HHO algorithm

T : Motor torque$(N.m)$, ω : Angular velocity of motor shaft(rad/s),

J : Inertia torque$(kg.m^2)$, K_b : e.m.f constant$(V.s/rad)$,

K : Motor torque constant$(N.m/A)$, B : Motor friction constant$(N.m.s/rad)$

A mathematical model for DC motor was formulated from the identified parameters. The equation governing the electrical behavior of armature controlled dc motor was given

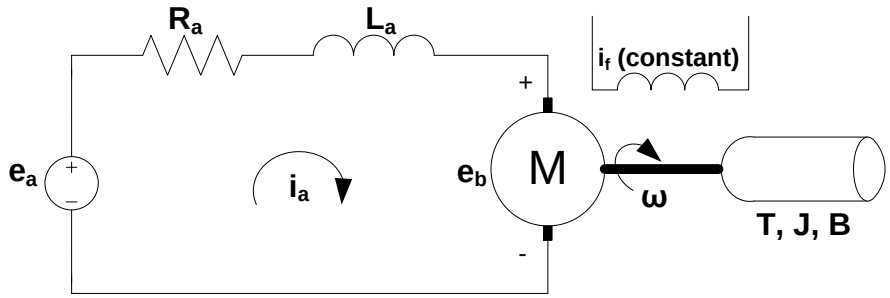

Figure 2.2 Electrical equivalent of DC motor

Table 2.1 Values of modeling parameters of DC motor [1, 2]

S.no	Parameter	Value
1.	R_a	0.4 Ω
2.	L_a	2.7 H
3.	J	0.0004 $kg.m^2$
4.	B	0.0692 $(N.m.s/rad)$
5.	K	0.0488 $(N.m/A)$
6.	K_b	$(V.s/rad)$

by (2.12).

$$e_a = R_a i_a + L_a \frac{\partial i_a}{\partial t} + e_b \tag{2.12}$$

Because of the current flow in the motor armature, torque is developed and is equal to sum of inertial torque and frictional torque as described in (2.13).

$$T = J\frac{\partial \omega}{\partial t} + B\omega = k i_a \tag{2.13}$$

The back emf voltage is proportional to angular velocity (ω) of the motor and was related by (2.14)

$$e_b = k_b \omega = k_b \frac{\partial \theta}{\partial t} \quad (since\ \omega = \frac{\partial \theta}{\partial t}) \tag{2.14}$$

Application of Laplace transforms to the equations (2.12) - (2.14) produce the corresponding transform domain equations (2.15) - (2.18). Assuming zero initial conditions, then

$$e_a = R_a I_a(s) + sL_a I_a(s) + e_b(s) = (R_a + sL_a) I_a(s) + e_b(s) \tag{2.15}$$

$$e_b = k_b \omega(s) = s k_b \theta(s) \tag{2.16}$$

$$T(s) = (Js + B)\omega(s) = kI_a(s) \tag{2.17}$$

The transfer function of the DC motor was given by (2.18)

$$G(s) = \frac{\omega(s)}{E_a(s)} = \frac{k}{(sL_a + R_a)(Js + B) + k_b k} \tag{2.18}$$

Substituting the values of table 2.1 parameters in (2.18), the final transfer function of DC motor was identified and given in (2.19).

$$G(s) = \frac{\omega(s)}{E_a(s)} = \frac{0.015}{(2.7s + 0.4)(0.0004s + 0.0022) + 0.00075} \tag{2.19}$$

Using the equation (2.19) a block diagram was created for the DC motor and implemented in Simulink. The corresponding diagram was shown in figure 2.3. Using the transfer

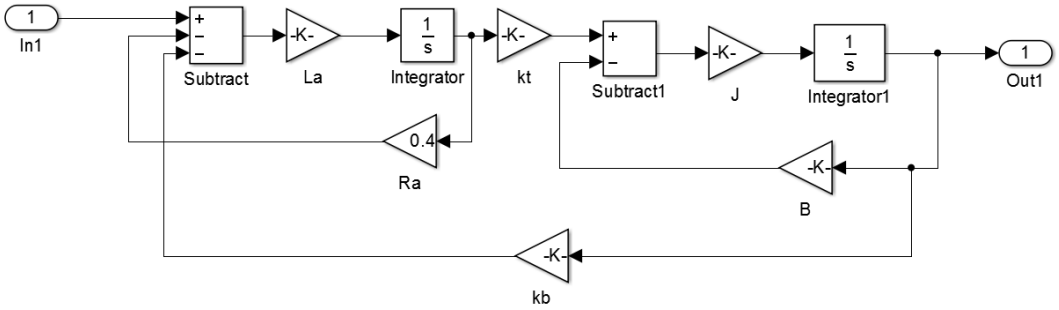

Figure 2.3 DC motor transfer function using Simulink

function mentioned in equation (2.19) the motor is simulated with the help of MATLAB software. The open loop response of the simulated DC motor is plotted and shown in the figure 2.4. From, the open loop response, it is found that the DC motor has 7.83s rise time, 14.09s settling time, with 0 overshoot, and absolute steady state error value 8.2 units. From the responses we can observe that the system has high values of rise time, settling time, and steady state error. Therefore to optimize the system response a controller is required.

2.4 Fractional order PID / PID controller

Fractional calculus deals with the evaluation of real order integro-differential equations. The traditional PID controllers use integer order calculus. Expanding the integer calculus to fractional calculus results in new type of controllers called fractional order PID

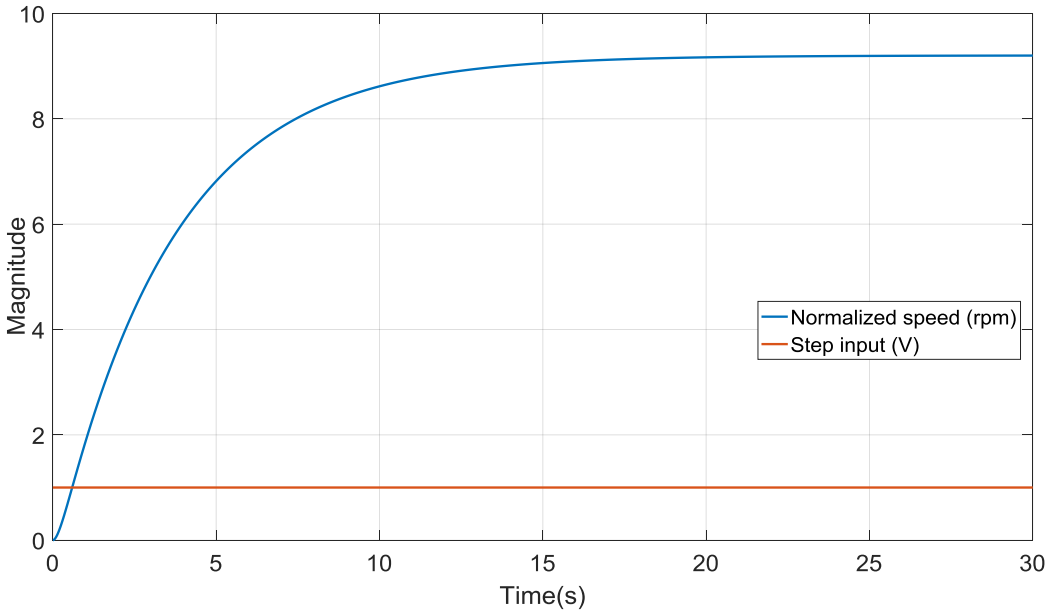

Figure 2.4 Open-loop response of DC motor

(FOPID) controllers. These controllers (FOPID) contain two additional tuning parameters (λ, μ) than integer order or normal PID controllers (IOPID). The general time-domain equation governing the operation of FOPID controller was given in (2.20).

$$r(t) = K_p er(t) + K_i D_t^{-\lambda} er(t) + K_d D_t^{-\mu} er(t) \tag{2.20}$$

Where $r(t)$ is the desired signal, $er(t)$ indicates error signal and $\lambda, \mu \, \epsilon \, (0, 2)$. The terms $D_t^{-\lambda}$ and $D_t^{-\mu}$ are related to fractional calculus [6] which are defined as

$${}_aD_t^{\alpha} = \begin{cases} \frac{d^{\alpha}}{dt^{\alpha}}, & R(\alpha) > 0, \\ 1, & R(\alpha) = 0, \\ \int_a^t (d\tau)^{-\alpha} & R(\alpha) < 0 \end{cases} \tag{2.21}$$

Where a, t are limits and α is the order of differ-integral term D. Another commonly used definition for fractional calculus operator is Grünwald-Letnikov definition [6] given by

$${}_aD_t^{\alpha} f(t) = \lim_{x \to \infty} \frac{1}{h^{\alpha}} \sum_{k=0}^{[\frac{t-a}{h}]} (-1)^k \binom{\alpha}{k} f(t - kh) \tag{2.22}$$

Where h is the computation step size. Application of Laplace transform to the equation (2.20) results (2.23)

$$R(s) = (K_p + \frac{K_i}{s^{\lambda}} + K_d s^{\mu}) * E(s) \tag{2.23}$$

Where R(s) and E(s) are Laplace domain representation of r(t) and er(t) respectively. Then the corresponding fractional order controller is given by (2.24).

$$G_c(s) = K_p + \frac{K_i}{s^\lambda} + K_d s^\mu \tag{2.24}$$

From equation (2.24) if the values of λ and μ are equal to 1, then the fractional order controller represents the integer order controller. Different forms of the fractional controller were shown in figure 2.5. From the figure, it is observed that all other integer-order controllers are special cases of FOPID controller.

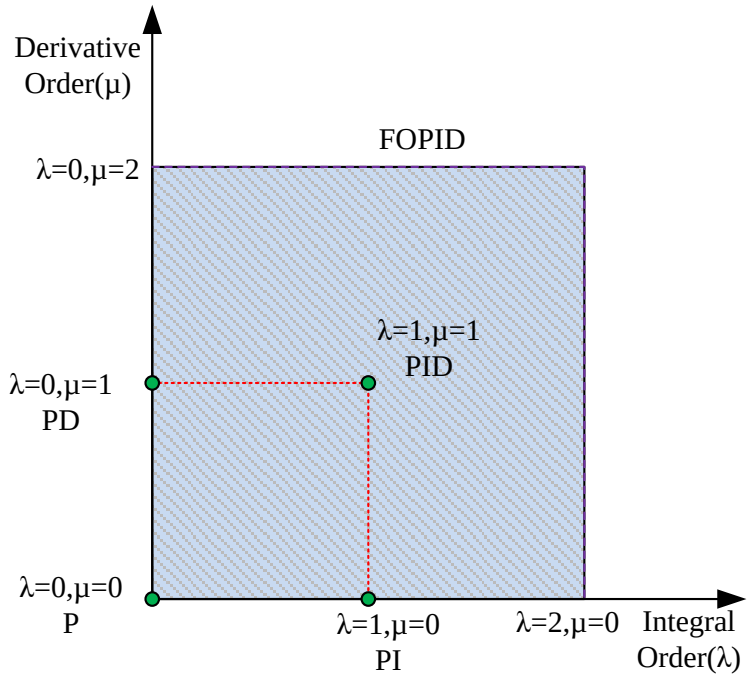

Figure 2.5 Operating region of FOPID controller

Designing a fractional order controller involves solving under defined fractional integro-differential equations. It involves complicated mathematical equations and lengthy process. The solution to optimal controller can also be found using soft-computing techniques like meta-heuristic algorithms. In the thesis different meta-heuristic algorithms are used for optimum fractional order controller design.

2.5 HHO based fractional PID controller design

To optimize DC motor fesponse, fractional order PID controller is considered because of its inherent advantages like additional tuning parameters. To tune the controller gains HHO algorithm is used. The block diagram of the complete system was shown in figure 2.6. For each iteration of the algorithm, the error value for the output of DC motor was calculated and substituted back into the corresponding objective function. The parameter error values were adjusted according to the described procedure in the optimization algorithm. The above process is repeated until the terminating criterion was met. Finally the tuned values are used to implement the FOPID controller.

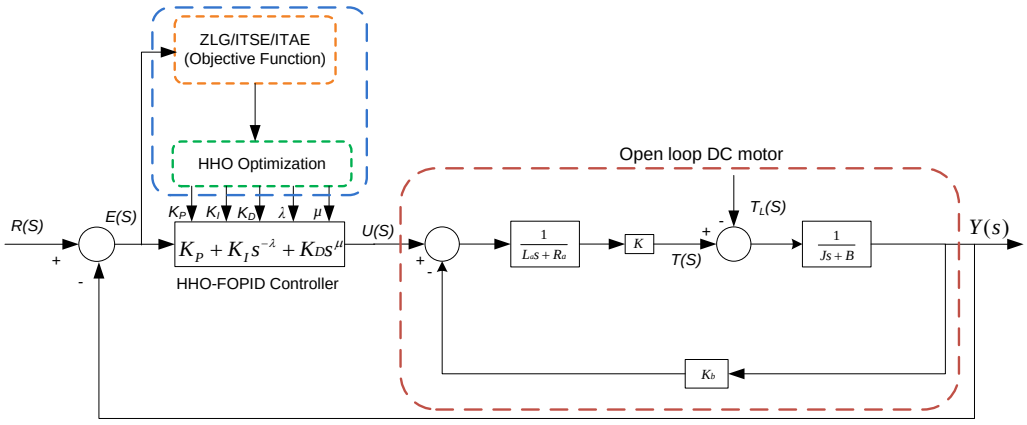

Figure 2.6 Proposed HHO-FOPID controller block diagram

2.5.1 objective function

To find the best tuning parameters for the controller a good objective function is required. Different objective functions(ITSE, ITAE, and ZLG [76] (Zwe Lee Gaing)) were proposed in the literture for proper tuning of PID controller parameters. Also, to investigate the effectiveness of the proposed algorithm these objective functions are used. Therefore to optimize controller parameters K_p, K_i, K_d, λ and μ, HHO algorithm is applied on the mentioned objective functions.

The equations for ITAE, ITSE, and ZLG functions were shown in equations (2.25), (2.26), and (2.27). The proportional, integral and derivative parameters (Kp, Ki, and Kd) were limited to the range [0, 20] and fractional powers (λ and μ) were limited to the

range [0, 2].

$$ITAE = \int 0at.|t.er(t)|dt \tag{2.25}$$

$$ITAE = \int 0at.er^2(t)dt \tag{2.26}$$

$$ZLG = (1 - er^{-\gamma}) * (O_p + E_{ss}) + er^{-\gamma} * (t_s - t_r) \tag{2.27}$$

Where E_{ss}, t_r, Op, ess, and ts indicates steady-state error, rise time, maximum overshoot, and settling time respectively and the value of γ is equal to 1.0 [76].

2.6 Simulation results of HHO-FOPID controller

All the simulations were performed using MATLAB/Simulink (with FOMCON toolbox) version 8.1a on the computer with Intel i5 processor @ 3.00GHz and 8GB RAM. For optimization algorithms, the population size is taken as 50 and the number of iterations performed was 30.

To verify the efficacy of HHO-PID and HHO-FOPID controllers, different controllers are compared by considering the same plant (DC motor). The FOPID controllers are implemented with the frequency range [0Hz, 1000Hz]. The range is selected based on the frequency response of the DC motor using Bode plots. The step reactions of proposed controllers were indicated in figure 2.7 and figure 2.8. Comparison of step response of different FOPID controllers is given in figure 2.9 and Bode plot of proposed FOPID-controller is given in figure 2.10. The tuned parameter (K_p, K_i, K_d, λ, μ) values of various controllers were shown in Table. 2.3. Also the proposed algorithm was compared for steady-state error, rise time, overshoot and settling time with other FOPID/PID controllers in Table 2.3.

The results clearly show that HHO-PID and HHO-FOPID controllers are performing better than GWO-FOPID/PID [50], SFS-PID [52] and IWO-PID [51] controllers. In the case of PID controllers, all the controllers are producing nearly the same values with very slight deviation. For FOPID controllers the settling time was improved from 0.05406s to 0.0373s, the steady-state error reduced from 3.446E-03 to 3.17E-04 whereas the rise time was slightly increased from 1.79E-02s to 1.92E-02s. To further investigate, ITAE, ITSE and ZLG values are calculated (shown in table 2.2) for all the controllers and it is

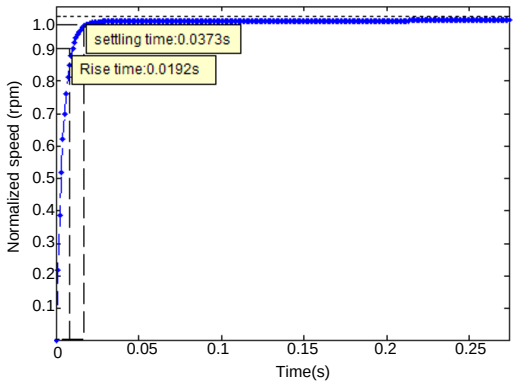

Figure 2.7 HHO-FOPID step response

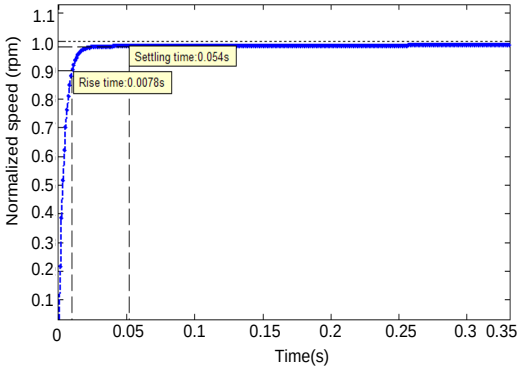

Figure 2.8 HHO-FOPID controller response

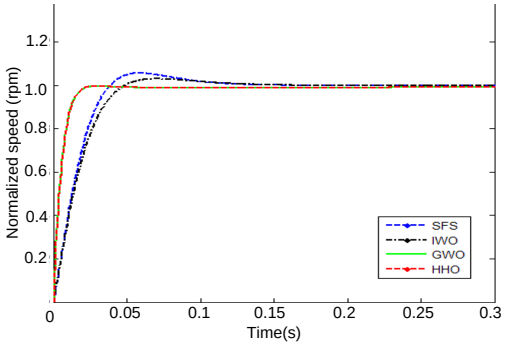

Figure 2.9 Comparison of different FOPID controllers

Figure 2.10 Bode plot for HHO-FOPID controller

Table 2.2 Comparison of objective function values for different controllers

Controller	ITSE	ZLG	ITAE
HHO-PID (Proposed)	4.82E-05	0.016704	0.0626
GWO-PID [50]	4.82E-05	0.016707	0.0654
SFS-PID [52]	4.84E-05	0.017032	0.0634
IWO-PID [51]	4.82E-05	0.017155	0.0985
HHO-FOPID	3.18E-05	0.00218	0.0819
GWO-FOPID [50]	3.18E-05	0.00256	0.0819
SFS-FOPID [52]	3.25E-05	0.00491	0.0843
IWO-FOPID [51]	4.26E-05	0.00361	0.0956

clear from the simulated results that the proposed controllers have a little edge over the others. It is also found that GWO based controllers performance is very nearer to the HHO based controllers and SWS and IWO based fractional controllers are producing little

Table 2.3 Comparison of controller FOPID/PID parameters

Controller	K_p	K_i	K_d	λ	μ	Rise time(s)	Settling time(s)	Overshoot(%)	E_{ss}
HHO-PID (Proposed)	20	19.71	20	1	1	1.79E-02	5.40E-02	0	3.446E-03
GWO-PID [50]	20	19.716	20	1	1	1.79E-02	5.40E-02	0	3.448E-03
SFS-PID [52]	20	19.026	20	1	1	1.79E-02	5.61E-02	0	3.448E-03
IWO-PID [51]	20	19.71	20	1	1	1.79E-02	5.50E-02	0	3.446E-03
HHO-FOPID	20	20	20	0.508	0.931	1.92E-02	3.73E-02	0	3.14E-04
GWO-FOPID [50]	20	20	20	0.503	0.933	1.95E-02	3.73E-02	0	3.44E-04
SFS-FOPID [52]	20	19.896	16.193	0.402	0.98	2.08E-02	4.56E-02	2.1	6.24E-04
IWO-FOPID [51]	19.482	15.02	12.802	0.431	0.989	2.43E-02	5.86E-02	5.7	8.06E-04

overshoot. Finally, the bode plot as shown in figure 2.10 was drawn for the HHO-FOPID and HHO-PID Controllers and it shows Both controllers have infinite gain margin and phase margin value of 178.9201 deg @ 0.928Hz.

2.7 Approximation of FOPID controllers using neural networks

Despite various techniques available for the design of FOPID controller, there is still scope for research in the realization of these controllers. In general, the fractional order transfer functions can be approximated using integer order systems by using Oustaloop method [24, 25] which suffers from accuracy. In the proposed method, a data driven neural network based approximation technique using NARXnets for realization of FOPID controller is discussed. Once the NARXnet based controller is designed, Real-time implementation of NARXnets does not require complex hardware. It is because the proposed method uses very few neurons.

2.7.1 NARXnets

In the thesis, system identification and controller design methodology was investigated using a class of RNNs (Recurrent Neural Networks) called NARX neural networks. In short form these networks are also called as NARXnets. These are different from regular other types of RNNs in such a way that the feedback for the network only comes from output layers instead of hidden and output layers. The continuous time representation of NARX networks is denoted by

$$\hat{y}(t) = \psi(u(t-n_u), ..., u(t-1), u(t), y(t-n_y), ..., y(t-1)) \tag{2.28}$$

In equation (2.28), the input and output functions at time t are represented as $u(t)$ and $\hat{y}(t)$, the order of delays for input and output are denoted as n_u and n_y, and the mapping function $\psi(\cdot)$ can be identified during the neural network training.

2.7.2 Artificial Neural Networks (ANNs) in NARXnet

Artificial neural networks have the capability to model any non-linear mapping between one or more variables. Because of this property, ANN's are used in different fields of applications like, regression, clustering, classification, function approximation, system identification, pattern classification, and optimization. But ANN performance degrades if the system consists of unknown delays. To accurately model these systems the concept of feedback has to be introduced into the network. On other side, NARXnets have inbuilt feedback from output layer to input layer which makes them suitable for time delayed system identification. This results increased neural network performance, less number of samples to train, early convergence and reduced error [111, 112]. All these advantages of NARXnets comes at the cost of increased number of inputs, external delay units, additional memory and more complexity. In general, NARXnets use ANN's internally for time series prediction applications.

NARXnets can be realized using two methods, known as series-parallel and parallel architectures. The series-parallel architecture does not use feedback, instead, it uses memory to store the temporary outputs produced by ANN. The parallel architecture incorporates feedback in the network so that the output can be directly fed to input using delayed feedback concept. The figures 2.11 and 2.12 describe the series and parallel architectures. In series-parallel architecture, future values are predicted from past and present values of inputs $x(t)$ and original past values of outputs $\hat{y}(t)$. In parallel architecture, the future values of output $\hat{y}$ are predicted from the present and past values of input $x(t)$ and past values of estimated output $\hat{y}(t)$. The mapping function is highly non-linear

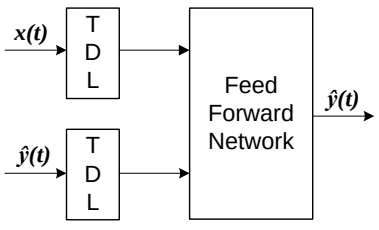

Figure 2.11 Series architecture

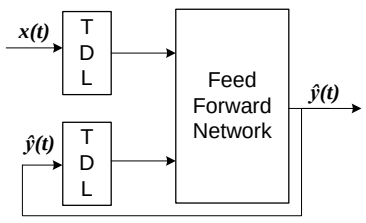

Figure 2.12 Parallel architecture

$\psi(\cdot)$ and is difficult to find. Therefore multi-layer perceptrons(MLP) will be used to find $\psi(\cdot)$. The MLPs have the capability to approximate any non-linear function by proper training. The proposed architecture consists of three layers of neurons named as input

layer, hidden layer, and output layer and all layers contain neurons with weights, biases, and activation functions. Mathematically, the MLP is represented as

$$Z_i = f(\sum_{j=1}^{n} x_j \cdot w_{ij}) \tag{2.29}$$

In equation (2.29), Z_i represents the response from output neuron i , x_j represents input j and w_{ij} indicate weights associated with the i and j layers. The function $f(\cdot)$ indicates the activation function and corresponding mathematical definitions are given by

$$Sigmoid : f(x) = \frac{1}{1+e^{-x}} \ \epsilon \ ]0,1[\tag{2.30}$$

$$Linear : f(x) = x \ \epsilon \ R \tag{2.31}$$

$$Tansigmoid : f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \ \epsilon \ ]-1,1[\tag{2.32}$$

These networks are trained to identify the input-output mapping based on various training algorithms [113], [123]. The weights of the neural network are updated during the training process based on the error between target and actual output, produced by the neural network.

2.7.3 Proposed neural network architecture

The entire system architecture was divided into five components as described in figure 2.13. The DC motor block, generates the necessary data to identify system dynamics with plant neural network. The HHO-FOPID controller, which acts as a reference data generator to train the controller neural network. Levenberg-Marquardt (LM) algorithm [113] was used to adjust the weights of plant and controller neural networks during training process. The working procedure of DC motor and HHO-FOPID controller was already discussed in section 2.4. In the system, the neural network is divided into two parts namely, plant network and controller network. Each of these networks consists of an input layer, a hidden layer, and the output layer. The input layer consists of delay blocks which delays each of the inputs by one and two units. If there are n number of inputs after passing through delay blocks 3n inputs are produced.

Therefore the delayed signals along with original inputs were fed as input to the network. The delayed signals will improve the dynamic behavior of the neural network

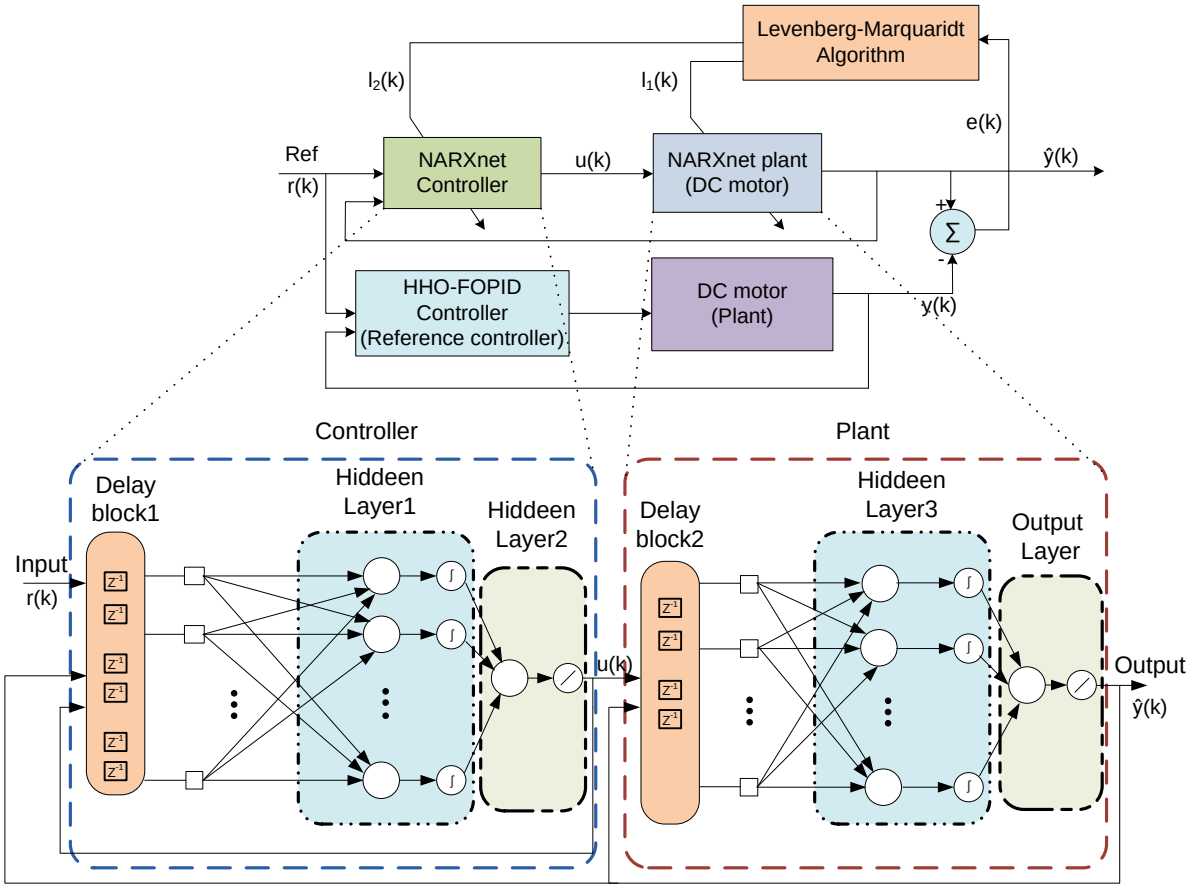

Figure 2.13 Proposed neural network architecture

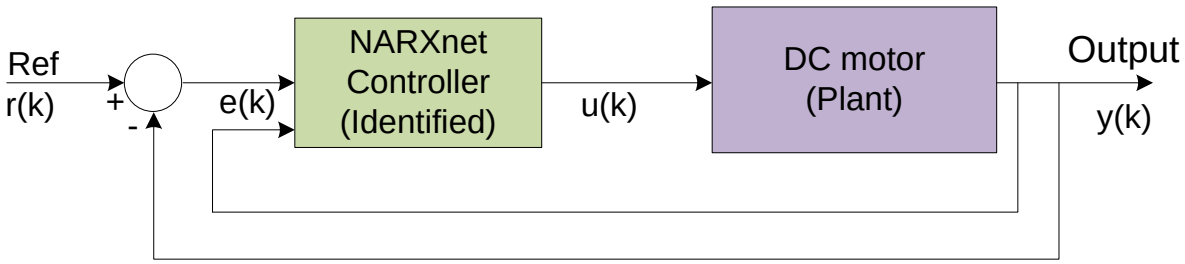

Figure 2.14 Architecture of proposed system

during training. Sigmoid and linear activation functions are used to activate hidden and output layers respectively. The overall neural network architecture was summarized in figure 2.14. The feedback connections are established from plant output to plant input and controller input, and from controller output to controller input.

2.7.4 Training plant neural network

To design the NARXnet controller, two neural networks (plant and controller neural networks) have to be trained. Therefore, the total training part was divided into two phases. In the first phase the dynamics of plant neural network are identified, in the second phase, the identified plant dynamics along with controller training data are used to train the controller network.

The necessary training data to train the neural networks was generated using Simulink as described in figure 2.15. The signals in_plant, out_plant and in_cf, out_cf represents the training data for plant and controller respectively.

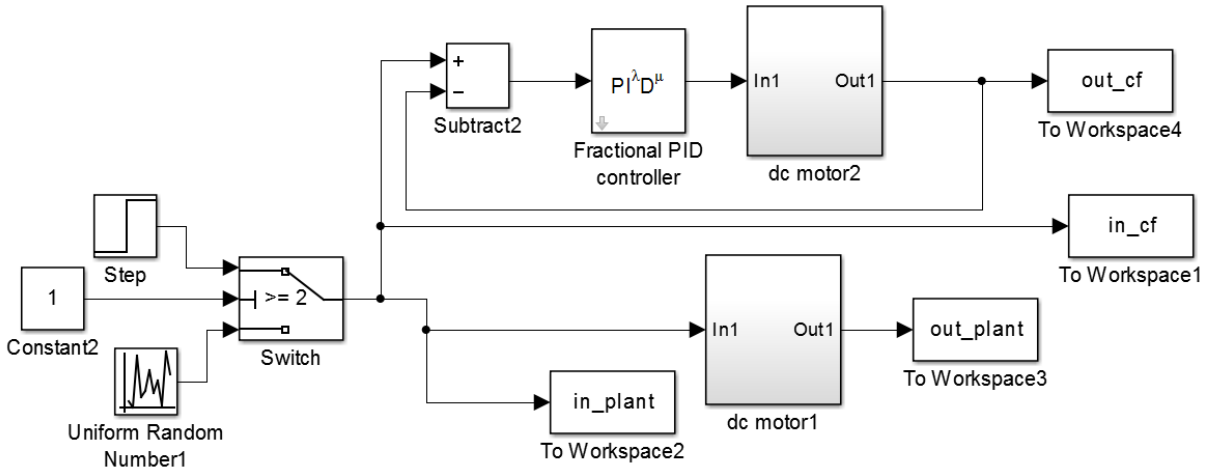

Figure 2.15 Data generation from Simulink

There are a total of 10,000 training samples are generated with sampling frequency of 1ms(0.001s) by varying magnitudes and time duration. The data set consists of excitation (input) and response (output) of DC motor. The graphical representation of training data for the plant network is shown in the figure 2.16.

Initially, the plant neural network was created with 10 hidden neurons and an output neuron. There are two input signals for the plant neural network namely reference signal and feedback signal. These signals are delayed by 1 and 2 time units respectively. Therefore there are four inputs including the delayed signals. During the plant network training, the controller network was deactivated and reference signal r(t) was directly supplied to the plant along with output feedback.

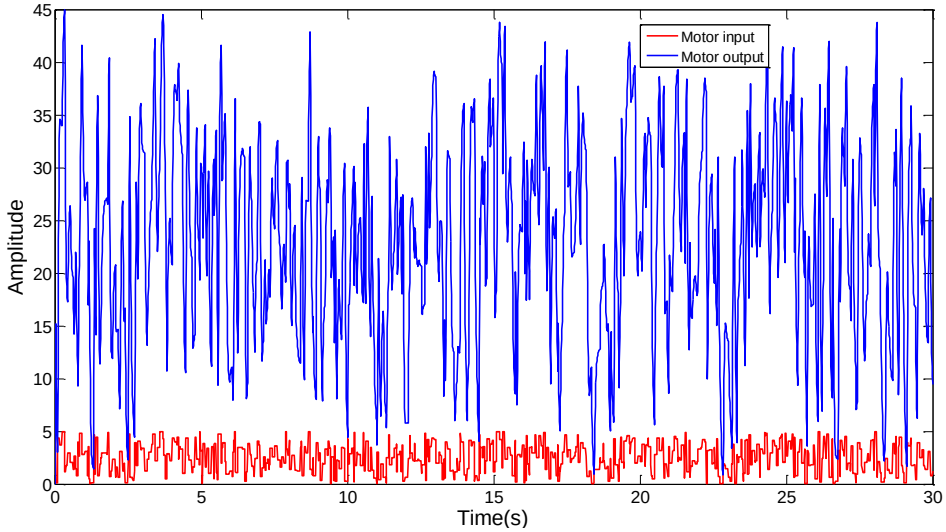

Figure 2.16 Training data for plant neural network

Levenberg−Marquardt algorithm [113] was used to identify the neural network weights with mean square error (MSE) evaluation criteria. The training performance curve of the plant network was shown in figure 2.17. At the end of training, the plant network produced minimum cost value 1.5719e-09 for 1000 epochs.

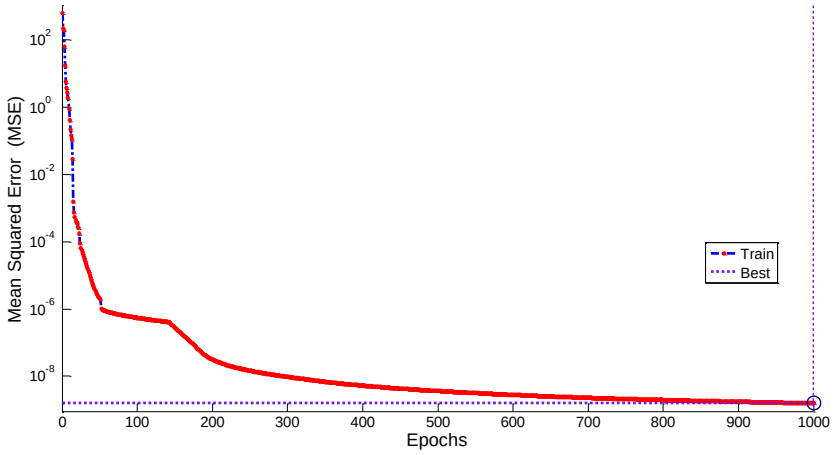

Figure 2.17 Training performance of the plant network

2.7.5 Training controller neural network

The controller training data is generated from the reference HHO-FOPID controller using Simuliunk. The training data set consists of 3000 samples of excitation and response of HHO-FOPID controller. For proper training, different targets are considered in the data

set with varying magnitudes in the range [0,5]. The training data for the plant network is shown in figure 2.18.

The controller network is designed with 10 neurons in the hidden layer and a neuron in the output layer. The controller neural network was also trained similar to the plant neural network. Here, the trained plant network was added to the controller network whose output can be used to calculate the error. The error value is used to update the weights of the controller network. Since the plant network was already trained its weights are not updated during controller network training i.e the plant network learning rate was made zero. To train the controller LM algorithm with mean square error (MSE) criteria

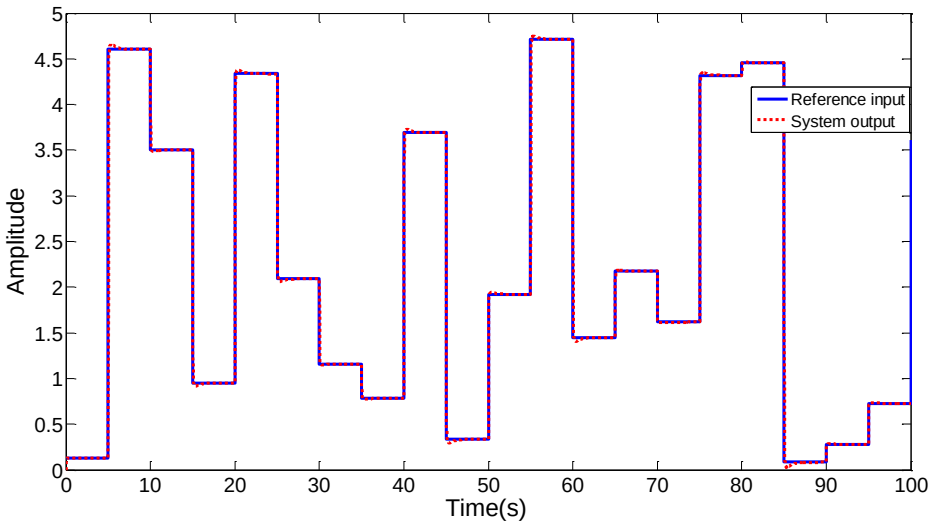

Figure 2.18 Training data for the controller neural network

was used. The training performance of the controller network is shown in figure 2.19. The controller network took 250 epochs to produce the minimum error value 0.003021. The corresponding parameter values used during the plant and controller network are listed in table 2.4.

For any neural network the number of epochs and neurons required are hyper parameters and depends on type of the input data. In the proposed architecture, the number of epochs for each network is determined based on the MSE value. The training started with random number of epochs. After several attempts it is found that the plant network produced the lowest value of MSE at 250 epochs. Using the similar procedure the controller network produced minimum MSE value after 1000 epochs. As there is no predetermined procedure or formula to identify the number of hidden layer neurons, the

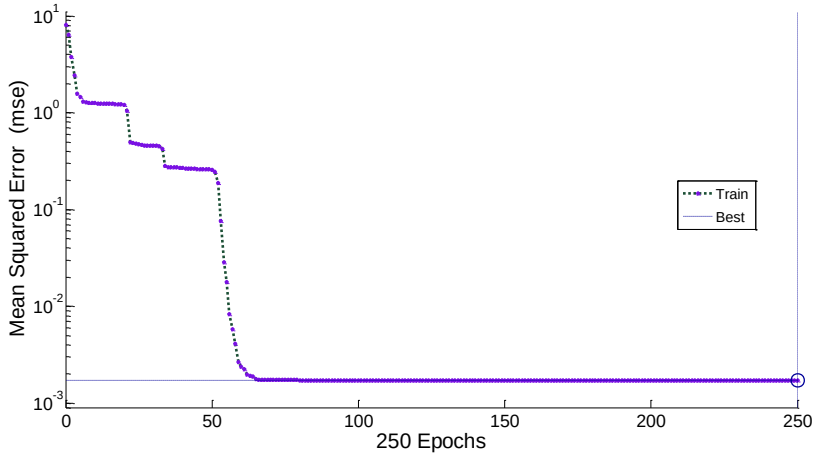

Figure 2.19 Training performance of the controller network

Table 2.4 Parameters for plant and controller neural networks

Parameter	Plant network	Controller network
Inputs	2	3
Unit delay blocks	4	6
Hidden layer size	10 neurons	10 neurons
Output layer size	1 neuron	1 neuron
Epochs	1000	250
Training samples	3000	10000
Mean Square Error(MSE)	1.572E-09	0.003021

plant and controller network hidden layer neurons are identified using the trial and error procedure.

The number of delayed signals required to train network depends on the type of the system. In general the discrete PID controllers are second order, therefore in the proposed architecture for each input 2 delay blocks are used.

2.8 Results and discussions

All the simulations are performed using MATLAB R2016b on a system with Intel core i5 Processor with 8GB RAM. To investigate the efficiency of the proposed neural

controller it was correlated with various optimization algorithm based FOPID controllers. For all controllers step and load responses analysis are performed.

2.8.1 Step response

Primarily, the step response is plotted for all the controllers as shown in figure 2.20. The response of proposed neural controller is compared with SFS-PID, IWO-PID, GWO-PID, ASO-PID, GWO-FOPID, ChASO-FOPID, HHO-FOPID, ASO-FOPID, HHO-PID, HGSO-PID, OBL/HGSO-PID, MRFO-FOPID and OBL-MRFO-SA-FOPID controllers. From the step response rise time, settling time, overshoot, and steady state error are calculated. The comparison of different performance metrics of the FOPID/PID controllers are listed in table 2.5.

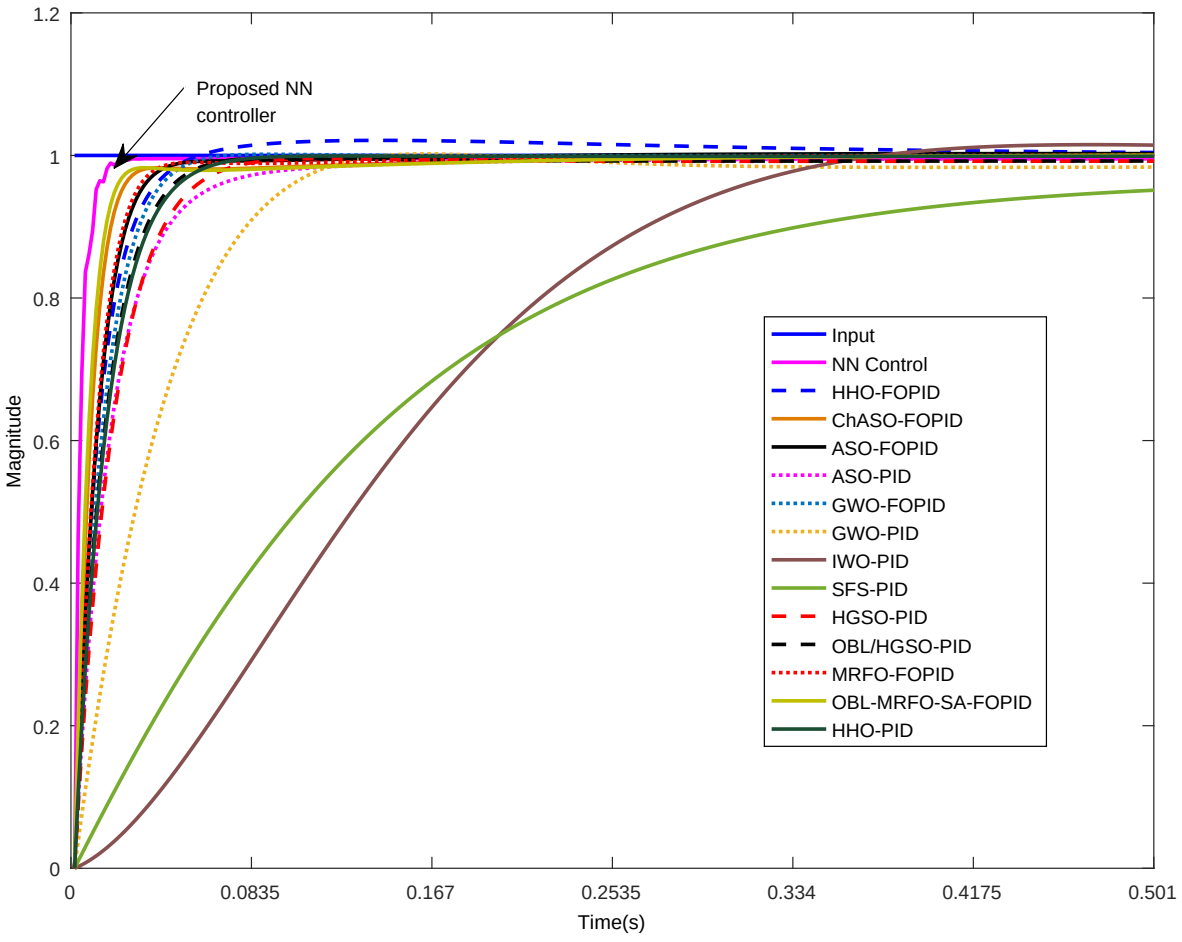

Figure 2.20 Comparison of step response for FOPID/PID controllers

Table 2.5 Comparison with FOPID/PID controllers

Algorithm-controller	Rise time(s)	settling time(s)	overshoot(%)	error(ss)
NN-Controller(Proposed)	**0.0079**	**0.0161**	**0**	4.50E-03
ChASO-FOPID [2]	0.0253	0.0405	0	7.32E-04
ASO - FOPID [2]	0.0376	0.0616	0	1.80E-03
ASO-PID [2]	0.0692	0.1535	0	3.70E-03
GWO-FOPID [50]	0.0488	0.0814	0.3145	**2.20E-05**
GWO-PID [50]	0.1388	0.2052	1.5062	1.51E-02
IWO-PID [52]	0.1489	1.2533	6.9759	2.86E-04
SFS-PID [51]	0.5436	1.4475	0	3.07e-02
HHO-PID [59]	0.0568	0.1003	0	8.80E-04
HGSO-PID [58]	0.1122	0.1966	0	7.6E-03
OBL/HGSO-PID [58]	0.0894	0.1559	0	7.6E-03
MRFO-FOPID [57]	0.0355	0.0562	0.1546	-
OBL-MRFO-SA-FOPID [57]	0.0214	0.0339	0	-

The results show that the proposed controller produced minimum rise time 0.0079s and took 0.0161s to settle. The controller do not produce any overshoot while GWO and IWO based controllers produced overshoot with varying steady-state errors. For better visualization, The pictorial representation for the data mentioned in table 2.5 is shown in figure 2.21. From the table 2.5, it can be observed that the NARXnet based controller shows improved performance than other FOPID/PID controllers in terms of settling time, rise time, and overshoot.

2.8.2 Load response

To further investigate, all the controllers are subjected to track different set points within a single run. The corresponding results are shown in the figure 2.22. The response curves show how the proposed neural controller able to track the load variations over the time. The results indicate that the NARXnet controller is able to quickly track the set point changes than the traditional FOPID/PID controllers.

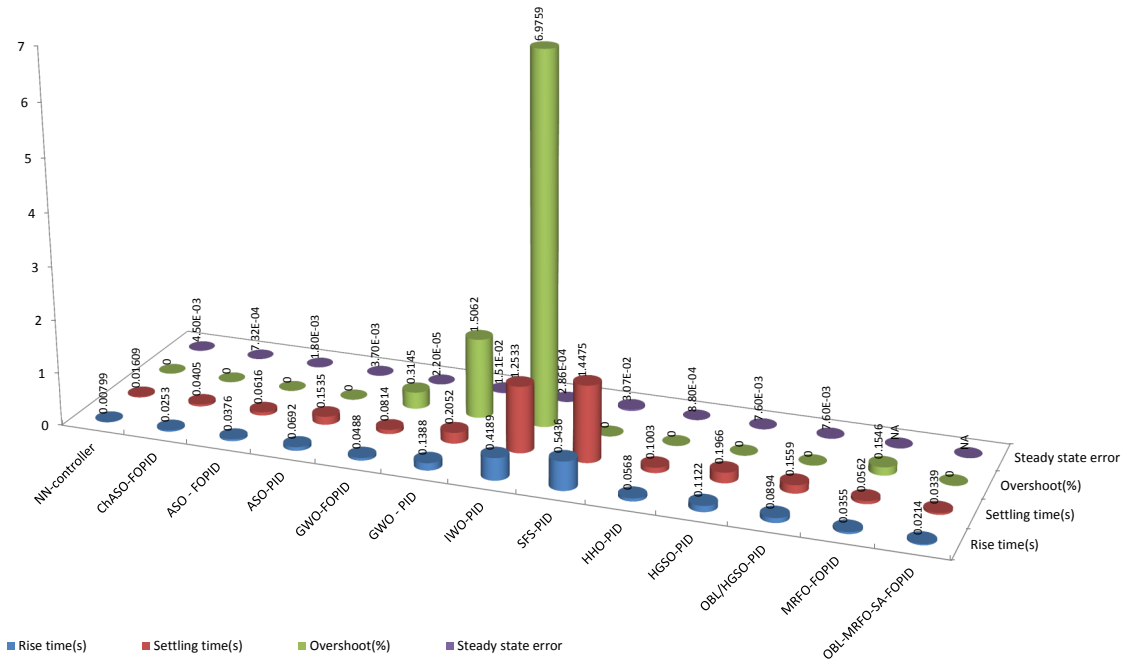

Figure 2.21 Comparison of performance metrics identified from step response

To get more insight about the NARXnet based FOPID controller, during the set point tracking the controller effort is calculated and plotted as shown in figure 2.23. The graph clearly shows that the controller responds only during the set point changes. This indicates that the controller tracks different set points efficiently.

Further, it is observed that the design of FOPID controllers is a difficult task because of their infinite order. So to get satisfactory performance FOPID controllers should be approximated using very high order integer systems. To overcome this problem, the proposed approach can be used for approximation of FOPID controllers using neural networks.

2.9 Real-time implementation of FOPID controller for DC motor speed control

To verify the working of FOPID controller it is implemented in real time to control the speed of DC motor. Since fractional order controllers are infinite order filters, for

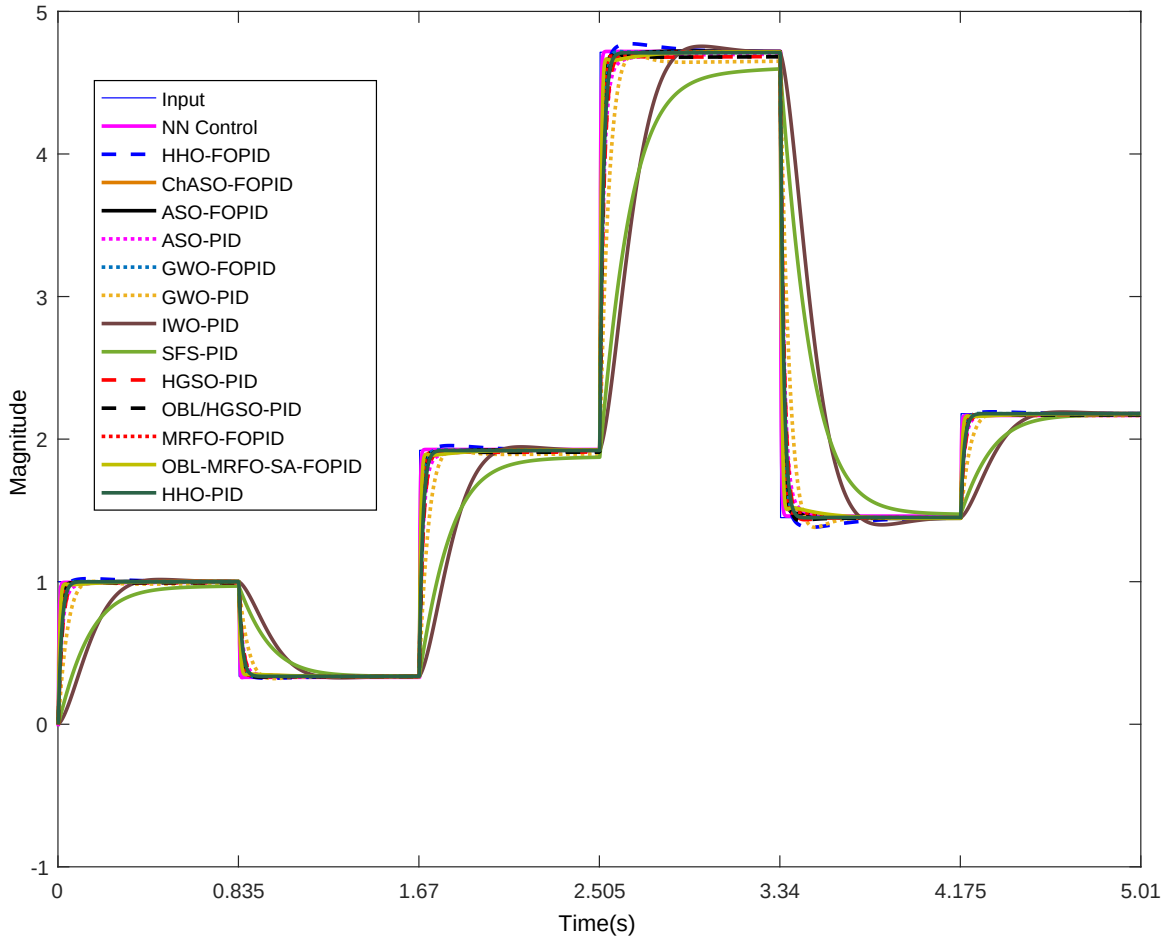

Figure 2.22 Comparison of load response for FOPID/PID controllers

real-time implementation different techniques are mentioned in the literature such as Oustaloop approximation [24], CRONE controller [26] and fractional lead-lag compensator [30]. In the proposed work, Oustaloop approximation technique was chosen because of its simplicity and reliability. The detailed description about the implementation procedure is mentioned as follows.

2.9.1 Identification of DC motor model

An externally excited armature controlled shunt DC motor is considered as plant. The manufacturer specifications are given the table 2.6. The following mathematical model is considered to identify the various parameters of the DC motor. The detailed

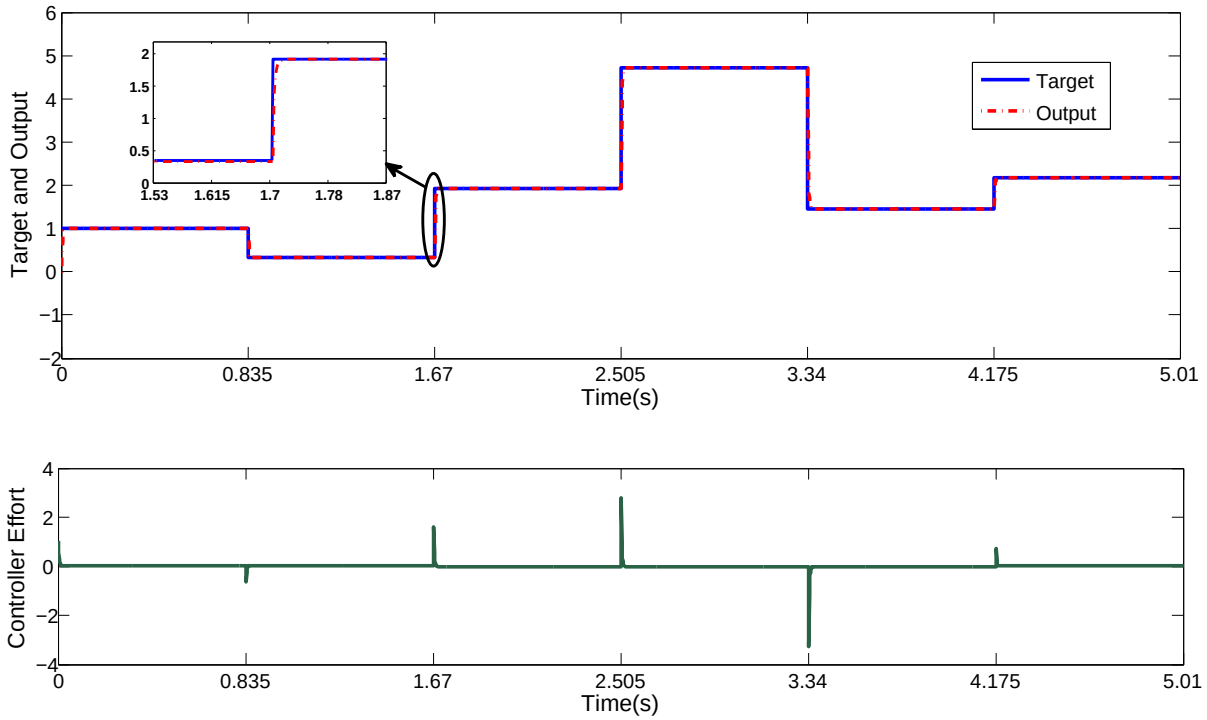

Figure 2.23 Narxnet controller response analysis

Table 2.6 Specifications of DC motor

S.No	Parameter	Value
1.	Rated Speed	1500RPM
2.	Rated Power	5HP (3.7KW)
3.	Max. Excitation Voltage	230V (DC)
4.	Max. Current	15A

description of the mathematical model can found in the section 2.3.

$$G(s) = \frac{\omega(s)}{E_a(s)} = \frac{k}{(sL_a + R_a)(Js + B) + k_b k} \tag{2.33}$$

where

R_a : Armature resistance(Ω), L_a :Inductance of armature (H),

J : Inertia torque($kg.m^2$), K_b : e.m.f constant($V.s/rad$),

K : Motor torque constant($N.m/A$), B : Motor friction constant($N.m.s/rad$)

From the mathematical model of the DC motor, using system optimization toolbox available in MATLAB the parameters of the DC motor are identified. For this purpose, the actual motor response is considered for the different input voltages. This data is fed as

input to the system identification toolbox and the corresponding model of the DC motor is identified. The identified parameters are listed in table 2.7.

Table 2.7 Identified parameters of DC motor

S.No	Parameter	Value
1.	R_a (Armature resistance)	1.86 Ω
2.	L_a (Armature Inductance)	1.1295E-04 H
3.	J (Inertia torque of motor)	8.241E-04 $Kg.m^2$
4.	B (Friction constant)	3.82E-03 $m.s/rad$
5.	Kt (Torque constant)	0.085711 Nm/A
6.	Kb (Back emf constant)	0.085711 $V.S$

Substituting the identified DC motor parameter values into equation (2.33), the plant model is obtained as mentioned in equation (2.34).

$$G(s) = \frac{\omega(s)}{E_a(s)} = \frac{9.208E-05}{9.308E-08S^2 + 0.00153S + 0.01445} \tag{2.34}$$

The comparison of responses of actual motor and simulated model are represented in 2.24. From the figure, it is clearly observed that the identified model response is almost similar to the actual motor response. Moreover most of the error values are very small which are nearer to zero.

By using the recognized model as mentioned in above equation, the step response is plotted and it is shown in figure 2.25. From the step response, different performance measures are calculated and shown in the table 2.8. From the table 2.8, it is found that

Table 2.8 Performances of DC motor

S.No	Parameter	Value
1.	Rise time	0.24s
2.	Settling time	0.46s
3.	Overshoot	0%
4.	Steady state error	4.93

the model has very large steady state error and rise time and settling time can be further

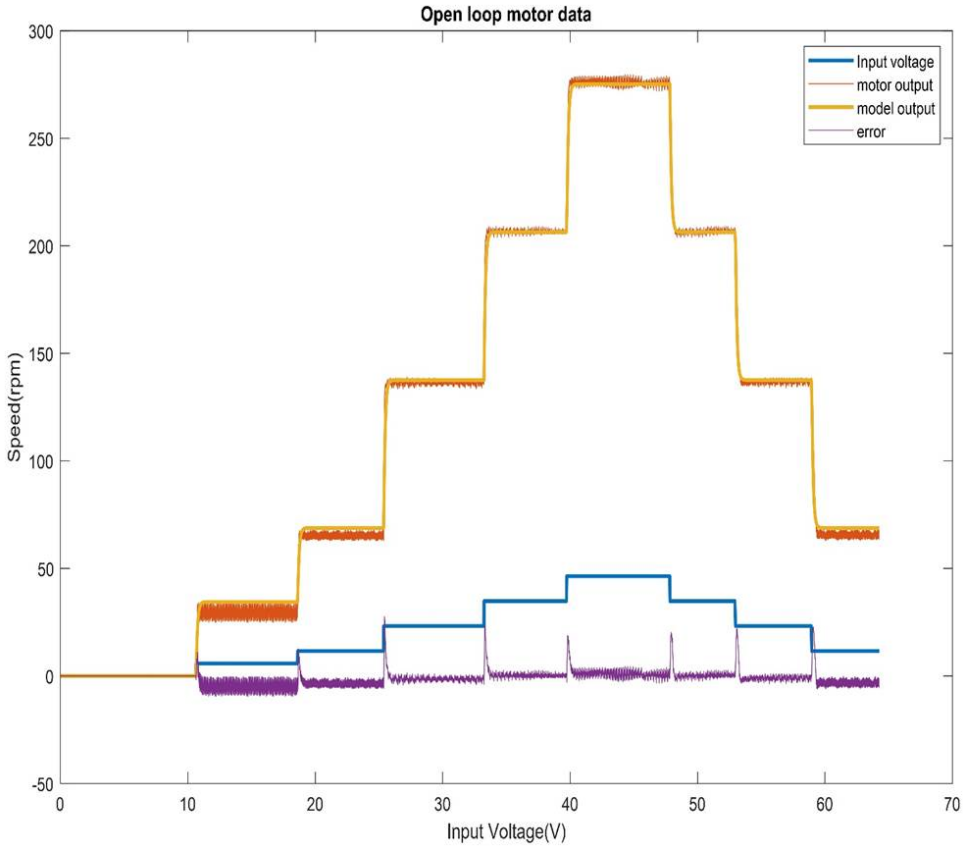

Figure 2.24 Comparison of response of identified model and actual plant

improved. Therefore a controller is designed to reduce the steady state error and to improve rise time and settling times.

2.9.2 Harris Hawks Optimization(HHO) based fractional PID controller design

The block diagram of the HHO based FOPID controller was shown in the figure 2.26. The FOPID controller uses the optimization algorithm to identify the tuning parameters K_p, K_i, K_d, λ, and μ. Initially the objective function has to be identified for the proper tuning of the parameters. There are different objective functions were proposed for the proper tuning of PID controller parameters like IAE, ISE, ITSE, and ITAE. Out of these functions ITSE yields less settling and rise times with comparable overshoot values in comparison to existing approaches. To further investigate the effectiveness of the proposed algorithm another objective function called ZLG (Zwe Lee Gaing) was used. In this algorithm the objective function was chosen as rabbit(prey) and the parameters

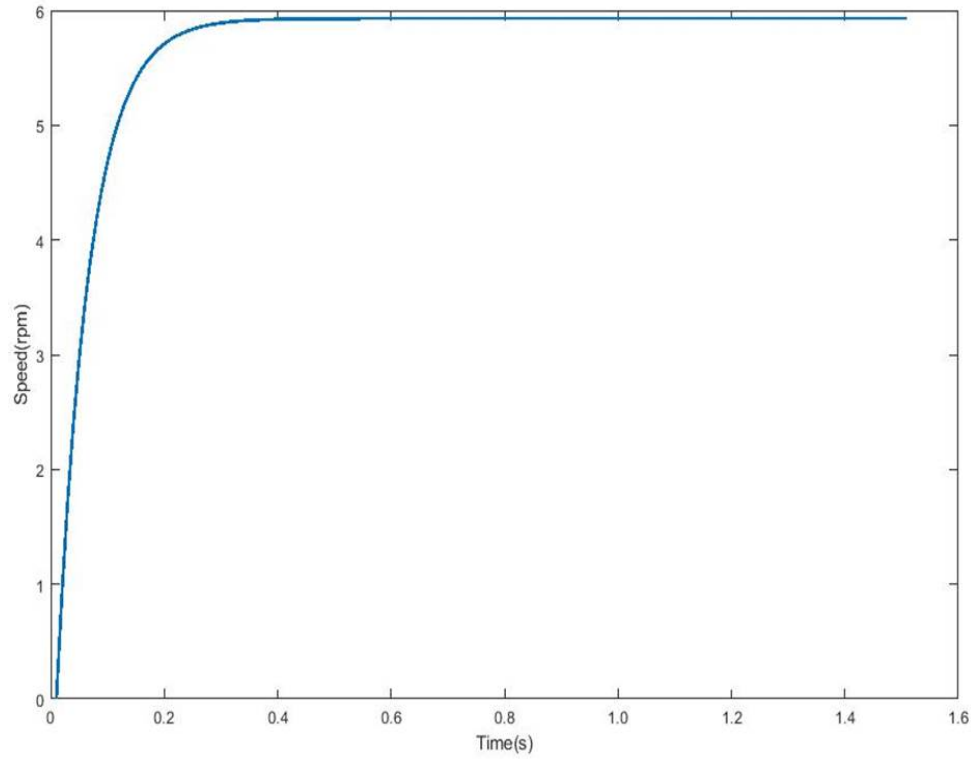

Figure 2.25 Step response of identified model

K_p, K_i, K_d, λ, and μ as Harris Hawks. For each iteration of the algorithm the error value for the output of DC motor was calculated and substituted into the objective function. Based on the error values the parameters are updated according to the described procedure of optimization algorithm. The above process is repeated until the terminating criteria was met. Then finally the tuned values are used to design the FOPID controller.

2.9.3 Objective function and constraints

To identify the optimum parameters of the FOPID controller the following objective functions are considered. The corresponding equations for ITSE, ITAE, and ZLG functions were mentioned in (2.35), (2.36), and (2.37) respectively. The proportional, integral and derivative terms(K_p, K_i, and K_d) were limited to the range $[0, 20]$ and fractional powers (λ and μ) were limited to the range $[0, 2]$.

$$ITSE = \int_0^T t.e^2(t)dt \tag{2.35}$$

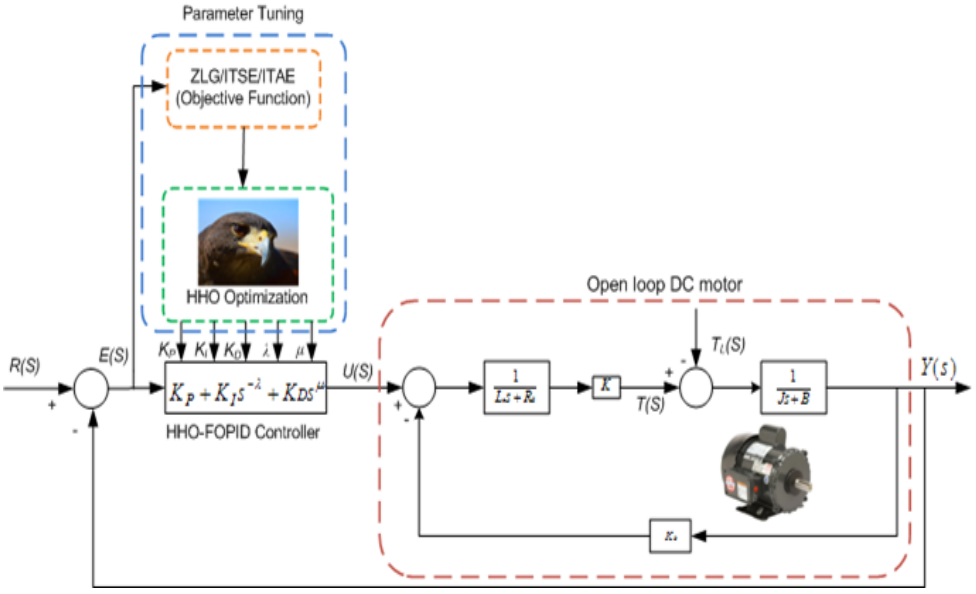

Figure 2.26 Controller tuning using HHO algorithm

$$ITAE = \int_0^T |t.e(t)|dt \tag{2.36}$$

$$ZLG = (1 - e^{-\beta}).(M_p + E_{ss}) + e^{-\beta}.(t_s - t_r) \tag{2.37}$$

M_p, E_{ss}, t_s, and t_r denotes the maximum overshoot, steady state error, settling time and rise time, respectively. The weight factor β is usually taken as 1.0. Using the methods discussed in the section 2.8.2, the simulations are performed in MATLAB to design and optimize the fractional order controller. The identified FOPID controller parameters using ITSE, ITAE, and ZLG cost functions are listed in table 2.9.

The corresponding step responses and performances of designed FOPID controllers are shown in figure 2.27 and table 2.9 respectively.

From the analysis of performances it is found that the ZLG objective function produced better controller parameters.

2.9.4 Implementation of FOPID controller on DSPACE platform

For the implementation of FOPID controller the following set up is established. To realize the identified FOPID controller DSPACE 1104 hardware is used. To drive the DC motor IGBT drivers and converter modules are used. The feedback is established by

Table 2.9 Comparison of FOPID controller parameters

Objective function	K_p	K_i	K_d	λ	μ	Rise time (ms)	Settling time (ms)	Overshoot(%)	E_{ss}
ITAE	20	19.99	20	1.53102	5.24e-03	1.8245	4.4529	0	0
ITSE	20	19.99	4.57	1.11463	0.319084	1.7754	6.997	0	0
ZLG	5.26452	3.6557	20	1.31267	0.817401	0.7942	1.9729	0	0

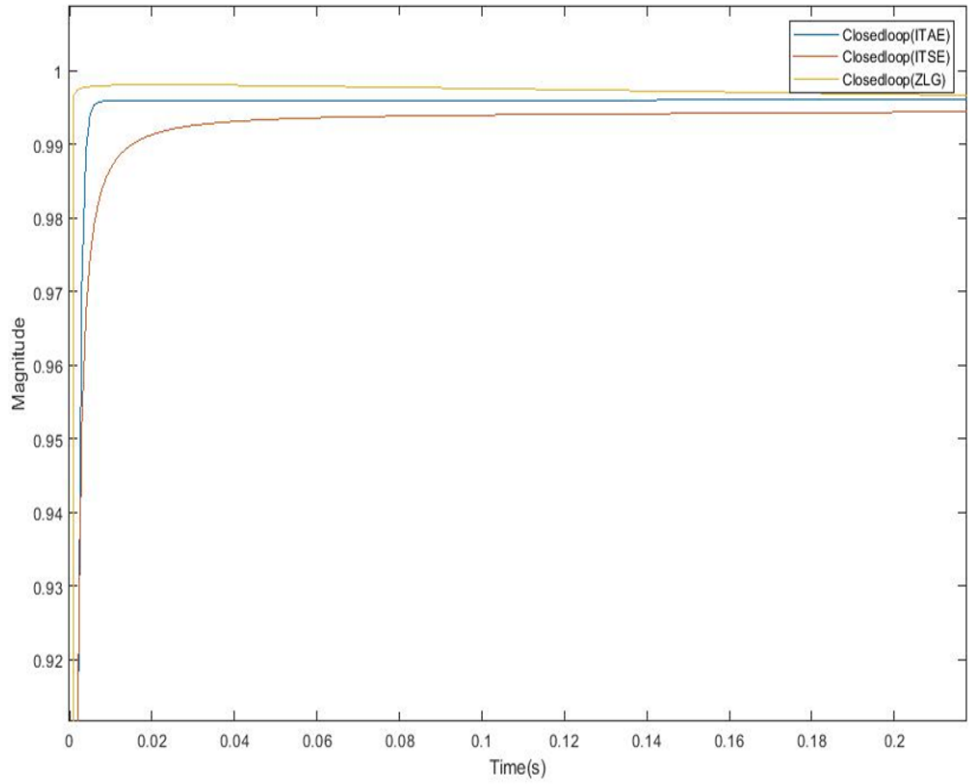

Figure 2.27 Step responses of designed FOPID controllers

connecting rotary encoder module between the motor and DSPACE board. The corresponding block diagram is shown in figure 2.28.

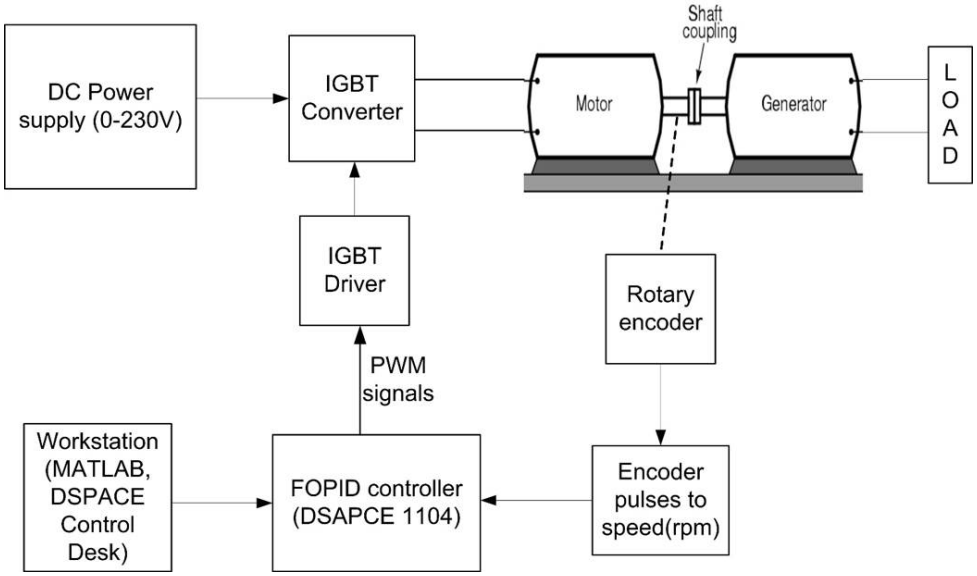

Figure 2.28 Block diagram for hardware setup

The FOPID controller is developed using the Simulink as shown in the figure2.29. Initially, rotary encoder (Autonics $E50S8 - 2500 - 3 - T - 24$) pulses are captured converted to position and velocity as shown in figure 2.30. During the conversion process to eliminate the noise in pulses a 12 order averaging filter is used whose Simulink diagram is shown in figure 2.31.

The measured speed is compared with the reference speed and error signal is generated which acts as input to FOPID controller. To avoid the error accumulation at the beginning due to integration, anti-wind up loop is added in the controller as shown in figure 2.30. To protect the electronics from the large control signals, a saturation block is used at the controller output. Finally, the controller output is given to the PWM generater to generate necessary control signals for the DC motor.

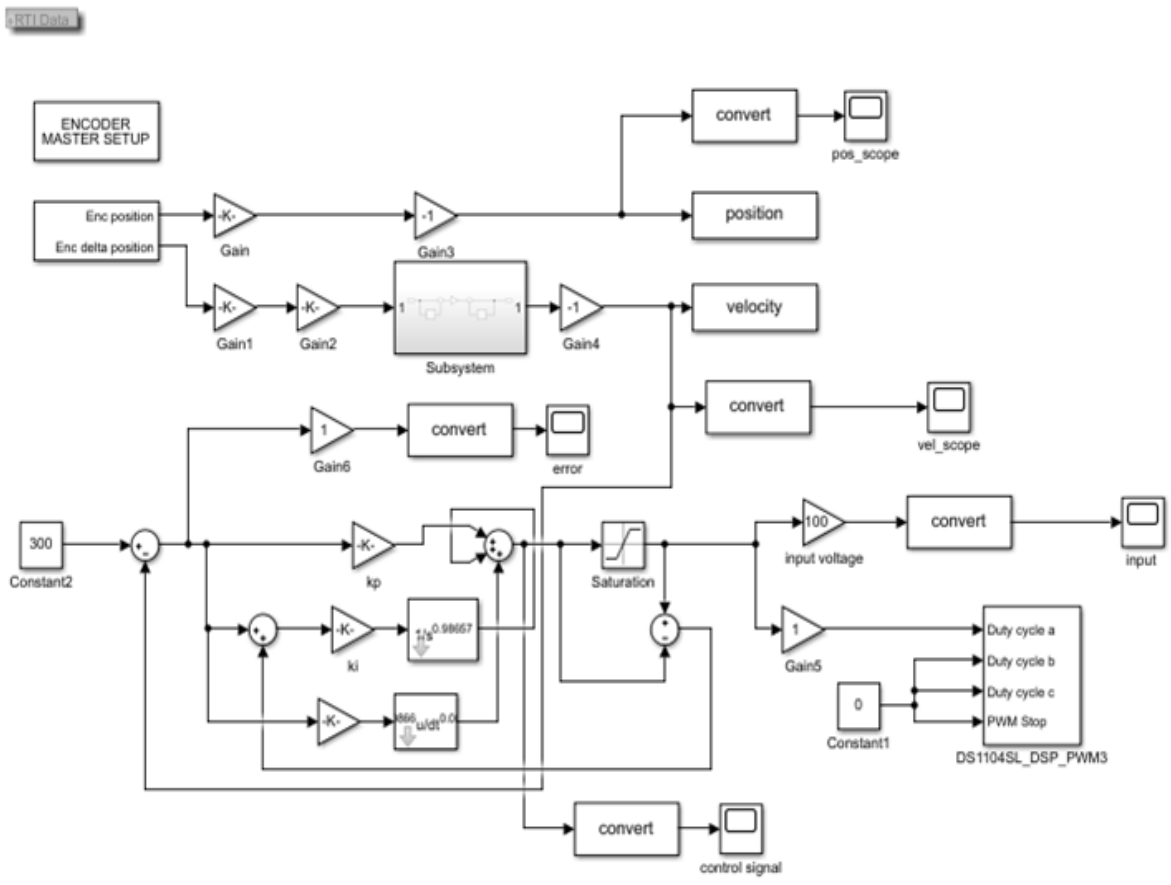

Figure 2.29 Simulink block diagram of FOPID controller implemented on DSPACE 1104

To develop the hardware model of the FOPID controller the following procedure is used.

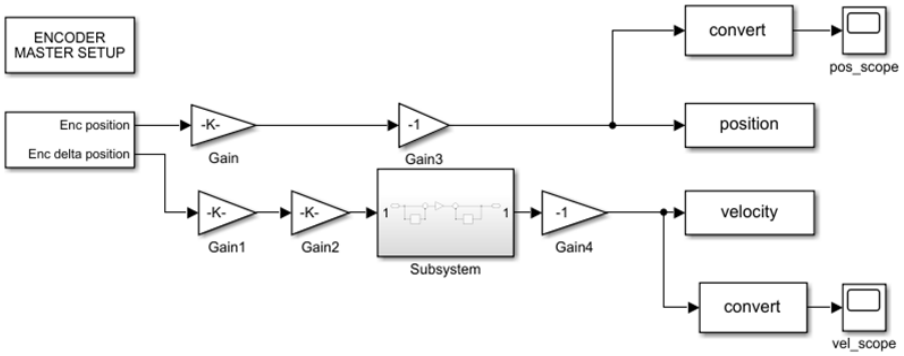

Figure 2.30 Block diagram for hardware setup

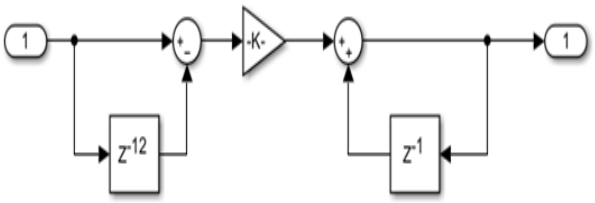

Figure 2.31 Block diagram for subsystem

Step1: Develop the model in Simulink

Step2: Generate the C/C++ code using DSPACE configuration tool

Step3: Generate .sdf file for DSPACE 1104

Step4: Upload .sdf file into the DSPACE 1104 control desk

Step5: Develop the layout for the visualization

Step6: Calibrate DSPACE board

Step7: Run the program using DSPACE 1104 control desk

Step8: Observe the results in control desk

Step9: End the execution

The real-time setup for the plant to verify the proposed FOPID controller functionality is shown in the figure 2.32. The response of the controller for the set point 350 RPM is obtained using control desk software. The corresponding result is shown in the figure 2.33.

Figure 2.32 Real-time plant setup for DC motor speed control using FOPID controller

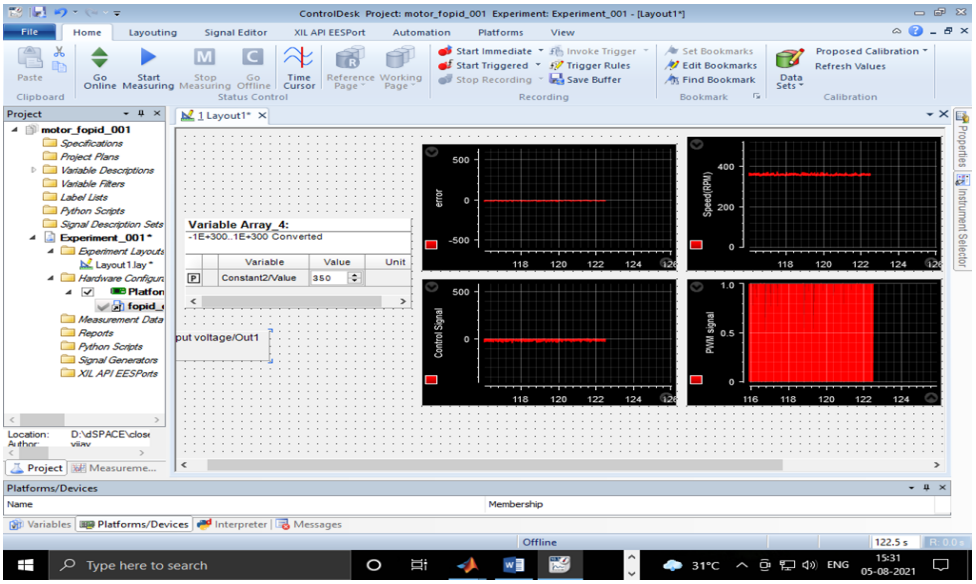

Figure 2.33 Response of the controller for 350 rpm set point

2.9.5 Summary

The techniques mentioned in the literature [9]-[20] use meta-heuristic optimization algorithms to identify the FOPID controller parameters. Where in the parameters of the PID controllers are tuned using both optimization algorithms and neural networks [6]-[8]. Moreover, realization of FOPID controllers requires infinite order filters. Practically, the reduced order filters are used for the realization but these filters limit the accuracy of controller response. To overcome this problem, in the proposed method an effort has

been made to approximate the FOPID controller using NARXnets.

Quantitatively, when we look at the comparison of step response, the NARXnet based controller produced no overshoot and 0.0079s rise time and 0.0161s settling time which are the lowest. The lowest steady state error is produced by ASO-FOPID controller which is 2.20E-05 at the same time the controller also has overshoot of 0.315%. From the values mentioned in the table 4.6, it can be said that the proposed NARXnet controller produces the quickest response from the DC motor than the other controllers.

2.9.6 Conclusion

NARXnet based system identification and controller design was presented for DC motor speed control. The training data for plant network was generated from the mathematical model of the plant. For the controller network, the data was generated from HHO tuned FOPID controller using Simulink. The proposed design method for controller was compared with traditional optimization based approaches. The results indicate that the NARXnet controller shows superior performance over the existing controllers in tracking the set points. In addition, it is found that the NARXnets can capture the dynamics of FOPID controller. Therefore the developed method can be used for approximation of FOPID/PID controllers using neural networks. Since these networks can perform better under noisy environment, the proposed idea can be extended for system identification under environmental noise and interference effects.

In addition, HHO-FOPID controller is implemented in real-time to verify its operation. Initially, the DC motor model is identified using the MATLAB system identification toolbox. Later the identified model is used to design and optimize the FOPID controller using HHO algorithm. Finally, the optimized controller is implemented on the DSPACE platform and its operation is verified on control desk software.

Chapter 3

A novel Single Neuron FOPID (SNFOPID) controller using Chaotic Political Optimizer algorithm with application to DC motor speed control

This chapter proposes a novel architecture for FOPID controller inspired from the working principle of artificial neuron. The architecture is developed by combining the FOPID controller and neural network structures. Various parameters of the proposed controller are optimized using a novel chaotic political optimizer algorithm. For efficient tuning a new objective function is proposed based on the controller performances.

It is mentioned that the method proposed in the chapter 2 requires two neural networks. Therefore two sets of training data is required to train the plant and controller neural networks. Moreover the performance of the overall system depends on two factors. The first one is amount of training given to the plant and controller networks and second one is how accurately the pant network is trained as compared to the actual system. Therefore multiple rounds of training is required causing increased computational complexity. To overcome the problem a different type of controller is proposed in the present chapter which reduces the training complexity as it requires only one neuron to be trained.

3.1 Discretizaion of fractional differ-integrals

Implementation of fractional order controllers requires infinite order filters and uses additional computing power and memory than PID controllers. Moreover, there are no standard architectures defined for fractional order PID controllers. In this regard, inspired from the FOPID controller equation, in the proposed method a neural network based architecture is presented. Since the neural networks perform better with discrete data, the proposed architecture uses a generating function developed based on Al-aloui operator [114] to discretize the fractional integral and differential systems.

The time-domain representation governing the operation of FOPID controller is given in equation (3.1).

$$u(t) = K_p\varepsilon(t) + K_i D_t^{-\lambda}\varepsilon(t) + K_d D_t^{-\mu}\varepsilon(t) \tag{3.1}$$

Where $u(t)$ is the desired signal, $\varepsilon(t)$ indicates error signal and $\lambda, \mu \, \epsilon \, (0, 1)$. The terms $D_t^{-\lambda}$ and $D_t^{-\mu}$ are related to fractional calculus [6] which are defined as

$${}_aD_t^{\alpha} = \begin{cases} \frac{d^{\alpha}}{dt^{\alpha}}, & \alpha \, \epsilon \, \mathbb{R}^+, \\ 1, & \alpha = 0, \\ \int_a^t (d\tau)^{-\alpha} & \alpha \, \epsilon \, \mathbb{R}^- \end{cases} \tag{3.2}$$

Where $(a, t) \, \epsilon \, \mathbb{R}^+$ are limits and $\alpha \, \epsilon \, \mathbb{R}$ is the order of differ-integral term D.

Because of convenient form for computations, in the research work, Grunwald-Letnikov definition [6] represented in equation (3.3) is used to simulate fractional order systems in computer.

$${}_aD_t^{\alpha} f(t) = \lim_{x \to \infty} \frac{1}{h^{\alpha}} \sum_{k=0}^{[\frac{t-a}{h}]} (-1)^k \binom{\alpha}{k} f(t - kh) \tag{3.3}$$

Where $h \, \epsilon \, \mathbb{R}$ is the computation step size. Application of Laplace transform to the equation (3.1) produces equation (3.4)

$$U(s) = K_p E(s) + K_i \frac{E(s)}{s^{\lambda}} + K_d E(s) s^{\mu} \tag{3.4}$$

The fractional order system mentioned in equation (3.4) can be descretized by approximating continuous s-function with the descretizing function $\omega(z^{-1})$. Various techniques

exist to descretize the continuous fractional transfer functions [114] and [115]. The generalized mathematical definition of the $\omega(z^{-1})$ is shown in equation (3.5).

$$\omega(z^{-1}) = \left(\frac{1+\sigma}{T}\right)\left(\frac{1-z^{-1}}{1+\sigma z^{-1}}\right) \tag{3.5}$$

In the equation (3.5), substituting $\sigma = 1/7$ results in Al-aloui descretization operator, $\sigma = 1$ and $\sigma = 0$ results Tustin and Euler's backward difference approximation techniques, respectively.

$$s^{\mu} = (\omega(z^{-1}))^{\mu} = \left[\left(\frac{1+\sigma}{T}\right)\left(\frac{1-z^{-1}}{1+\sigma z^{-1}}\right)\right]^{\mu} \tag{3.6}$$

Considering the power series expansion (PSE) of equation (3.6)

$$\left(\frac{1+\sigma}{T}\right)^{\mu}\left(\frac{1-z^{-1}}{1+\sigma z^{-1}}\right)^{\mu} = \left(\frac{1+\sigma}{T}\right)^{\mu}\sum_{i=0}^{\infty}\psi_i(\mu)z^{-i} \tag{3.7}$$

where $i \,\epsilon\, \mathbb{N}$ and

$$\psi_i(\mu) = \frac{1}{i!}\frac{d^i}{d\omega^i}\left(\frac{1-\omega}{1+a\omega}\right)^{\mu}_{\omega=0} \tag{3.8}$$

The integral term $s^{-\lambda}$ is re-written as

$$s^{-\lambda} = \frac{1}{s}s^{1-\lambda} \tag{3.9}$$

Then the corresponding generating function expansion is given by

$$s^{-\lambda} = \left(\frac{T}{1+\sigma}\right)\left(\frac{1+\sigma z^{-1}}{1-z^{-1}}\right)\cdot\left[\left(\frac{1+\sigma}{T}\right)\left(\frac{1-z^{-1}}{1+\sigma z^{-1}}\right)\right]^{1-\lambda} \tag{3.10}$$

substituting (3.8) into (3.10)

$$s^{-\lambda} = \left(\frac{1+\sigma}{T}\right)^{1-\lambda}\left(\frac{1+\sigma z^{-1}}{1-z^{-1}}\right)\sum_{i=0}^{\infty}\psi_i(1-\lambda)z^{-i} \tag{3.11}$$

Using equations (3.7) and (3.11) the discrete equation for the controller is given by

$$C(z) = \frac{U(z)}{E(z)} = k_p + k_i\left(\frac{1+\sigma}{T}\right)^{1-\lambda}\left(\frac{1+\sigma z^{-1}}{1-z^{-1}}\right)\sum_{i=0}^{\infty}\psi_i(1-\lambda)z^{-i} + k_d\left(\frac{1+\sigma}{T}\right)^{\mu}\sum_{i=0}^{\infty}\psi_i(\mu)z^{-i} \tag{3.12}$$

substituting $K_p = k_p, K_d = k_d\left(\frac{1+\sigma}{T}\right)^{\mu}, K_i = k_i\left(\frac{1+\sigma}{T}\right)^{1-\lambda}$ and limiting approximation length to L, then the controller equation is given as

$$C(z) = K_p + K_d\sum_{i=0}^{L}\psi_i(\mu)z^{-i} + K_i\left(\frac{1-z^{-1}}{1+\sigma z^{-1}}\right)\sum_{i=0}^{L}\psi_i(1-\lambda)z^{-i} \tag{3.13}$$

rewriting equation (3.13)

$$\begin{aligned}(1-z^{-1})C(z) =& K_p(1-z^{-1}) + K_d\sum_{i=0}^{L}\psi_i(\mu)z^{-i} + K_i(1+\sigma z^{-1})\\ &\sum_{i=0}^{L}\psi_i(1-\lambda)z^{-i}\end{aligned} \tag{3.14}$$

we know that

$$U(z) = C(z)E(z) \tag{3.15}$$

$$\begin{aligned}(1-z^{-1})U(z) =& K_p(1-z^{-1})E(z) + K_d(1-z^{-1})E(z)\sum_{i=0}^{L}\psi_i(\mu)z^{-i} +\\ & K_iE(z)(1+\sigma z^{-1})\sum_{i=0}^{L}\psi_i(1-\lambda)z^{-i}\end{aligned} \tag{3.16}$$

Converting equation(3.16) to discrete time domain

$$\begin{aligned}U(k) =& u(k-1) + K_p(\varepsilon(k)-\varepsilon(k-1)) + K_d(\varepsilon(k)-\varepsilon(k-1))\sum_{i=0}^{L}\psi_i(\mu)\\ &(\varepsilon(k-i)-\varepsilon(k-i-1)) + K_i\sum_{i=0}^{L}\psi_i(\mu)(\varepsilon(k-i)+\sigma\varepsilon(k-i-1))\end{aligned} \tag{3.17}$$

The generating function $\psi_i(.)$ can be approximated using PSE (power series expansion). Table 3.1 lists the approximation equations upto order 8. In the paper, we have considered Al-aloui approximation because it combines the advantages of Euler and Tustin operators.

3.2 Single Neuron FOPID (SNFOPID) controller architecture

Based on the equation (3.17), a single neuron FOPID architecture is developed. The detailed diagram of the proposed system is shown in figure 3.1. There are a total of 5 weights need to be tuned which are denoted as w_1, w_2, w_3, w_4, and w_5. These weights corresponds to the FOPID controller parameters K_p, K_i, K_d, λ, and μ, respectively.

In the figure 3.1, the terms $\varepsilon(k), \varepsilon(k-1), ..., \varepsilon(k-L)$ indicates the delayed error signals and $L \,\epsilon\, \mathbb{N}$ represents the length of approximation for the function ψ_i. Consider,

$$\zeta_1 = \varepsilon(k) - \varepsilon(k-1) \tag{3.18}$$

Table 3.1 Approximation of generating function $\psi_i(\alpha)$

Order(L)	Function	Approximation
0	$\psi_0(\alpha)$	1
1	$\psi_1(\alpha)$	-2α
2	$\psi_2(\alpha)$	$2\alpha^2$
3	$\psi_3(\alpha)$	$-\frac{4\alpha^3+2\alpha}{3}$
4	$\psi_4(\alpha)$	$\frac{2\alpha^4+4\alpha^2}{3}$
5	$\psi_5(\alpha)$	$-\frac{4\alpha^5+20\alpha^3+6\alpha}{15}$
6	$\psi_6(\alpha)$	$\frac{4\alpha^6+40\alpha^4+46\alpha^2}{45}$
7	$\psi_7(\alpha)$	$-\frac{8\alpha^7+140\alpha^5+392\alpha^3+90\alpha}{315}$
8	$\psi_8(\alpha)$	$\frac{2\alpha^8+56\alpha^6+308\alpha^4+264\alpha^2}{315}$

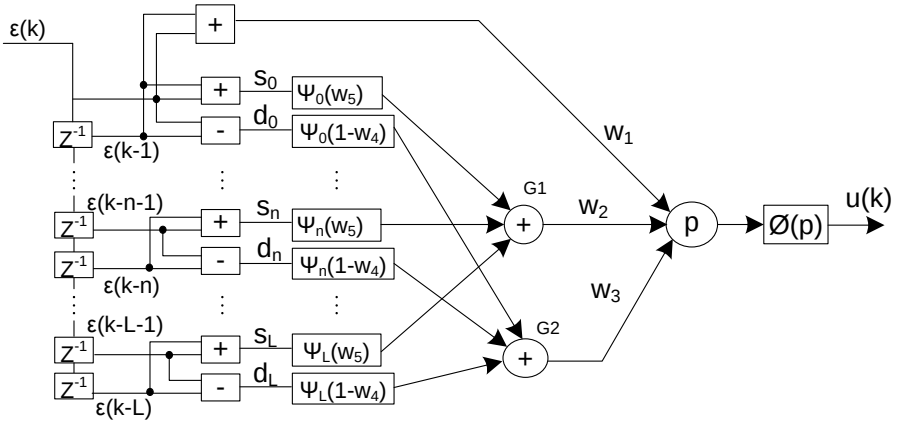

Figure 3.1 SNFOPID controller architecture

Let the signals produced from the sum and difference blocks are denoted by $s_0, s_1, s_2, ..., s_L$ and $d_0, d_1, d_2, ..., d_L$, respectively. The inputs for the gate G1 and gate G2 are $s_0 * \psi_0(w_5)$, $s_1 * \psi_1(w_5)$,...,$s_L * \psi_1(w_5)$ and $d_0 * \psi_0(1-w_4)$, $d_1 * \psi_1(1-w_4)$,...,$d_L * \psi_1(1-w_4)$, respectively. Then the output from the gate G1 is given by

$$\zeta_2 = \sum_{i=0}^{L} s_i * \psi_i(w_5) \tag{3.19}$$

Similarly, the output from the gate G2 is

$$\zeta_3 = \sum_{i=0}^{L} d_i * \psi_i(1 - w_4) \tag{3.20}$$

The outputs of ζ_2 and ζ_3 represents fractional integration and differentiation values, re-

spectively. The three inputs for the neuron p are w_1, $\zeta_1 * w_2$, and $\zeta_2 * w_3$. In figure 3.1, $\phi(p)$ indicates the tansigmoid activation function and its mathematical formula is given by

$$\phi(x) = \frac{1 - e^{-x}}{1 + e^{-x}}, \quad x \epsilon \mathbb{R} \tag{3.21}$$

The reason for choosing tansigmoid function is it can limit the output to the range [1,-1] which better suits for controlling applications than the other activation functions.

To verify the functionality of the proposed SNFOPID controller, it is subjected to optimize the DC motor response(as mentioned in section 3.3). Correspondingly the transfer function of the DC motor is given by equation (3.22).

$$G(s) = \frac{\omega(s)}{E_a(s)} = \frac{0.015}{(2.7s + 0.4)(0.0004s + 0.0022) + 0.00075} \tag{3.22}$$

Assuming the sampling period $T_s = 0.001$ and applying Euler's formula as defined in equation (3.23),

$$s \approx \left(\frac{1 - z^{-1}}{T_s}\right) \tag{3.23}$$

Correspondingly, the discretized transfer function of DC motor is

$$G(z) = \frac{Y(z)}{X(z)} = \frac{6.931e^{-06}z^{-1} + 6.918e^{-06}z^{-2}}{1 - 1.994z^{-1} + 0.9944z^{-2}} \tag{3.24}$$

Rewriting the equation (3.24)

$$Y(z)(1 - 1.994z^{-1} + 0.9944z^{-2}) = X(z)(6.931e^{-06}z^{-1} + 6.918e^{-06}z^{-2}) \tag{3.25}$$

The corresponding discrete time domain representation of D.C motor is

$$y(n) = 1.994y(n-1) - 0.9944y(n-2) + 6.913e^{-06}x(n-1) \tag{3.26}$$

The equation (3.26) is used as plant and to optimize the plant response SNFOPID controller is included in the control loop. To tune the parameters of proposed controller an optimization algorithm is required. The next section discusses about chaotic political optimization.

3.3 Chaotic Political Optimization (CHPO) Algorithm

Political optimizer (PO) is a social behavior inspired meta-hueristic algorithm developed by [116] based on different phases of politics in a country. According to the al-

gorithm, the important phases in the politics are represented as election campaign, party switching, election, and parliamentary affairs. The algorithm mimics the different roles of politicians to find the optimal solution. The important characteristic of population in the PO is each member plays dual role as constitution and political party. Therefore the members update their position based on constituency leader and party leader.

To study the effect of chaotic maps on the political optimizer algorithm, a total of 10 chaotic maps are used and listed in table A.1(appendix). The chaotic maps can produce statistically divergent and non-repeating random numbers. Therefore optimization algorithms often use these chaotic maps to improve their search capability [117, 118]. The chaotic maps are extremely sensitive to initial conditions. Therefore, a small change in initial value can lead to completely different behavior. In the proposed method, to increase the exploitation efficiency of existing algorithm chaotic maps are incorporated in the parliamentary affairs stage. This prevents the algorithm to trap in the local minima and improves the accuracy of the results.

3.3.1 Mathematical representation

The algorithm starts the optimization by initializing the population P. Let n represents number of constituencies, parties, and candidates in each party, then the corresponding population matrices are given by equations (3.27-3.31).

$$P = [P_1, P_2, P_3, ..., P_n] \tag{3.27}$$

$$P_i = [p_i^1, p_i^2, p_i^3, ..., p_i^n] \tag{3.28}$$

$$p_i^j = [p_{i,1}^1, p_{i,2}^2, p_{i,3}^3, ..., p_{i,d}^n] \tag{3.29}$$

$$C = [C_1, C_2, C_3, ..., C_n] \tag{3.30}$$

$$C_j = [p_i^1, p_i^2, p_i^3, ..., p_i^n] \tag{3.31}$$

Where P represents set of parties, P_i is set of candidates in the party i, p_i^j, represents candidate j of party P_i, or election candidate in a constitution, and each member in the party $p_{i,k}^j$ indicates potential solution of d dimensions. Where C represents set of constituencies and C_j represents the constituencies and its elements are election candidates or party members. The term d represents the number of variables to be solved in the optimization problem.

After initialization of population, fitness is evaluated for each of the candidate. This process is denoted by the general election stage (election between parties) and evaluated using

$$q = \underset{1\leq j\leq n}{\operatorname{argmin}} f(p_i^j) \quad \forall\, i = 1, 2, ..., n \tag{3.32}$$

Then the fittest members in the parties are denoted by p_i^* (party leaders) and the corresponding matrix is given by

$$P^* = [p_1^*, p_2^*, ..., p_n^*] \tag{3.33}$$

Similarly the winners from all the constituencies (c_j^*) are denoted by

$$C^* = [c_1^*, c_2^*, ..., c_n^*] \tag{3.34}$$

After general election of party leaders, to update the positions of the candidates the following equations are used. This stage is denoted as election campaign.

$$p_{i,k}^j(t+1) = \begin{cases} m^* + r(m^* - p_{i,k}^j(t)), & p_{i,k}^j(t-1) \leq p_{i,k}^j(t) \leq m^* \text{ or } p_{i,k}^j(t-1) \geq p_{i,k}^j(t) \geq m^* \\ m^* + (2r-1)|m^* - p_{i,k}^j(t)|, & p_{i,k}^j(t-1) \leq m^* \leq p_{i,k}^j(t) \text{ or } p_{i,k}^j(t-1) \geq m^* \geq p_{i,k}^j(t)\ l \\ m^* + (2r-1)|m^* - p_{i,k}^j(t-1)|, & m^* \leq p_{i,k}^j(t-1) \leq p_{i,k}^j(t) \text{ or } m^* \geq p_{i,k}^j(t-1) \geq p_{i,k}^j(t) \end{cases} \tag{3.35}$$

$$p_{i,k}^j(t+1) = \begin{cases} m^* + (2r-1)|m^* - p_{i,k}^j(t)|, & p_{i,k}^j(t-1) \leq p_{i,k}^j(t) \leq m^* \text{ or } p_{i,k}^j(t-1) \geq p_{i,k}^j(t) \geq m^* \\ p_{i,k}^j(t-1) + r(p_{i,k}^j(t) - p_{i,k}^j(t-1)), & p_{i,k}^j(t-1) \leq m^* \leq p_{i,k}^j(t) \text{ or } p_{i,k}^j(t-1) \geq m^* \geq p_{i,k}^j(t) \\ m^* + (2r-1)|m^* - p_{i,k}^j(t-1)|, & m^* \leq p_{i,k}^j(t-1) \leq p_{i,k}^j(t) \text{ or } m^* \geq p_{i,k}^j(t-1) \geq p_{i,k}^j(t) \end{cases} \tag{3.36}$$

where m^* represents the value of dimension k of the party leader $p_{i,k}^*$ and r denotes a random number in the range [0,1].

After the election campaign the algorithm enters party switching phase. Here an adaptive parameter $\lambda\ \varepsilon\ [0, \lambda_{max}]$ is considered to select the switching candidate p_i^j (with probability λ). The selected candidate p_i^j swaps position with least fit member p_r^q of some randomly chosen party P^j. The least fit member is calculated from the equation ((3.37)).

$$q = \underset{1\leq j\leq n}{\operatorname{argmax}} f(p_r^j) \tag{3.37}$$

Then election is conducted to identify the constituency winners. This is represented mathematically as

$$q = \underset{1\leq j\leq n}{\operatorname{argmin}} f(p_i^j) \tag{3.38}$$

Then, in the parliamentary affairs stage, the constituency winners / parliamentarians c_j^* updates their position by comparing with a randomly chosen member c_r^* using the equation (3.39). If the comparison improves the corresponding member position then they are updated, otherwise they retain their values.

$$C_j^* = C_r^* + (2\gamma - 1)|C_r^* - C_j^*| \tag{3.39}$$

Where γ is chaotic random number generator which produce random numbers in the range (0,1).

The algorithm runs through all the stages as mentioned until the termination criteria is met. The detailed description of the algorithm can be found in [116]. To identify the best map for the political optimizer algorithm, a total of 10 chaotic maps are considered and given in table A.10. The complete flow chart for CHPO algorithm is shown in figure 3.2. The detailed analysis related to effect of chaotic maps on the algorithm is presented in next subsection.

3.3.2 Benchmark functions

To investigate the effect of chaotic maps on the algorithm a total of 9 benchmark functions are considered, which include both uni-modal and multi-modal functions. The detailed definitions of bench mark functions are given in table A.2.

3.3.3 Statistical analysis

To analyze the effect of chaotic maps on the political optimizer algorithm, statistical parameters such as best, worst, mean, median, and standard deviation are considered. The comparison results of chaotic PO and PO algorithm for various benchmark functions are mentioned in table 3.2. The bold values indicates the best values produced for each benchmark function. From the table, it is found that the CHPO6 chaotic map produced better optimized values than the other maps. The results show that the chaotic maps improved the original algorithm by surpassing the best values of most of the benchmark functions. From the results, it is further observed that the CHPO6 algorithm significantly

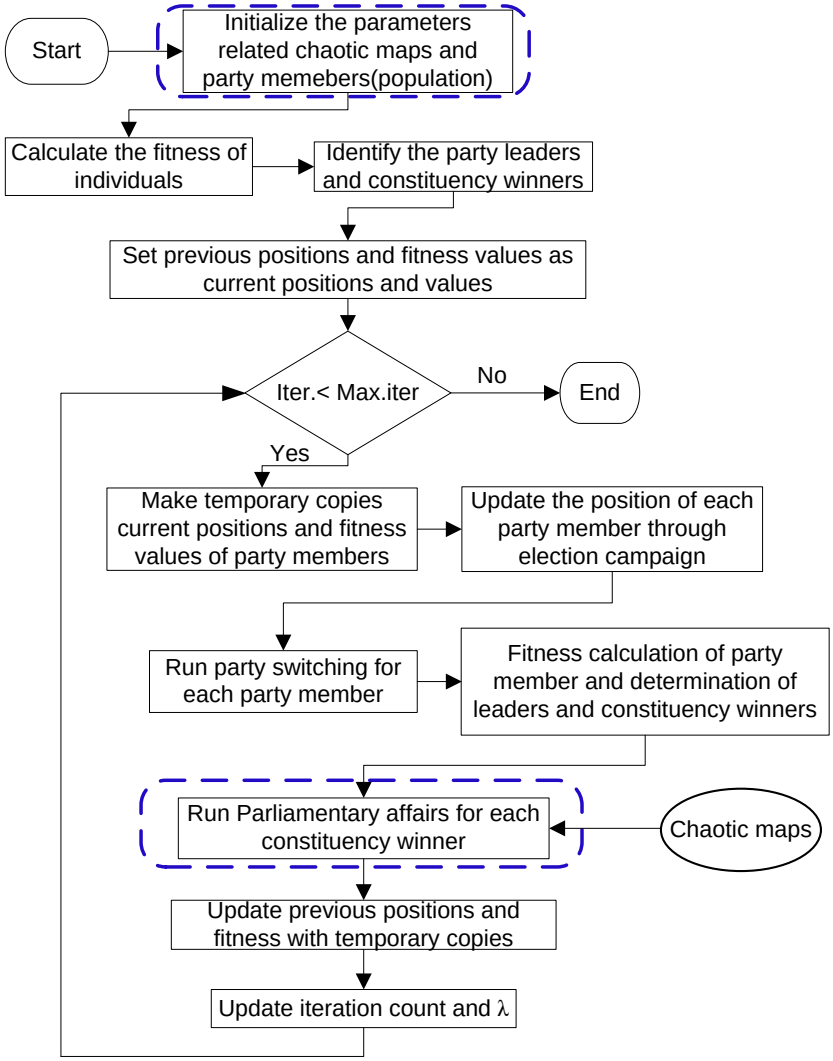

Figure 3.2 Proposed Chaotic Political Optimizer flow chart

improved PO performance.

3.3.4 Convergence curves

To further analyze behavior of different CHPO algorithms, the convergence curves are drawn. The convergence curves of various benchmark functions are shown in figure 3.3. For each benchmark function, 30,000 function evaluations are performed over 430 iterations. The curves are plotted by taking number of iterations on x-axis and log vale of cost function on the y-axis. The convergence curves shows that the CHPO based algorithms surpassed the original PO algorithm. It is also found that the use of chaotic

Table 3.2 Statistical analysis of benchmark functions

Algorithm	Parameter	F1	F2	F3	F4	F5	F6	F7	F8	F9
	Best	**7.35E-06**	2.27E-189	1.32E-237	-1.00E+00	2.17E-306	7.23E-188	9.00E-01	7.08E-204	3.00E+00
	Worst	1.02E-03	3.49E-173	5.71E-179	0.00E+00	9.57E-267	1.25E-177	1.00E+00	1.01E-191	3.00E+01
PO	Median	2.31E-04	3.76E-184	6.33E-208	0.00E+00	4.09E-288	4.69E-184	9.00E-01	3.77E-200	7.15E+00
	Mean	2.65E-04	1.40E-174	2.28E-180	-2.80E-01	3.83E-268	5.00E-179	9.32E-01	4.07E-193	1.07E+01
	SD	2.12E-04	0.00E+00	0.00E+00	4.58E-01	0.00E+00	0.00E+00	4.76E-02	0.00E+00	9.25E+00
	Best	3.14E-05	2.05E-168	2.67E-230	-1.00E+00	7.81E-281	1.26E-167	9.00E-01	4.64E-192	3.00E+00
	Worst	6.36E-04	2.78E-159	8.68E-184	0.00E+00	1.95E-240	2.34E-154	1.00E+00	6.70E-178	3.00E+01
CHPO1	Median	2.11E-04	9.47E-162	2.51E-206	0.00E+00	1.53E-263	3.02E-161	9.00E-01	3.45E-188	3.84E+00
	Mean	2.57E-04	3.69E-160	3.47E-185	-4.00E-01	7.81E-242	9.39E-156	9.44E-01	2.68E-179	8.22E+00
	SD	1.56E-04	7.19E-160	0.00E+00	5.00E-01	0.00E+00	4.68E-155	5.07E-02	0.00E+00	8.93E+00
	Best	**6.55E-06**	2.17E-197	8.22E-251	-1.00E+00	1.08E-312	4.62E-194	9.00E-01	5.31E-209	3.00E+00
	Worst	7.55E-04	8.59E-182	6.54E-198	0.00E+00	2.26E-268	4.60E-182	1.00E+00	8.05E-195	3.00E+01
CHPO2	Median	1.67E-04	2.86E-189	4.75E-220	0.00E+00	2.59E-298	4.60E-190	9.00E-01	1.95E-201	3.33E+00
	Mean	2.06E-04	3.50E-183	2.62E-199	-3.20E-01	9.04E-270	1.85E-183	9.48E-01	5.96E-196	5.90E+00
	SD	1.68E-04	0.00E+00	0.00E+00	4.76E-01	0.00E+00	0.00E+00	5.10E-02	0.00E+00	5.78E+00
	Best	2.35E-05	6.26E-186	6.14E-247	-1.00E+00	0.00E+00	8.33E-187	9.00E-01	3.82E-198	3.00E+00
	Worst	7.61E-04	4.86E-176	1.76E-194	0.00E+00	3.56E-270	4.16E-176	1.00E+00	4.06E-186	2.38E+01
CHPO3	Median	2.90E-04	1.61E-180	8.29E-223	0.00E+00	1.27E-286	3.98E-181	9.00E-01	1.16E-193	3.72E+00
	Mean	3.30E-04	3.86E-177	7.04E-196	-4.40E-01	1.43E-271	2.55E-177	9.40E-01	2.17E-187	6.44E+00
	SD	2.12E-04	0.00E+00	0.00E+00	5.07E-01	0.00E+00	0.00E+00	5.00E-02	0.00E+00	5.26E+00
	Best	3.44E-05	2.09E-187	1.07E-220	-1.00E+00	1.79E-307	1.89E-183	9.00E-01	5.46E-204	3.00E+00
	Worst	1.22E-03	3.79E-173	1.55E-165	0.00E+00	1.98E-262	6.60E-174	1.00E+00	2.00E-187	3.00E+01
CHPO4	Median	2.50E-04	1.76E-180	3.55E-201	0.00E+00	3.05E-285	1.74E-179	9.00E-01	3.02E-197	4.00E+00
	Mean	3.70E-04	1.54E-174	6.19E-167	-3.20E-01	7.92E-264	4.78E-175	9.40E-01	7.99E-189	7.80E+00
	SD	2.97E-04	0.00E+00	0.00E+00	4.76E-01	0.00E+00	0.00E+00	5.00E-02	0.00E+00	8.84E+00
	Best	3.15E-05	2.08E-181	1.47E-229	-1.00E+00	2.00E-301	3.54E-181	9.00E-01	3.92E-198	3.01E+00
	Worst	6.95E-04	2.55E-169	2.91E-178	0.00E+00	4.96E-262	3.19E-170	1.00E+00	7.13E-185	3.00E+01
CHPO5	Median	2.07E-04	1.86E-175	8.33E-210	0.00E+00	4.88E-286	3.84E-175	1.00E+00	1.94E-193	7.76E+00
	Mean	2.40E-04	1.45E-170	1.17E-179	-2.00E-01	2.04E-263	1.41E-171	9.52E-01	2.85E-186	1.14E+01
	SD	1.66E-04	0.00E+00	0.00E+00	4.08E-01	0.00E+00	0.00E+00	5.10E-02	0.00E+00	9.33E+00
	Best	4.19E-05	**1.50E-200**	**1.68E-273**	**-1.00E+00**	**0.00E+00**	**2.58E-200**	**9.00E-01**	**1.09E-215**	**3.00E+00**
	Worst	7.35E-04	4.49E-186	2.52E-210	0.00E+00	4.88E-301	2.52E-186	1.00E+00	1.23E-200	3.00E+01
CHPO6	Median	2.21E-04	1.75E-192	5.31E-241	0.00E+00	9.00E-322	9.57E-193	9.00E-01	8.37E-210	3.82E+00
	Mean	2.65E-04	1.85E-187	1.72E-211	-2.80E-01	1.95E-302	1.40E-187	9.40E-01	4.91E-202	5.53E+00
	SD	2.23E-04	0.00E+00	0.00E+00	4.58E-01	0.00E+00	0.00E+00	5.00E-02	0.00E+00	5.40E+00
	Best	5.58E-05	1.92E-182	1.07E-232	-1.00E+00	6.96E-296	3.69E-183	9.00E-01	3.24E-204	3.00E+00
	Worst	9.29E-04	9.00E-169	4.04E-184	0.00E+00	2.05E-263	2.73E-168	1.00E+00	3.82E-190	3.00E+01
CHPO7	Median	1.87E-04	1.09E-175	2.39E-205	0.00E+00	1.75E-281	3.62E-176	9.00E-01	6.92E-196	4.24E+00
	Mean	2.63E-04	4.77E-170	1.62E-185	-2.40E-01	8.33E-265	1.09E-169	9.36E-01	2.34E-191	9.23E+00
	SD	2.37E-04	0.00E+00	0.00E+00	4.36E-01	0.00E+00	0.00E+00	4.90E-02	0.00E+00	9.08E+00
	Best	3.61E-05	9.83E-178	1.87E-233	-1.00E+00	2.69E-303	5.38E-178	9.00E-01	2.38E-198	3.00E+00
	Worst	1.13E-03	2.20E-168	3.10E-179	0.00E+00	1.44E-262	1.51E-168	1.00E+00	1.30E-183	3.00E+01
CHPO8	Median	2.14E-04	1.70E-173	1.85E-212	0.00E+00	2.10E-280	5.47E-175	9.00E-01	1.66E-189	3.68E+00
	Mean	3.09E-04	1.06E-169	1.24E-180	-2.40E-01	5.77E-264	6.34E-170	9.44E-01	5.65E-185	6.82E+00
	SD	2.86E-04	0.00E+00	0.00E+00	4.36E-01	0.00E+00	0.00E+00	5.07E-02	0.00E+00	6.54E+00
	Best	1.65E-05	2.62E-190	3.23E-257	-1.00E+00	2.204e-321	3.93E-191	9.00E-01	3.10E-201	3.00E+00
	Worst	1.07E-03	3.28E-179	8.08E-197	0.00E+00	3.17E-278	3.14E-178	1.00E+00	1.78E-186	3.00E+01
CHPO9	Median	3.44E-04	5.07E-186	1.67E-230	0.00E+00	2.21E-300	4.00E-185	9.00E-01	3.98E-195	4.86E+00
	Mean	3.34E-04	1.32E-180	3.23E-198	-2.80E-01	1.95E-279	1.82E-179	9.48E-01	8.56E-188	1.05E+01
	SD	2.40E-04	0.00E+00	0.00E+00	4.58E-01	0.00E+00	0.00E+00	5.10E-02	0.00E+00	1.02E+01
	Best	2.37E-05	1.26E-189	1.43E-233	-1.00E+00	1.45E-314	3.61E-190	9.00E-01	2.82E-205	3.00E+00
	Worst	8.75E-04	1.79E-180	6.25E-173	0.00E+00	9.50E-274	5.88E-178	1.00E+00	1.01E-191	3.00E+01
CHPO10	Median	2.94E-04	2.92E-185	5.83E-214	0.00E+00	1.87E-295	1.87E-184	9.00E-01	4.42E-199	4.01E+00
	Mean	3.07E-04	1.59E-181	2.50E-174	-2.40E-01	3.80E-275	3.75E-179	9.40E-01	4.17E-193	9.18E+00
	SD	2.17E-04	0.00E+00	0.00E+00	4.36E-01	0.00E+00	0.00E+00	5.00E-02	0.00E+00	9.08E+00

maps reduced the number of iterations to produce optimum value.

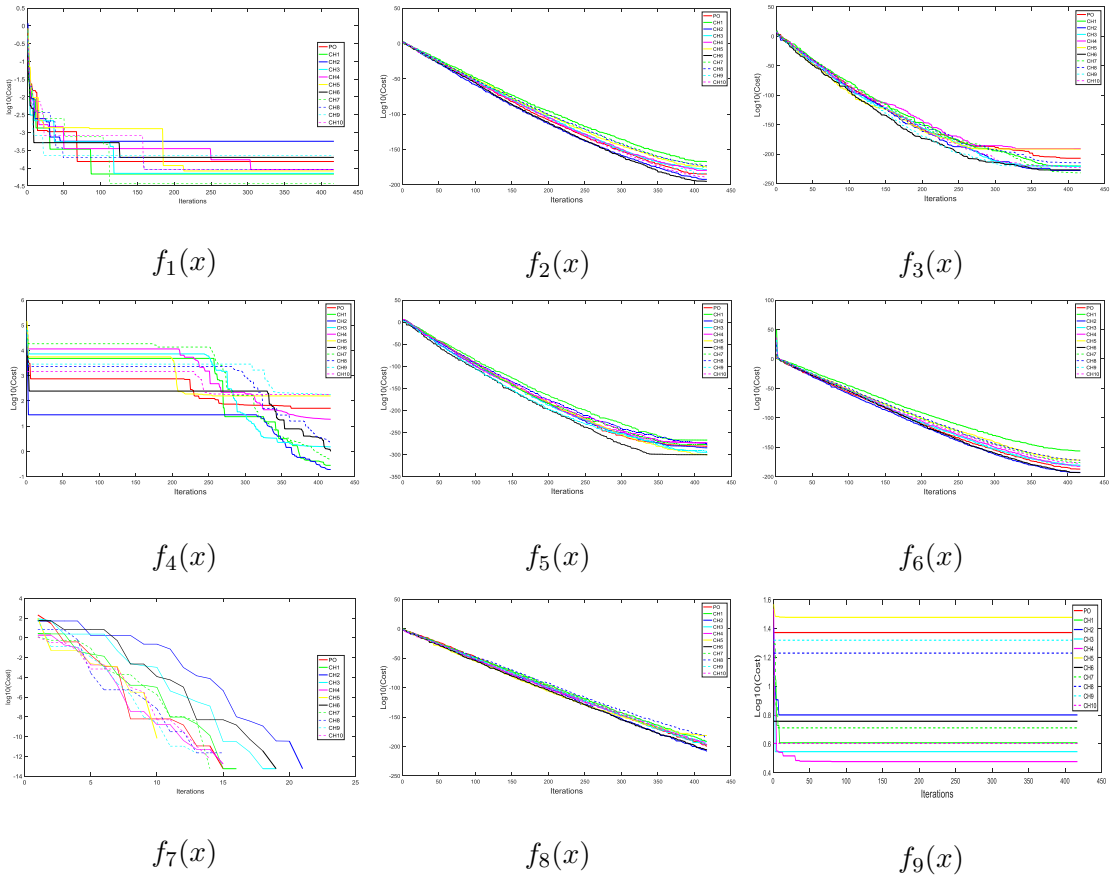

Figure 3.3 Comparison of convergence curves for benchmark functions

3.3.5 Wilcoxon ranksum test

Since statistical analysis and convergence behavior does not give complete behavior of an algorithm, to further analyze, Wilcoxon ranksum test is conducted by considering the best values produced in 25 consecutive runs. The p-values are obtained by comparing the CHPO algorithms with PO algorithm. In table 3.3, the values with $\leq$0.05 statistical significance level are indicated in bold. From the comparison of table 3.2 and table 3.3, it is observed that the Chebyschev chaotic map (CHPO6) produced lowest values for most of the benchmark functions.

Therefore to train the proposed SNFOPID controller, CHPO6 algorithm is used. The next section describes the analysis of simulation results when proposed controller parameters are optimized using CHPO6 algorithm.

Table 3.3 Wilcoxon's Ranksum test results of Chaotic PO algorithms. The proposed algorithms are compared with the actual PO algorithm for 25 consecutive runs and ranksum test is performed. The p-values which are ≤ 0.05 are highlighted with bold facing

Function	CHPO1	CHPO2	CHPO3	CHPO4	CHPO5	CHPO6	CHPO7	CHPO8	CHPO9	CHPO10
F1	**0.0144**	0.4332	0.4652	0.1517	0.4545	0.257	0.4438	0.331	0.4017	0.3811
F2	**6.34E-06**	**7.17E-06**	**4.19E-02**	**8.80E-03**	**6.34E-06**	**6.34E-06**	**2.53E-04**	**1.87E-05**	7.99E-02	2.32E-01
F3	5.85E-02	7.22E-02	**8.80E-04**	3.41E-01	2.66E-01	**1.60E-03**	3.41E-01	2.75E-01	**2.00E-03**	1.90E-01
F4	3.41E-01	1.93E-01	**3.96E-02**	3.91E-01	1.39E-01	1.71E-01	8.40E-02	4.23E-02	**1.67E-02**	7.99E-02
F5	**1.11E-04**	9.26E-02	1.58E-01	7.22E-02	**8.80E-03**	**6.34E-06**	8.40E-02	**1.34E-02**	**5.00E-03**	**1.00E-02**
F6	**6.34E-06**	**3.32E-05**	2.00E-01	6.17E-02	**1.03E-05**	**6.34E-06**	**1.23E-04**	**6.34E-06**	**0.0331**	7.00E-02
F7	1.73E-01	8.39E-01	1.62E-01	9.42E-02	1	2.44E-01	8.20E-01	4.97E-01	3.39E-01	6.70E-01
F8	**6.34E-06**	**3.31E-02**	**2.07E-04**	0.1335	**6.34E-06**	**3.72E-05**	**1.25E-02**	**6.34E-06**	**1.52E-04**	**3.52E-02**
F9	2.06E-01	2.44E-01	3.10E-01	**4.80E-02**	**9.40E-02**	1.02E-01	**4.19E-02**	1.00E-01	2.40E-01	**3.52E-02**

3.4 Speed control of DC motor using SNFOPID controller

3.4.1 Objective function and tuning of SNFOPID controller

The objective of the SNFOPID controller is to generate a control signal such that the system output optimally tracks the reference signal. To achieve this, we need to identify the weights of the controller $K_p(w_1), K_i(w_2), K_d(w_3), \lambda(w_4)$, and $\mu(w_5)$ in such a way that the control signal u(k) minimizes the error between actual system output and reference signal.

For optimal tuning of SNFOPID controller weights, an objective function is required. Since, identification of neural network weights involves fractional order differential equations, application of gradient descent algorithm increases the complexity and does not guarantee the optimal solution if the objective function consists of more than one minima. This can be solved by using non-gradient methods such as meta-heuristic algorithms. Therefore, in the proposed method CHPO algorithm is used to identify the SNFOPID controller weights. The process of tuning the SNFOPID controller parameters is represented as a block diagram and shown in figure 3.4.

In the case of traditional neural networks, MSE (mean square) is used. Since the proposed SNFOPID controller has to optimize the weights that produce best output from the system, cost functions like IAE (integral of absolute error), ITAE (integral of time multiplied absolute error), and ITSE (integral of time multiplied squared error) can be used. Even though these functions can reduce overall system error, they cannot optimize individual performances of the system like rise time, settling time, overshoot and steady

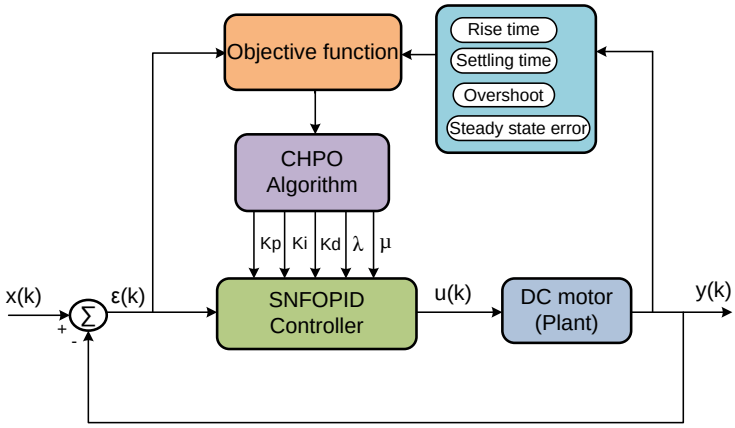

Figure 3.4 Tuning procedure for SNFOPID controller

state error. Therefore, in the proposed system we have developed a new cost function that optimizes overall system error as well as individual performance metrics. It uses the ITAE cost function to reduce the overall system error and the weighted combination of overshoot, rise time, and settling time are used to optimize the individual performances. The corresponding mathematical formula for the proposed cost function is given in the equation (3.40).

$$f = ITAE + \alpha.(Overshoot) + \beta.(Rise\ time + Settling\ time + |ess|) \tag{3.40}$$

Where the parameters α and β represents the weighing coefficients of objective function. After several trials, it is found that $\alpha = 10$ and $\beta = 2$ produced the optimum weights for the SNFOPID controller.

3.4.2 Working of SNFOPID controller

The identified weights are used in the realization of SNFOPID controller. To verify the controlling ability of designed controller, it is added in the control loop of DC motor as shown in figure 3.5. The working procedure of proposed system is as follows. Initially, the error signal $\varepsilon(k)$ is generated using equation (3.41).

$$\varepsilon(k) = x(k) - y(k) \tag{3.41}$$

Later, the delayed error signals are generated using delay blocks based on the order (L) of the generator function $\psi(\alpha)$. Then the signals are passed through sum and difference

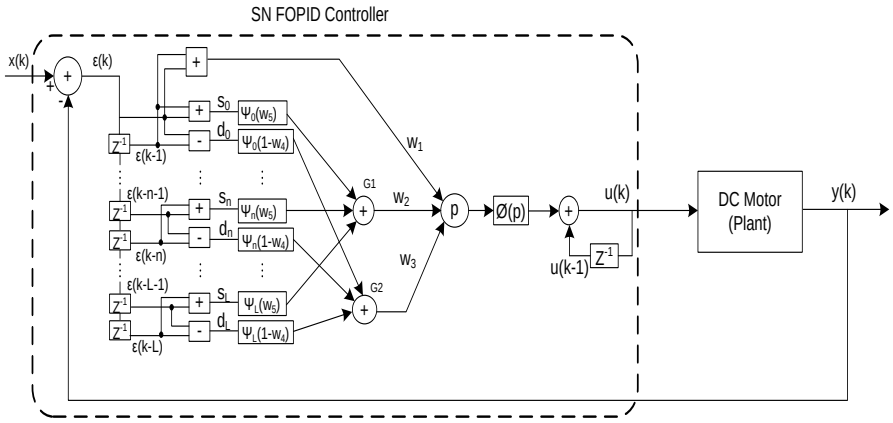

Figure 3.5 DC motor speed control using SNFOPID controller

blocks and fed as input to the neuron P. The neuron generates the output and the signal is activated by passing through tansigmoid function $\phi(x)$. This activated signal acts as control signal u(k) which drives the DC motor.

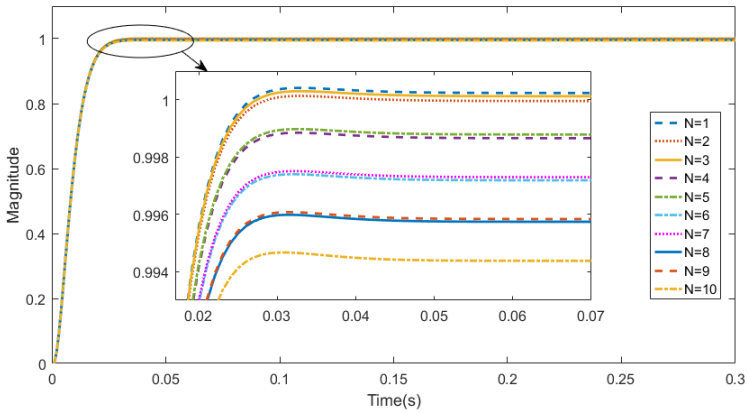

Figure 3.6 Effect of generating function order on controller response

3.5 Results and discussion

All the simulations are performed using MATLAB 2016a software on a system with Intel core i5 processor and 8GB RAM. The controller parameters are limited to the range as shown in table 3.4. The lower bounds and upper bounds are selected from the literature to maintain the consistency.

Table 3.4 Controller parameters range

Parameter	Lower limit	Upper limit
K_p	0.001	20
K_i	0.001	20
K_d	0.001	20
λ	0.001	1
μ	0.001	1

To identify the optimum parameters of SNFOPID controller step response is considered. To reduce the error between reference signal and system output, the network is trained using CHPO6 algorithm. The weights of the neural network are updated using the cost function described in equation (3.40).

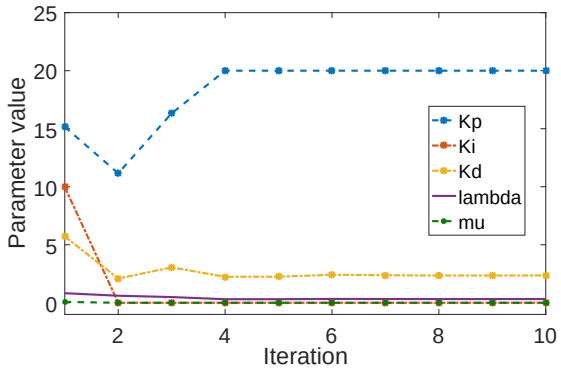

Figure 3.7 Convergence curves of SNFOPID controller weights

The identified weights of the SNFOPID controller (K_p, K_i, K_d, λ, and μ) are shown in table 3.6. Moreover, the proposed controller parameters are compared with OBL/HGSO-FOPID, MRFO-SA-FOPID, CHASO-FOPID, ASO-FOPID, ASO-PID, GWO-FOPID, GWO-PID, and SFS-PID controllers.

The response of the proposed system for different orders (L) of generating function is plotted as shown in figure 3.6. The convergence curves for the first 10 iterations of CHPO6 algorithm are shown in figure 3.7. To determine the order of the generating function, using the step response performance metrics of the controller for different orders are calculated. The corresponding results are mentioned in table 3.5 and figure 3.8. From

the analysis, it is found that the settling time and rise time are decreasing with increase in order. Whereas, the overshoot and steady state error are increasing with increase in order. Moreover, it is also found that even order shows better performance than the odd order. For the simulations, we considered generating function order (L) as 2, since the system has lowest steady state error for this order.

Table 3.5 Effect of generating function order on controller performances

Order(L)	Rise time (ms)	Settling time (ms)	Overshoot(%)	Steady-state error
1	13.795	23.8137	0.0243	-1.71E-04
2	13.7943	23.8188	0.018	4.60E-05
3	13.7959	23.8202	0.0196	-1.03E-04
4	13.7663	23.7499	0.0259	0.0014
5	13.7767	23.7931	0.043	0.0011
6	13.7299	23.6592	0.0341	0.0029
7	13.7479	23.738	0	0.0024
8	13.6885	23.5507	0.0422	0.0044
9	13.7125	23.6593	0	0.0038
10	13.6441	23.4255	0.0512	0.0058

Since direct observation of these values doesn't give an idea of controller behavior, step, load, disturbance, and sinusoidal response analysis are carried out.

3.5.1 Step response

To check the controlling ability of the SNFOPID controller, its step response is compared with various existing FOPID/PID controllers. The step responses are calculated using the controller parameter values mentioned in table 3.6. Figure 3.9 and 3.10 shows the comparison of responses obtained. For all the systems, performance metrics like rise time, settling time, overshoot, and steady state error are calculated as shown in table 3.6. From the table, it is found that the proposed SNFOPID controller produced best rise time 0.0137s and settling time 0.0238s with negligible overshoot. SNFOPID controller performance is compared with the state-of-the-art FOPID/PID controllers like MRFO-SA-FOPID, CHASO-FOPID, ASO-FOPID, GWO-FOPID, OBL/HGSO-PID, HGSO-PID, and GWO-PID controllers.

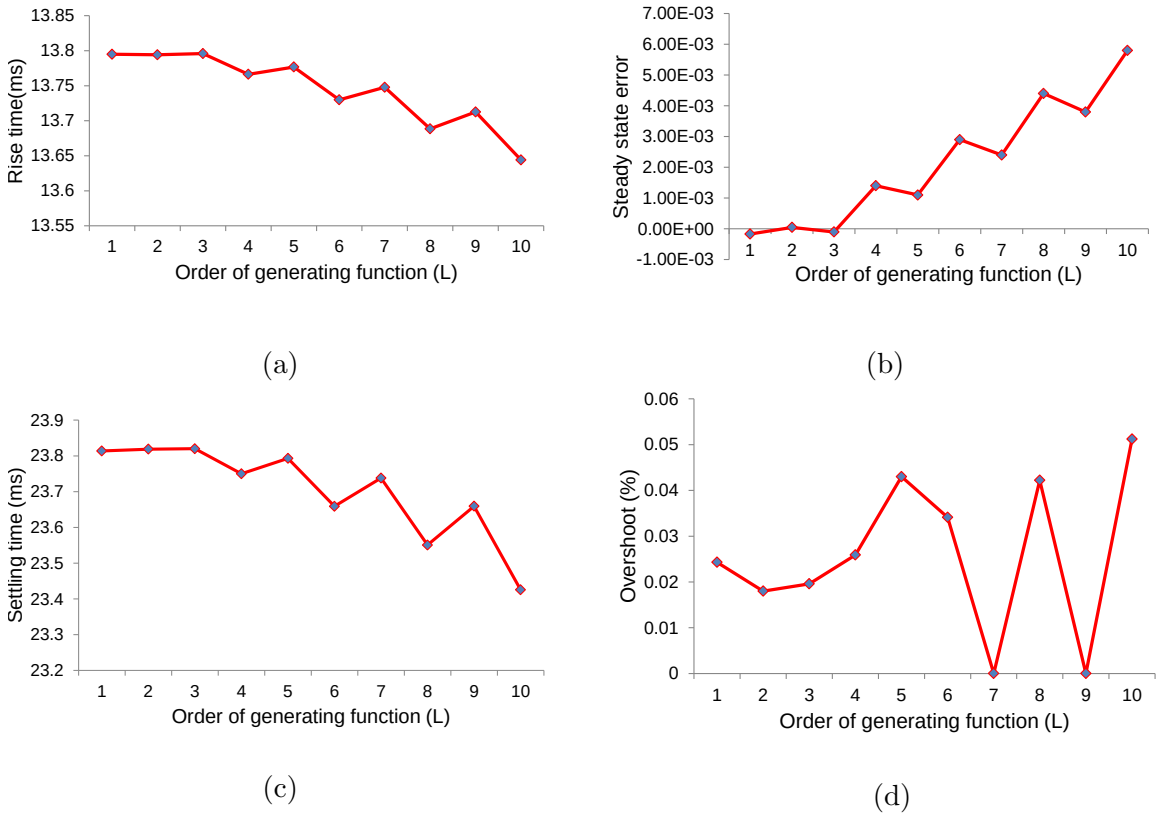

Figure 3.8 Effect of generating function order on (a) Rise time (b) Steady state error (c) Settling time (d) Overshoot

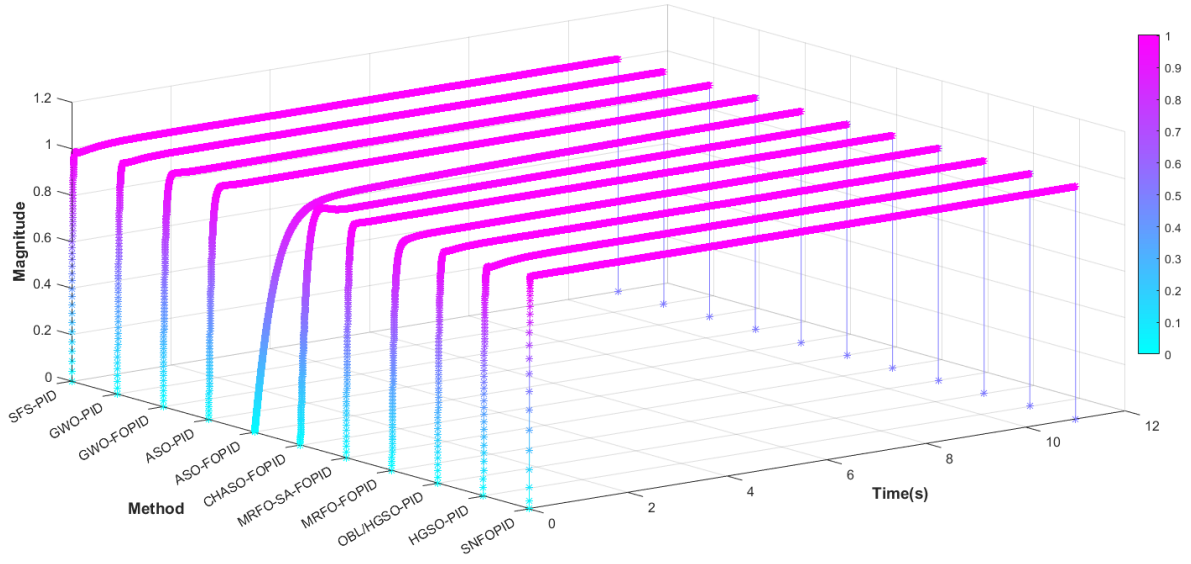

Figure 3.9 Step response comparison of FOPID controllers

Table 3.6 Comparison of FOPID controller parameters

Objective function	K_p	K_i	K_d	λ	μ	Rise time (ms)	Settling time (ms)	Overshoot(%)	E_{ss}
ITAE	20	19.99	20	1.53102	5.24e-03	1.8245	4.4529	0	0
ITSE	20	19.99	4.57	1.11463	0.319084	1.7754	6.997	0	0
ZLG	5.26452	3.6557	20	1.31267	0.817401	0.7942	1.9729	0	0

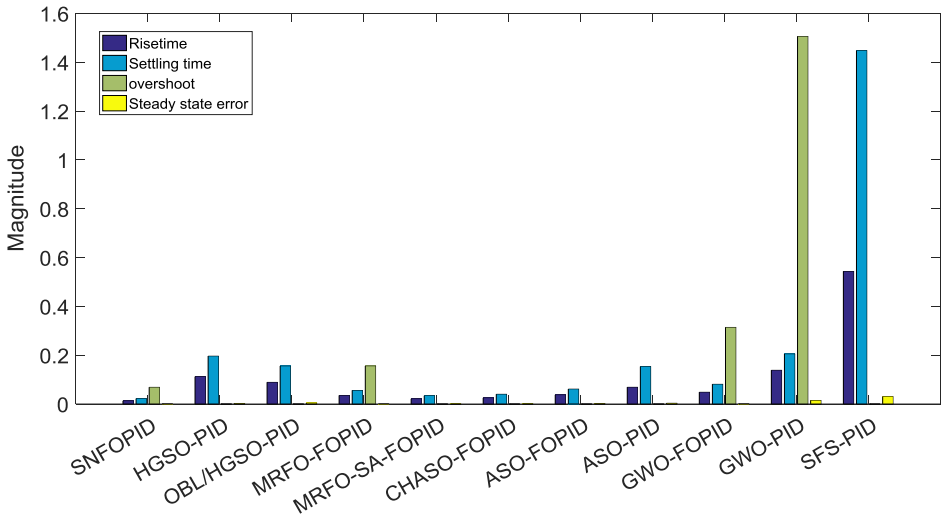

Figure 3.10 Performance comparison of FOPID controllers

3.5.2 Load response

To further investigate the controller behavior, different set points are given to the SNFOPID controller. All the set points are tracked by the proposed controller efficiently. On comparison with other state-of-the-art techniques the proposed controller tracked the set points efficiently.

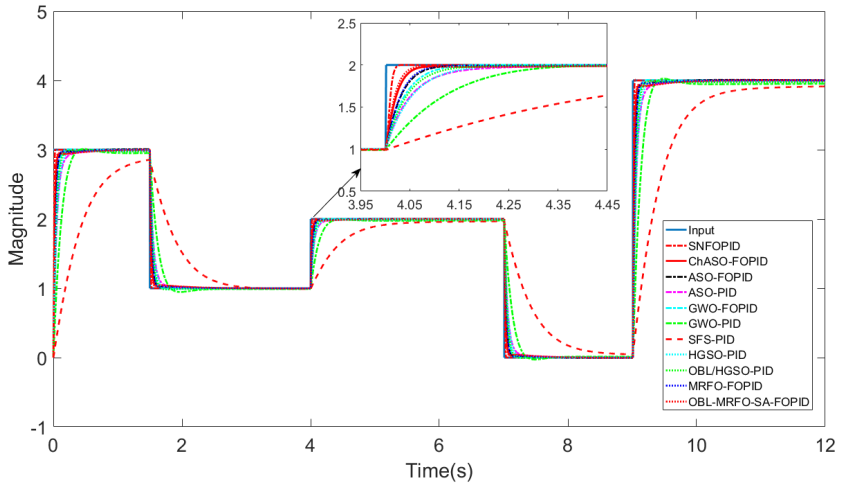

Figure 3.11 Comparison of FOPID controllers for set point changes

The controller behavior and the error signals are plotted in figure 3.12 when tracking the reference signal. From the figure 3.12, it is observed that the controller performance

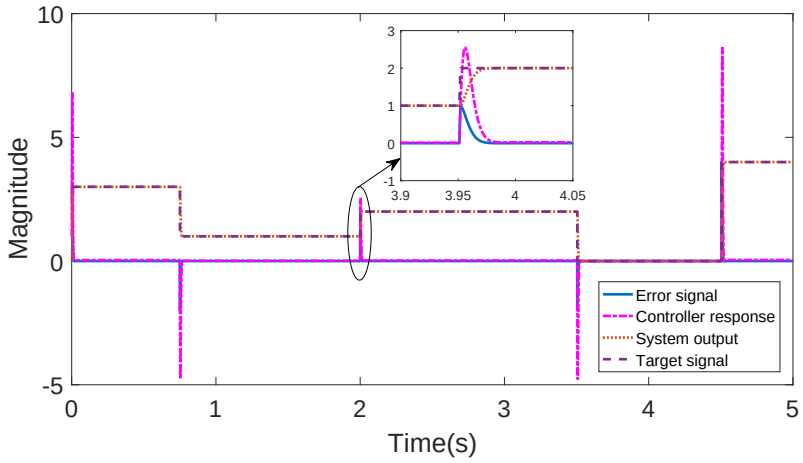

Figure 3.12 SNFOPID controller response for setpoint changes

is smooth and tracked the set points with minimum effort. Moreover, the responses of various controllers are also compared with the SNFOPID controller. The proposed controller produced good tracking behavior than the other controllers.

3.5.3 Disturbance response

Sudden disturbances in the input can lead to system instability. Therefore to check the controller response for abrupt changes in the input, disturbance analysis is carried out. Sudden impulses are added to the actual step signal and the controller response is plotted as shown in figure 3.13. The proposed SNFOPID controller has shown good response for disturbances. When compared with others the SNFOPID controller, it recovered early from the sudden disturbances.

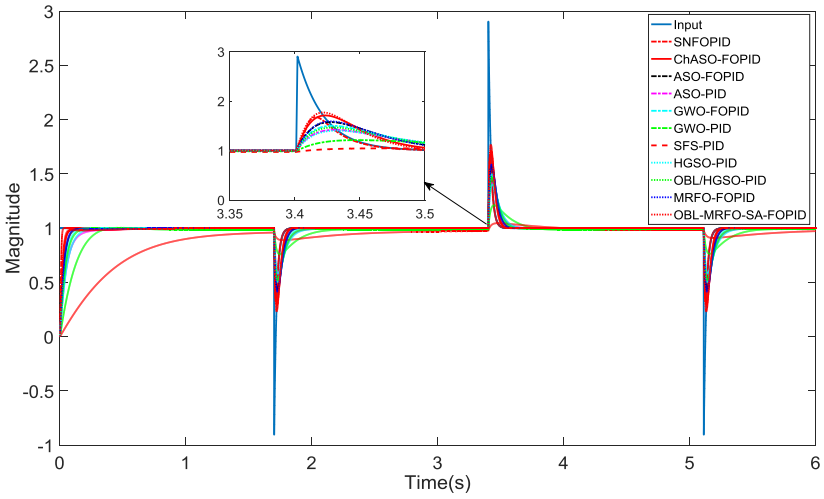

Figure 3.13 Disturbance response comparison of FOPID controllers

3.5.4 Sinusoidal response

To observe behavior of the controller for continuous changes in the set point, sinusoidal response is considered. The figure 3.14 represents reaction of different controllers for sinusoidal reference signal.

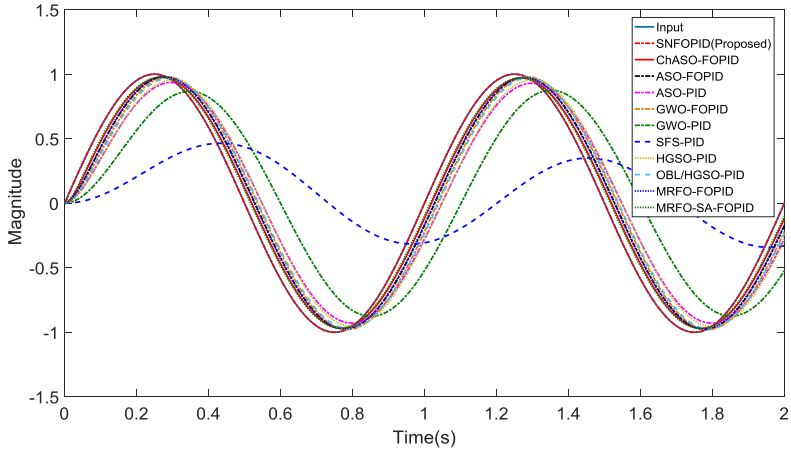

Figure 3.14 Sinusoidal response comparison of FOPID controllers

From the analysis, it is found that the proposed SNFOPID controller has given the best tracking performance than the existing FOPID/PID controllers. The error between actual signal and the SNFOPID controller response is minimum when compared with other controllers.

3.5.5 Quantitative analysis

Quantitatively the MSE (mean of squared error) values for step, load, disturbance, and sinusoidal signal responses of different controllers are given in table 3.7. The best values are indicated in bold. The corresponding pictorial representation is given in figure 3.15.

Table 3.7 Comparison of MSE values for step, load, disturbance and sinusoidal responses

Controller	Step	Load	Disturbance	Sinusoidal
SNFOPID(Proposed)	**4.65E-04**	**2.21E-02**	**8.6E-06**	**5.45E-07**
HGSO-PID [57]	2.25E-03	7.27E-02	1.54E-02	4.53E-02
OBL/HGSO-PID [57]	1.82E-03	5.82E-02	1.39E-02	2.95E-02
MRFO-FOPID [58]	1.32E-02	4.24E-02	1.23E-02	1.40E-02
OBL-MRFO-SA-FOPID [58]	8.41E-04	2.66E-02	9.30E-03	5.48E-03
ChASO-FOPID [2]	9.66E-04	3.08E-02	1.02E-02	7.52E-03
ASO-FOPID [2]	1.35E-03	4.40E-02	1.23E-02	1.59E-02
ASO-PID [2]	2.17E-03	7.11E-02	1.50E-02	4.43E-02
GWO-FOPID [1]	1.64E-03	5.38E-02	1.34E-.02	2.52E-02
GWO-PID [1]	4.98E-03	1.61E-01	2.26E-02	1.77E-01
SFS-PID [52]	1.88E-02	5.97E-01	4.77E-02	4.63E-01

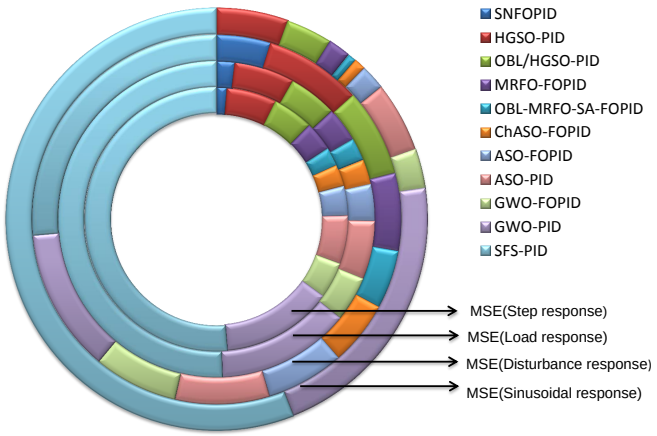

Figure 3.15 Comparison of MSE values for different responses

3.5.6 Response for noisy reference

Finally, to study the effect of noise in target signal, a noisy reference signal is considered. Then the SNFOPID controller response along with error and system response are plotted. The figure 3.16 shows that SNFOPID controller is able to track the reference input under continuous fluctuations. It is also observed that the system is stable and the controller quickly responds to small changes in the target signal.

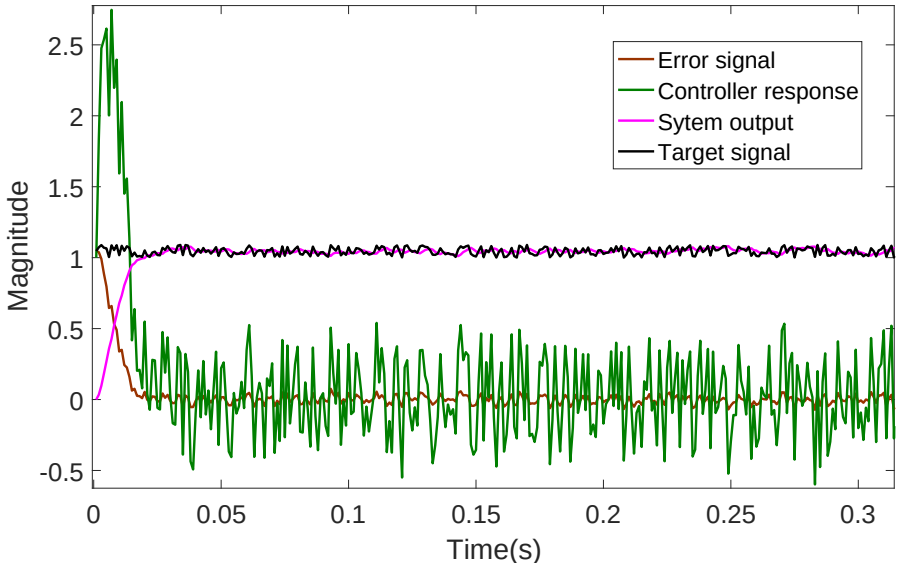

Figure 3.16 Response of SNFOPID controller for noisy target

3.5.7 Summary

Two different kinds of techniques mentioned in chapter 2 and chapter 3. On comparison of these techniques it is found that the NARXnet based controller produced better rise time, settling time, and overshoot values whereas SNFOPID based controller produced better steady state response. Overall, it can be said that the NARXnet based controller produced better response than the SNFOPID controller. But it is also mentioned that the behavior of neural network based controllers is unpredictable under heavy disturbances and changes in operating conditions. The performance of these controllers entirely depends on the training data. For the reliable performance neural networks need sufficient training data that covers different operating conditions. Since SNFOPID controller does not uses the training data related to either plant or controller, as mentioned shown in the

disturbance and noise analysis (section 3.5.3 and section 3.5.5) the SNFOPID controller is more reliable.

3.6 Conclusion

A novel architecture based on neural networks for realization of fractional order PID controllers (FOPID) has been presented. The controller is developed from the standard discretized fractional order controller equation. For the descretization purpose Al-aloui operator is used as it combines the advantages of Tustin and Euler methods. To optimize the controller parameters a novel Chaotic Political Optimizer (CHPO) is developed by hybridizing chaotic maps with the political optimizer algorithm. To analyze the performance of single neuron FOPID (SNFOPID) controller, a DC motor has been considered as plant and the proposed controller is subjected to optimize its response. Moreover, a new objective function is defined with the combination ITAE and system performance measures to optimize system response. To check the reliability of the controller, it is tested with step input, load changes, sudden disturbances, and sinusoidal tracking signals. In all the cases the response of the SNFOPID controller is compared with other controllers. The simulation results show that the proposed controller produced better performances than the existing techniques. Moreover, for the sinusoidal input signal, it is observed that the SNFOPID controller showed very good tracking behavior.

Chapter 4

Tuning of FOPID controller for AVR system using Chaotic Black Widow Optimization (ChBWO) algorithm

This chapter presents optimization of fractional order PID controller parameters to obtain the desired response from the automatic voltage regulator (AVR) system. Initially, the FOPID controller parameters are tuned using a meta-heuristic algorithm called Black Widow Optimization(BWO) algorithm. Later, chaotic maps are introduced into the original BWO algorithm to improve its convergence behavior. The improved Chaotic BWO is used to optimize FOPID controller for AVR system.

4.1 Automatic Voltage Regulator (AVR) System

Synchronous generators are commonly used in power generation systems. Due to the variations in the load or sudden changes in power usage, the generators produce oscillations at the output for a significant amount of time. These oscillations may lead to system instability and can cause catastrophes. To improve the terminal voltage stability, generators are controlled by excitation systems and AVR systems. Various constituents of the AVR system are amplifier, exciter, generator, and sensor [68]. The interconnections of various system blocks were shown in figure 4.1.

Initially, the generator terminal voltage given to the sensor circuit, converts the

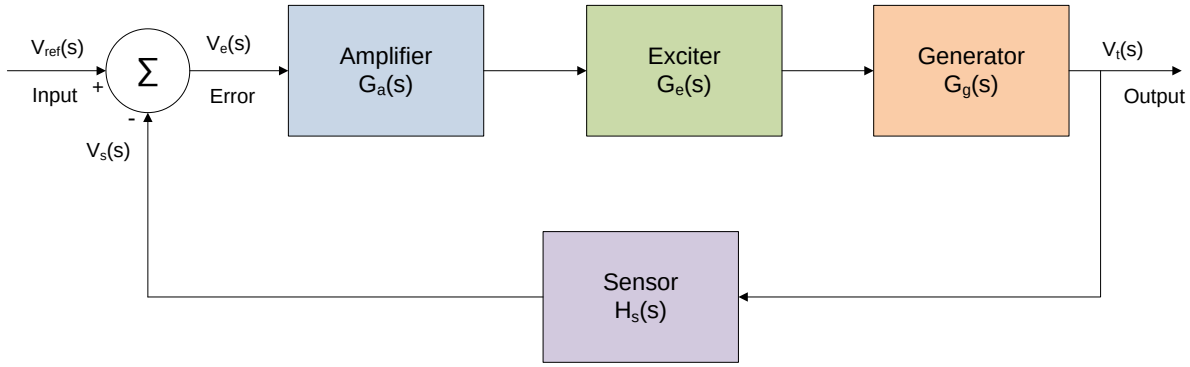

Figure 4.1 AVR system block diagram

terminal voltage into a proportional voltage signal. Then this signal is subtracted from the reference voltage and the error signal will be generated. The amplifier improves the strength of the error signal and the output of the amplifier connected to the exciter. The exciter converts the input signal into a suitable form to drive the generator. The corresponding mathematical representations of various blocks in the AVR system are given by the following equations.

The transfer function of the amplifier is represented as

$$G_a(s) = \frac{K_a}{1 + s\tau_a} \tag{4.1}$$

Where K_a is amplifier gain which has values in the range [100,400] and τ_a is the time constant of the amplifier and lies in the range [0.02, 0.1]. The transfer function of the exciter iss represented as

$$G_e(s) = \frac{K_e}{1 + s\tau_e} \tag{4.2}$$

Where K_e is exciter gain which has values in the range [10,400] and τ_e is the time constant of the exciter and lies in the range [0.5, 1.0]. The transfer function of the generator is represented as

$$G_g(s) = \frac{K_g}{1 + s\tau_g} \tag{4.3}$$

Where K_g is generator gain which has values in the range [0.7, 1.0], and τ_g is the time constant of the generator and lies in the range [1.0, 2.0]. The transfer function of the sensor is represented as

$$H_s(s) = \frac{K_s}{1 + s\tau_s} \tag{4.4}$$

Where K_s is exciter gain which has values in the range [1.0, 2.0], and τ_s is the time constant of the sensor and lies in the range [0.001, 0.06]. The range of various parameters of AVR system are chosen from the literature [66, 68, 73, 75] and summarized as shown in table 4.1. To understand the behavior and dynamics of the system its step response

Table 4.1 Range of modeling parameters of AVR system

Parameter	Description	Range	Value
K_a	Amplifier gain	[10,400]	10
τ_a	Amplifier time constant	[0.02,1]	0.1
K_e	Exciter gain	[1,10]	1
τ_e	Exciter time constant	[0.4,1]	0.4
K_g	Generator gain	[0.7,1]	1
τ_g	Generator time constant	[1,2]	1
K_s	Sensor gain	[1,2]	1
τ_g	Sensor time constant	[0.001,0.06]	0.01

is plotted in figure 4.2. From the step response key performance parameters identified. Table 4.2 shows the variation of key parameters with a change in K_g value. Since the terminal voltage varies with load changes, different values of K_g were considered in the range [0.7, 1.0] for step response. The gain margin and phase margin were calculated from the bode plot mentioned in figure 4.3.

Table 4.2 Identified key performances for AVR system

Parameter	K_g=0.7	K_g=0.8	K_g=0.9	K_g=1
Rise Time	32.021	29.73	27.87	0.26
Settling Time	423.08	472.15	520.11	7.02
Steady State Error	0.125	0.11	0.1	0.09
Peak Overshoot	47.85	52.94	57.61	65.21
Gain Margin	1.74	1.39	1.125	1.91
Phase Margin	17.6	9.71	3.19	2.32

The step response has 0.26s rise time, 7.02s settling time, and with overshoot value 65.21%. The system has very high overshoot and settling time. Therefore,to optimize the response of AVR system, a controller is required.

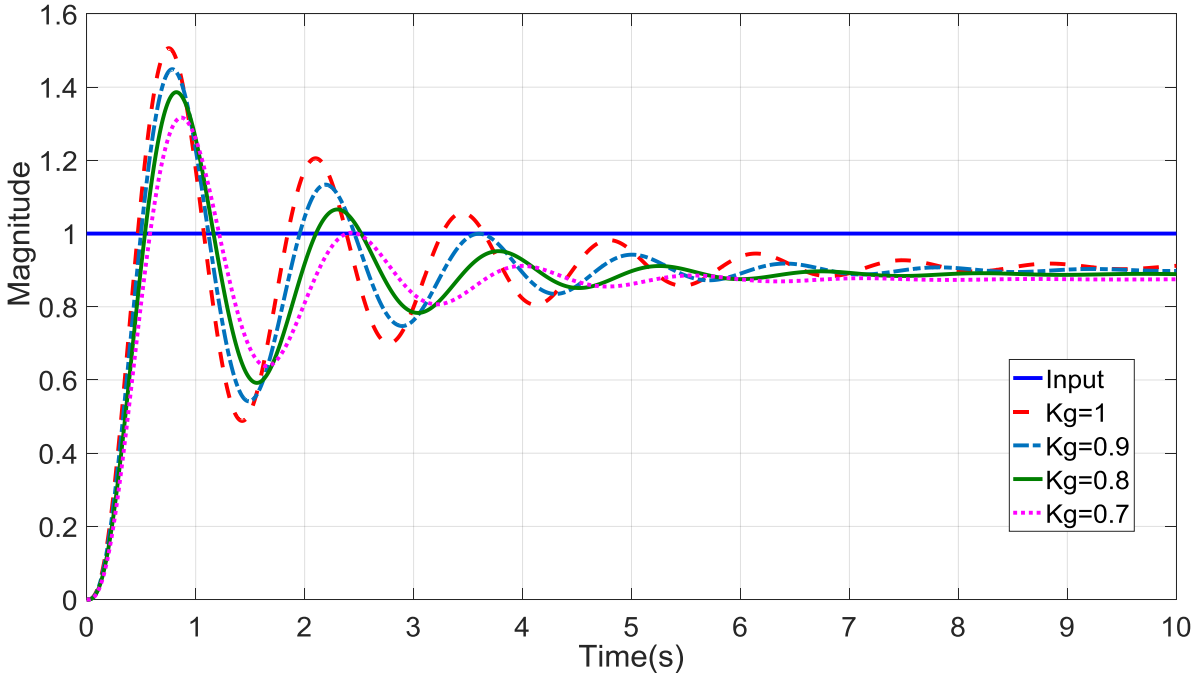

Figure 4.2 Step response variation of AVR system

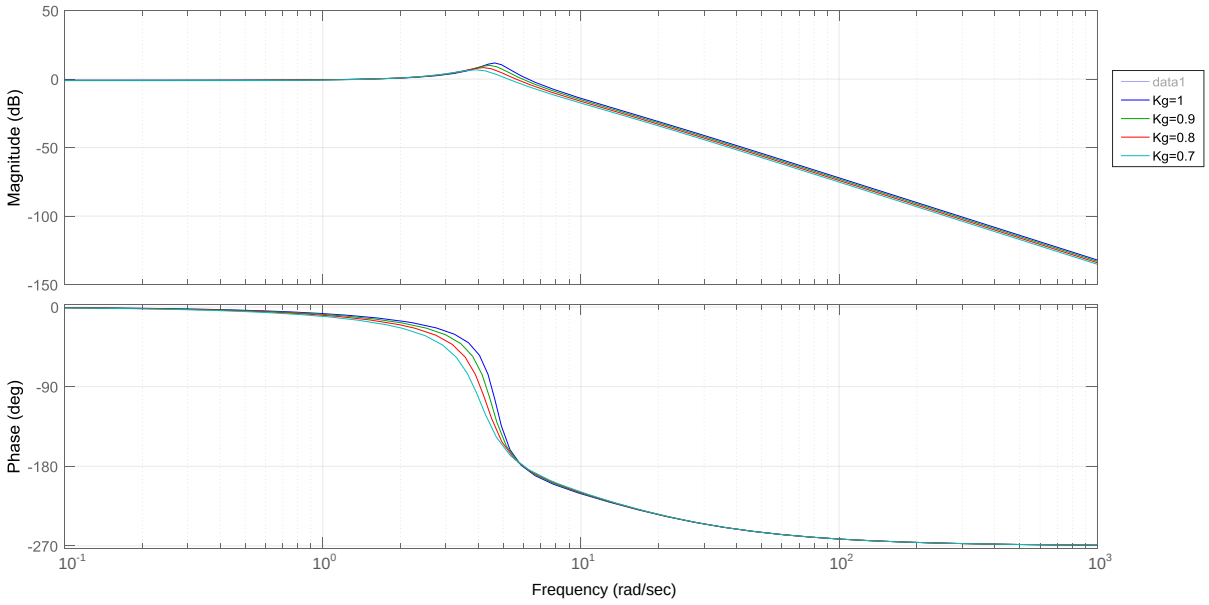

Figure 4.3 Bode plot for AVR system

4.2 Black Widow Optimization algorithm

Black widow optimization (BWO) proposed in [119], is a meta-heuristic algorithm developed based on the lifestyle of black widow spiders to solve engineering optimization

problems. An important behavior of black widow spiders is cannibalism. Due to this, only stronger spiders survive and produce the next generation. During the mating process, the female spider exhibits sexual cannibalism and consumes male spider. Thereafter, the female spider lays eggs in her nest and spiderlings will hatch within 8-11 days. The hatched spiderlings exhibit sibling cannibalism in which the weak siblings are consumed by stronger ones. Under special conditions, the siblings may consume the mother entirely. A reference picture related to black widow spiders and spiderlings is given in figure 4.4. The total process is developed as an optimization algorithm that consists of 4 stages known as initialization, procreation, cannibalism, and mutation.

Figure 4.4 Black widow spider and spiderlings

4.2.1 Initial population generation

In the algorithm, a variable in the solution space is called a widow and the solution is called black widow spider. For an m_{var} dimensional optimization problem, the algorithm starts by generating initial widow matrix with population of $n_{pop} \times m_{var}$. Then fitness is evaluated for each element in the widow matrix using

$$val = f(x_1, x_2, ...x_{m_{var}}) \tag{4.5}$$

where f is fitness function and $x_1, x_2, ...x_{m_{var}}$ represents dimensional variables.

4.2.2 Procreation and Cannibalism

The next generation of population (offspring) are generated from the initial population using the process of mating. This is implemented in the algorithm with the help of

ζ matrix of length m_{var} whose elements are random numbers having length of m_{var}. If x_1 and x_2 are parents then the offspring z_1 and z_2 are generated using equations (4.6) and (4.7).

$$z_1 = \zeta * x_1 + (1 - \zeta * x_2) \tag{4.6}$$

$$z_2 = \zeta * x_2 + (1 - \zeta * x_2) \tag{4.7}$$

The number of parents participating in procreation is chosen based on procreation rate (PR). The process is repeated for $m_{var}/2$ times without repetition of parents. The best of offspring and parents are selected using fitness calculation. Evaluation of fitness and identifying best spiders are similar to sexual cannibalism and sibling cannibalism, where only stronger ones survive. The number of spiders showing cannibalism are decided based on cannibalism rate (CR).

4.2.3 Mutation

Mutation increases the exploration capabilities of the algorithm. In the BWO optimization, mutation is performed based on mutation rate (MR), which describes the number of spiders to be mutated. In the proposed algorithm, swap mutation is used to preserve the adjacent position information. For the mutation, two random positions are selected in the widow matrix and their positions are exchanged to generate a new spider. The diagram representing the mutation process is shown in figure 4.5.

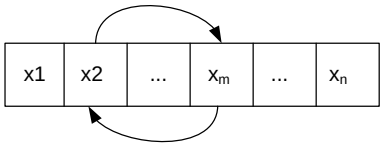

Figure 4.5 Mutation operation

In the proposed algorithm, the procreation rate (PR), cannibalism rate (CR), and mutation rate (MR) are chosen as 0.6, 0.44, and 0.4, respectively [119]. It is mentioned that the effect of PR, CR, and MR on convergence behavior of BWO algorithm can be considered for future scope. The complete flow chart of the BWO algorithm is shown in figure 4.6.

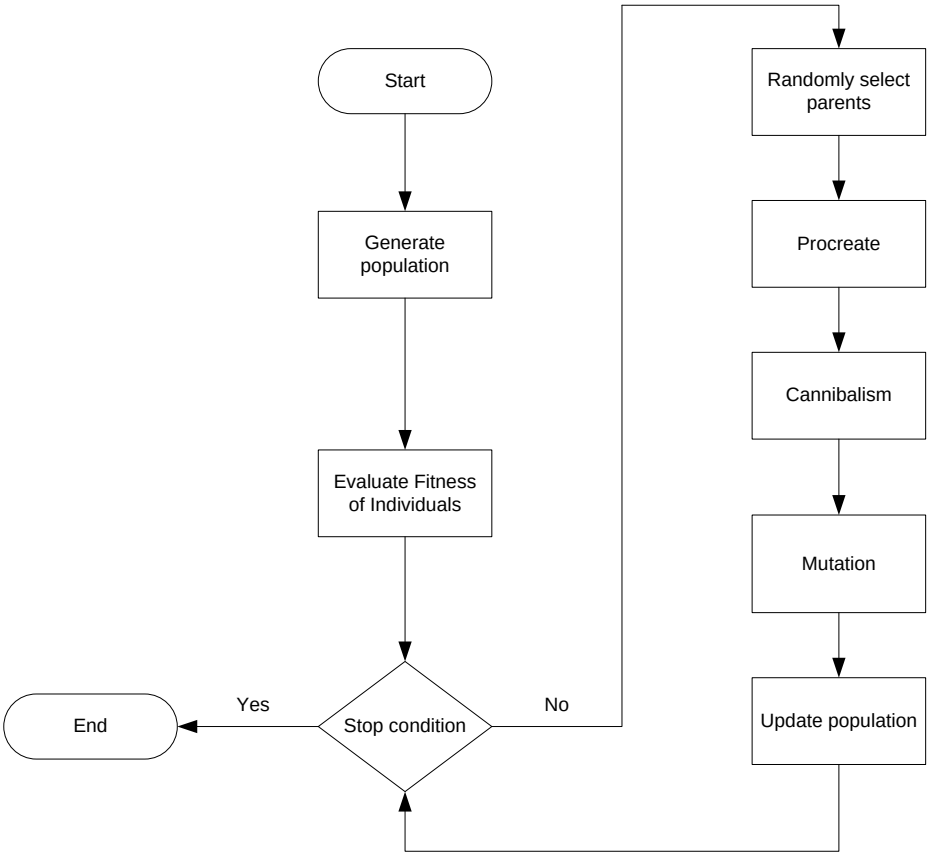

Figure 4.6 Flow chart of BWO algorithm

4.3 Proposed BWO-FOPID controller

The BWO-FOPID controller provides two additional degrees of freedombwhen compared to existing PID controllers. This allows designing a robust controller for a given application. The process to tune the FOPID controller using the BWO algorithm was represented as a block diagram in figure 4.7. $V_{ref}(s)$ is the reference voltage that should be maintained by the AVR system at its terminals. $V_t(s)$ is the actual terminal voltage produced by the system. $V_e(s)$ is error voltage, which indicates the difference of $V_{ref}(s)$ and $V_t(s)$. For each iteration of the BWO algorithm, a population of K_p, K_i, K_d, λ, and μ are generated and used to find the objective function value. FOPID controller takes $V_e(s)$ as the input signal and produces the corresponding control signal. For this signal, the AVR system terminal voltage and error are calculated. The process is repeated until the termination criteria were met. Finally, the best values of the parameters will be identified and are used to design the optimum FOPID controller.

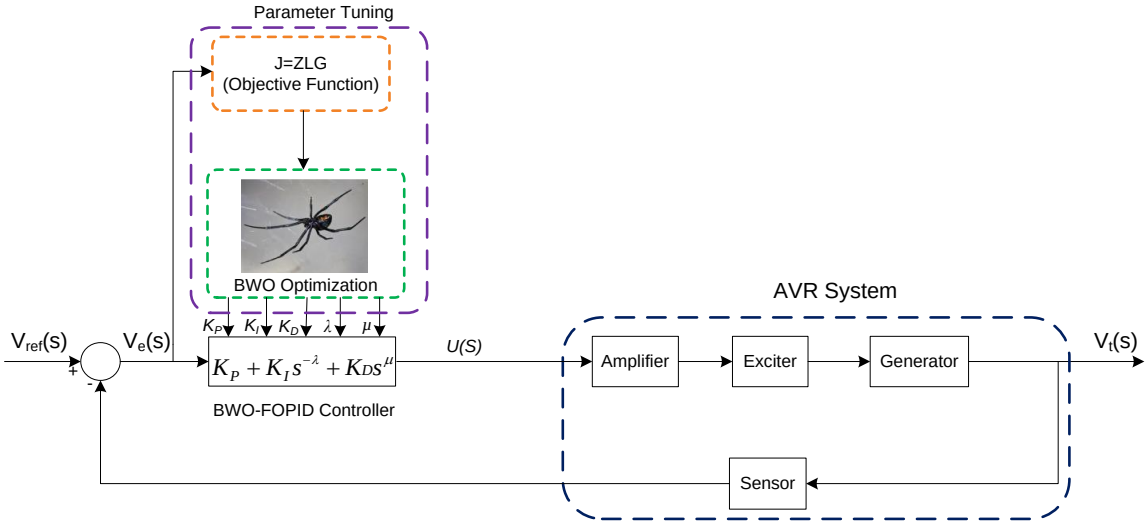

Figure 4.7 Proposed BWO-FOPID controller block diagram

The designed controller was inserted into the system. The controller output is given as input to the AVR system. The terminal voltage of AVR was again compared with the reference voltage and the error signal is produced. The process is repeated until the error signal becomes zero. When the desired level was reached, the controller produces a constant $U(s)$ voltage signal to uphold the output level at the terminal voltage.

4.3.1 Objective function and optimization

During the FOPID controller design, to tune the parameters ZLG optimization function was used. Although various standard optimization functions like IAE, ISE, ITAE, and ITSE are available, ZLG produced better results according to [76]. The equation for the ZLG optimization function [76] was given in (4.8).

$$ZLG = (1 - e^{-\beta}) * (M_p + E_{ss}) + e^{-\beta} * (T_s - T_r) \tag{4.8}$$

The term M_p represents peak overshoot, E_{ss} is the steady-state error, T_s and T_r represents settling time and rise time of the system respectively. β is an adjustment parameter and its value is taken as 1 [76]. The convergence curve for the BWO-FOPID algorithm during parameter identification was shown in figure 4.8. The range of parameters considered for the optimization process was mentioned in table 4.3.

Table 4.3 Parameter range for BWO-FOPID controller

Parameter	Lower value	Upper Value
K_p	0.1	3
K_i	0.1	1
K_d	0.1	0
λ	0.5	0.5
μ	0.5	1.5

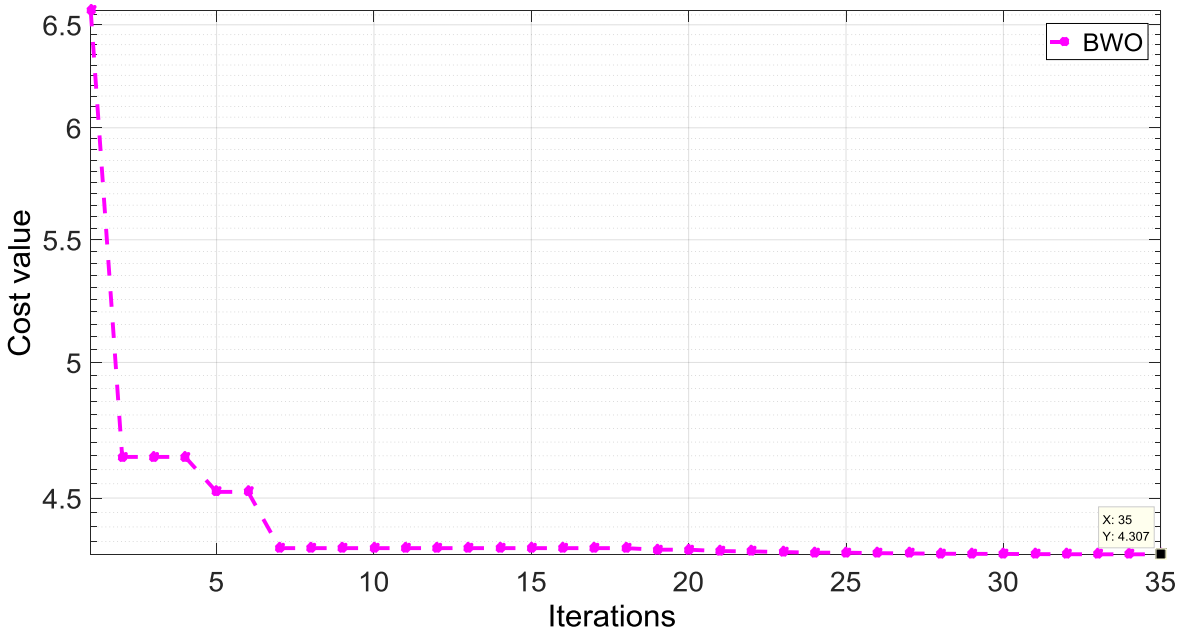

Figure 4.8 Convergence curve for BWO-FOPID controller

4.4 Results and Discussions

All the simulations were performed using MATLAB/Simulink (with FOMCON toolbox) version 8.1a on the computer with Intel i5 processor @ 3.00GHz and 8GB RAM. For the BWO optimization algorithm, the total population was chosen as 50 and 35 iterations were performed.

4.4.1 Step response

The tuned values of FOPID parameters were mentioned in table 4.4. For the simulation the FOPID controllers are designed by considering the frequency range [0Hz, 1000Hz],

identified from the frequency response of the system. Different FOPID controllers for AVR system are compared w.r.t step response as shown in figure 4.9. From the figure, it can be observed that the BWO-FOPID controller produced lowest overshoot 1.27%. To further investigate the controller performance T_s, T_r, and E_{ss} were calculated and compared with other FOPID controllers.

Table 4.4 Tuned controller parameters of FOPID controllers

Algorithm - Controller	K_p	K_i	K_d	λ	μ
BWO-FOPID	2.6597	0.7462	0.4263	1.0106	1.3442
C-YSGA-FOPID	1.7775	0.9463	0.3525	1.206	1.1273
PSO-FOPID	1.5338	0.6523	0.9722	1.209	0.9702
CS-FOPID	2.459	0.1759	0.3904	1.38	0.97
GA-FOPID	0.9632	0.3599	0.2816	1.8307	0.5491

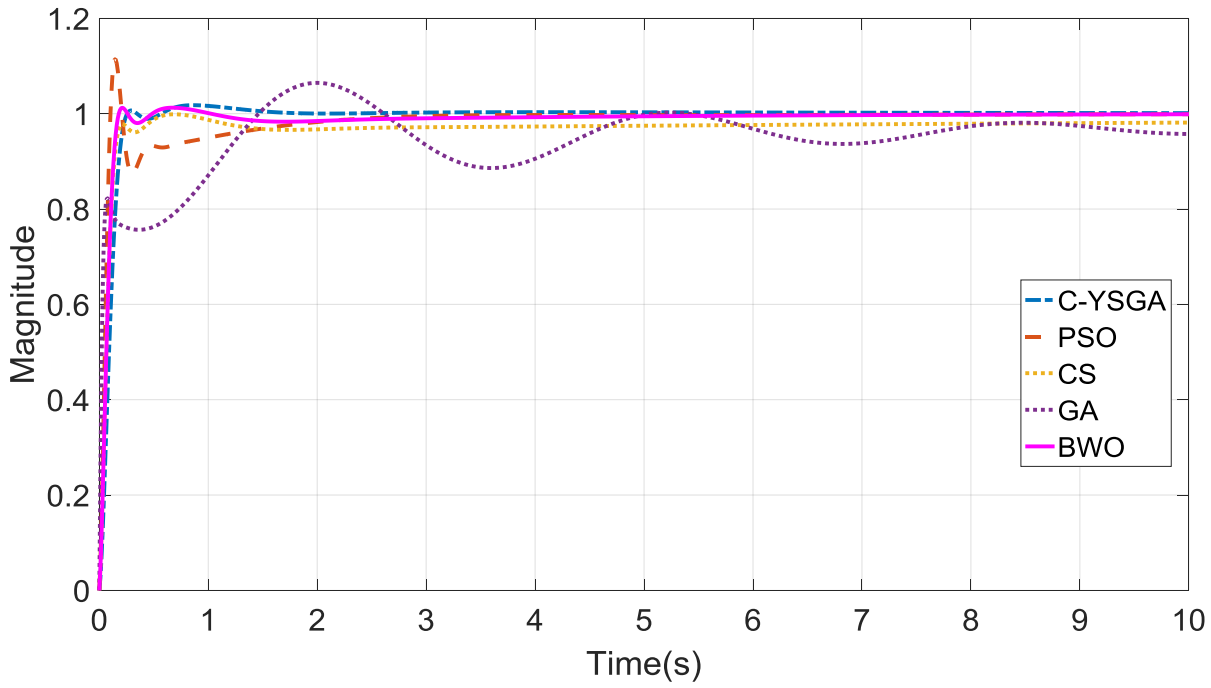

Figure 4.9 Comparison of step response of different FOPID controllers

The BWO algorithm produces the best parameter values because of the cannibalism stage, in which the weak solutions are automatically omitted and only strong solutions exist. It is observed that the BWO-FOPID controller has a better settling time of 0.1727s and overshoot 1.2774s and produced a very less steady-state error. The rise time of the controller is a little higher than the PSO-FOPID and GA-FOPID controllers. Moreover,

the PSO-FOPID controller produced the highest overshoot of 22.58% whereas the GA-FOPID controller produced a high rise time of 1.3008s and settling time of 1.6967s.

Table 4.5 Comparison of performance metrics of FOPID controllers

Controller	Rise time(s)	Settlign time(s)	Overshoot(%)	Steady state error
BWO-FOPID	0.1127	0.1727	1.2774	1.90E-04
C-YSGA-FOPID	0.1347	0.2	1.89	0.009
PSO-FOPID	0.0614	1.3313	22.58	0.0175
CS-FOPID	0.0963	0.9774	3.56	0.0321
GA-FOPID	1.3008	1.6967	6.99	0.0677

Since in the voltage regulator systems overshoot causes severe problems than rise time issues more importance should be given to optimizing overshoot. If the operating environment of the system is strictly constrained then a trade off can be made between rise time and overshoot. The comparison for performance parameters of different FOPID controllers were mentioned in table 4.5.

4.4.2 Robust Analysis

To test the reliability of the designed controller, the robust analysis was performed by changing the time constants of various subsystems in the range of -20% to 20%. The corresponding step responses were plotted as shown in figures 4.10,4.11,4.12, and 4.13 for variation in τ_a, τ_e, τ_g, and τ_s respectively.

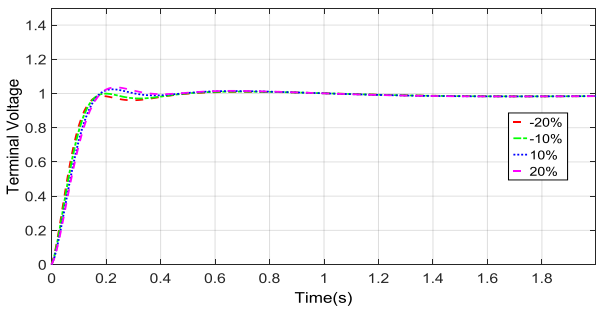

Figure 4.10 Robust analysis for τ_a

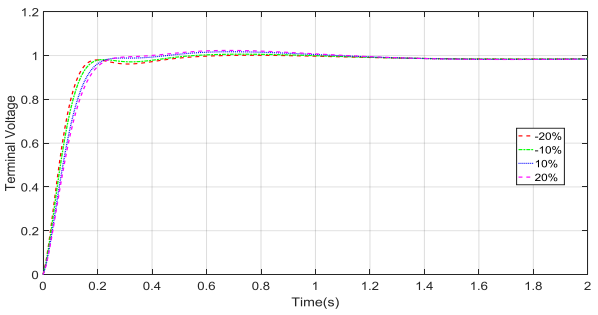

Figure 4.11 Robust analysis for τ_e

From the figures, it is found that the BWO-FOPID controller performs well even though there is a change of parameter values up to 40%.

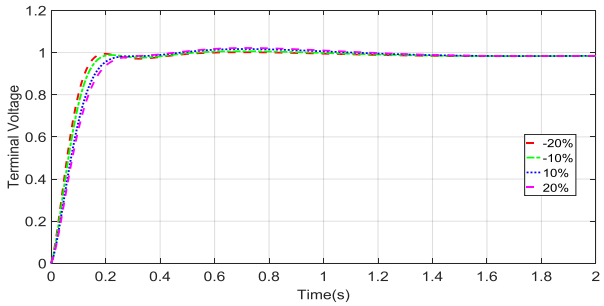

Figure 4.12 Robust analysis for τ_g

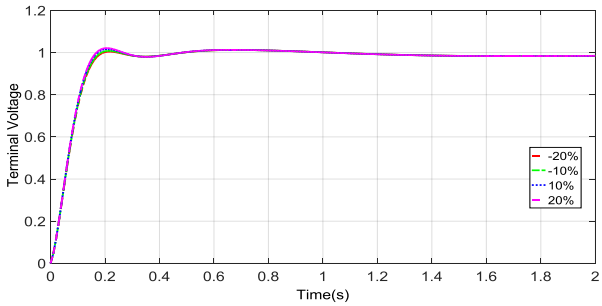

Figure 4.13 Robust analysis for τ_s

4.5 Proposed Chaotic Black Widow Optimization (ChBWO) algorithm

In general, most of the meta-heuristic optimization algorithms generate initial population using random number generators. The poor selection of random numbers results in the deterioration of performance of the optimization algorithm. On the other hand, chaotic maps can also be used as random number generators. Since they generate non-repeating random numbers, the search space of the algorithm increases, which is an advantage. Different types of chaotic maps are used in various optimization problems [2, 65, 68, 73]. These include Chebyshev map, tent map, logistic map, iterative map, piece-wise map, Gauss map, quadratic map, circle map, sinusoidal map, and cubic map.

To identify the best chaotic map for the BWO algorithm, a total of 10 chaotic maps are considered (appendix A.1). In the proposed algorithm, the initial population is generated from equation (4.9).

$$p_{i+1} = C_j(p_i) \tag{4.9}$$

where i=0,1,2, . . .,N and j=1,2, . . .,10. The variable C_j represents a chaotic mapping function.

4.5.1 Optimization of benchmark functions using ChBWO algorithm

To verify the efficiency of the proposed ChBWO algorithm, eight benchmark functions are considered, as listed in table 4.6. The statistical analysis is performed considering dimensionality, population size, and number of iterations as 10, 50, and 500, respectively. The table 4.7 compares the performance of different chaotic maps with respect to best,

mean, and median values. The detailed description of chaotic maps used for the study is mentioned in appendix A.1. It is found from the results that the logistic map produced best minimum values for 7 benchmark functions. Other than logistic map, sine map produced best minimum values for 4 benchmark functions. Later, Wilcoxon ranksum test with 0.05 significance level is performed for all the chaotic maps. For the test, the data related to 25 consecutive runs are considered. Table 4.8 shows the Wilcoxon ranksum results (p-values) which make comparison between the original BWO algorithm and BWO incorporated with chaotic maps. In table 4.8, the values that are less than 0.05 indicate that there is considerable difference in statistical results when compared with the actual BWO algorithm. The bold values indicate p-values that are greater than 0.05. It is identified from the comparison of p-values that the Chebyshev, Gauss, logistic, singer, sinusoidal and tent maps produce lower p-values than other chaotic maps. From the results presented in table 4.7 and 4.8, it is found that the logistic map produces better optimized values for the 7 benchmark functions and has p-values that are less than 0.05. Therefore, for the proposed ChBWO algorithm, the logistic map is used to initialize the spider population.

Table 4.9 shows the comparison of statistical results(best, mean, and median) between ChBWO algorithm and GA, PSO, BWO, ABC, and BBO algorithms. The best values are indicated in bold font. Table 4.9 indicates that ChBWO algorithm produced the best values for most of the objective functions.

Table 4.6 Cost functions for AVR system FOPID controller

Objective function	Range	min.
$f_1(x) = \sum_{i=1}^{n} x_i^2$	[-5.12,5.12]	0
$f_2(x) = \sum_{i=1}^{n-1}(100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2)$	[-30,30]	0
$f_3(x) = (x_1 - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$	[-10,10]	0
$f_4(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10cos(2\pi x_i))$	[-5.12,5.12]	0
$f_5(x) = \sum_{i=1}^{n}(x_i^2 - i)^2$	[-500,500]	0
$f_6(x) = -20exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) -$ $exp(\frac{1}{n}\sum_{i=1}^{n} cos(x_i)) + 20 + exp(1)$	[-35,35]	0
$f_7(x) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}})$	[-100,100]	0
$f_8(x) = 1 - cos(2\pi\sqrt{\sum_{i=1}^{n} x_i^2}) + 0.1\sqrt{\sum_{i=1}^{n} x_i^2}$	[-100,100]	0

Table 4.7 Comparison of statistics of ChBWO algorithm with different chaotic maps

		Chebyshev	Circle	Gauss	Logistic	Iterative	Piecewise	Sine	Singer	Sinusoidal	Tent
		C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
f1	Best	2.02E-06	5.34E-10	2.435717	2.24E-30	5.69E-12	2.80E-30	**1.13E-47**	4.54E-02	2.93E+00	1.81E-09
	Mean	1.75E-03	2.4E-04	8.63E+00	**1.84E-09**	4.99E-06	1.14E-08	1.64E-07	2.97E-01	5.07E+00	2.88E-05
	Median	4.21E-04	4.55E-05	8.76E+00	**1.30E-12**	3.22E-07	3.68E-07	4.90E-11	2.87E-01	5.02E+00	5.76E-07
f2	Best	6.94E-01	7.99E+00	5.44E+06	**1.94E-02**	2.61E+00	1.41E-01	3.34E+00	8.73E+02	1.21E+06	7.41E+00
	Mean	**4.81E+00**	9.05E+00	4.28E+07	6.72E+00	1.85E+01	1.03E+01	1.41E+01	1.26E+06	5.26E+06	4.16E+01
	Median	**3.52E+00**	8.82E+00	3792815	5.85E+00	1.05E+01	1.67E+01	1.00E+00	1.09E+05	5.55E+06	4.35E+01
f3	Best	8.29E-03	6.67E-01	1.69E+03	**2.866E-03**	1.10E+01	8.94E-02	4.55E-01	9.71E-01	4.43E+03	1.21E-01
	Mean	0.519237	0.667629	16179.89	**4.90E-01**	4.95E-01	6.83E-01	6.83E-01	8.78E+01	1.37E+03	4.98E-01
	Median	**4.12E-01**	6.67E-01	1.26E+05	4.48E-01	4.32E-01	6.02E-01	6.80E-01	6.88E+01	1.41E+01	5.93E-01
f4	Best	1.13E-08	6.10E-09	1.68E+00	**0**	**0**	**0**	**0**	4.06E-01	9.95E+00	7.25E-13
	Mean	2.73E-01	7.76E-02	1.29E+01	**1.31E-03**	6.95E-02	2.72E-03	9.49E-03	4.75E+01	1.26E+01	9.22E-02
	Median	4.77E-02	9.63E-03	1.14E+02	**5.51E-08**	9.55E-04	2.26E-02	7.77E-08	3.16E+00	1.23E+01	1.32E-02
f5	Best	2.91E-01	1.97E-01	8.85E+08	**1.08E-02**	6.30E-02	1.47E-01	3.37E-01	5.08E+06	3.1E+08	**1.79E-02**
	Mean	2.53E+02	1.97E+01	4.47E+09	3.53E+01	6.67E+01	**1.45E+01**	2.51E+01	2.39E+08	7.54E+08	2.38E+01
	Median	1.08E+01	1.32E+00	3.31E+09	1.47E+00	1.61E+00	2.01E+00	2.92E+00	1.66E+08	7.42E+08	**0.75774**
f6	Best	4.22E-02	5.74E-03	3.50E-01	**2.66E-15**	**2.66E-15**	2.79E-10	8.72E-12	5.33E-01	1.17E+01	4.33E-05
	Mean	5.10E-01	1.19E-01	1.87E+01	3.77E-04	3.86E-03	2.99E-04	**2.31E-01**	4.03E+01	1.36E-01	2.31E-02
	Median	3.97E-01	6.59E-02	1.15E+00	1.48E-05	8.07E-04	1.17E-03	**2.79E-06**	4.15E+01	1.39E+01	1.54E-02
f7	Best	4.73E-06	1.03E-07	5.38E-01	**0**	1.11E-16	**0**	**0**	1.38E-01	5.81E-01	6.55E-15
	Mean	4.29E-02	2.32E-02	1.25E+00	5.00E-04	6.24E-03	**3.73E-07**	2.31E-03	1.92E-01	8.67E-01	1.00E-02
	Median	3.90E-02	2.03E-02	1.11E+00	**8.05E-13**	4.40E-05	1.94E-03	5.20E-08	1.93E-01	8.67E-01	1.97E-03
f8	Best	**9.98E-02**	**9.98E-02**	4.09E+01	**9.98E-02**	**9.98E-02**	**9.98E-02**	**9.98E-02**	4.99E-01	4.09E+01	**9.98E-02**
	Mean	2.59E-01	2.03E-01	6.85E+01	1.07E+01	**9.98E-02**	**9.98E-02**	**9.98E-02**	1.59E+01	4.89E+01	1.35E-01
	Median	2.99E-01	1.99E-01	7.19E+01	**0.099873**	**9.98E-02**	**9.98E-02**	**9.98E-02**	1.69E+01	4.79E+01	**9.98E-02**

Table 4.8 Wilcoxon ranksum test results obtained by comparing 25 consecutive runs of BWO with various chaotic maps. The bold values indicates $\alpha \geq 0.05$ significance level

	f1	f2	f3	f4	f5	f6	f7	f8
C1	6.34E-06	6.07E-04	2.14E-02	9.14E-06	4.44E-o3	6.34E-06	6.34E-06	5.64E-05
C2	6.34E-06	**7.59E-02**	1.47E-05	5.19E-05	**2.84E-01**	6.34E-06	1.66E-05	3.59E-05
C3	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.26E-06
C4	**5.85E-02**	**2.66E-01**	**4.65E-01**	**3.02E-01**	**3.21E-01**	**3.81E-01**	**4.69E-01**	6.26E-06
C5	6.34E-06	2.44E-02	5.02E-04	2.13E-03	1.76E-02	2.97E-03	1.23E-04	6.26E-06
C6	1.79E-03	3.12E-02	**2.00E-01**	2.51E-03	6.05E-03	1.17E-02	6.59E-03	6.26E-06
C7	**6.85E-02**	1.09E-02	**4.12E-01**	**3.02E-01**	**1.07E-01**	**3.02E-01**	5.19E-03	6.26E-06
C8	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.23E-06
C9	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.34E-06	6.18E-06
C10	1.03E-05	9.14E-06	1.64E-03	4.14E-04	2.97E-03	9.14E-06	1.37E-04	6.18E-06

The comparison of convergence curves of different chaotic maps are shown in figure 4.14. From the curves, it is observed that the ChBWO algorithm converged in less than 100 iterations for all the benchmark functions. Moreover, except for function f1, the logistic map produced best cost values.

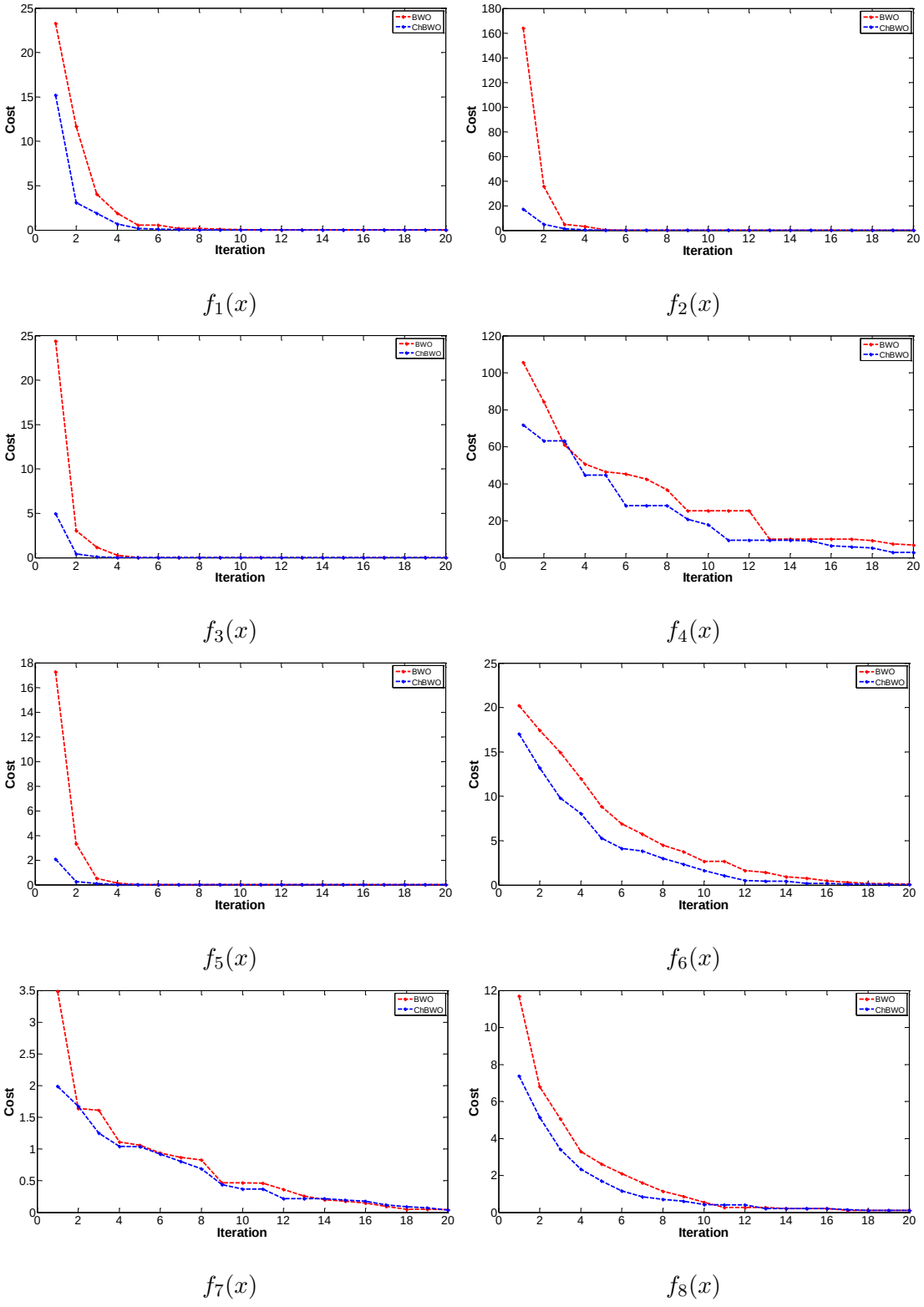

Figure 4.14 Convergence curves of ChBWO algorithm for different chaotic maps

Table 4.9 Comparison of statistical parameters of ChBWO algorithm for benchmark functions

Function	Parameter	ChBWO	GA	PSO	BWO	ABC	BBO
$f_1(x)$	Best	**2.24E-30**	1.30E-11	8.33E-05	2.35E-30	4.82E-05	5.70E-08
	Mean	1.84E-09	6.15E-04	8.60E-03	2.45E-07	6.54E-04	**1.69E-07**
	Median	**1.30E-12**	7.90E-06	3.72E-03	6.10E-12	4.38E-04	1.71E-07
$f_2(x)$	Best	**1.94E-01**	4.45E-01	9.12E-01	3.54E-01	4.52E+00	6.52E-01
	Mean	6.72E+00	1.02E+01	2.38E+02	7.90E+00	1.38E+01	**5.39E+00**
	Median	5.85E+00	7.40E+00	1.61E+01	7.22E+00	1.14E+01	**4.68E+00**
$f_3(x)$	Best	**2.87E-03**	4.40.E-01	4.97E-03	4.37E-01	2.39E-01	1.03E+00
	Mean	**4.90E-01**	7.23E-01	9.46E+00	3.26E-01	5.72E-01	2.56.E+00
	Median	**4.48E-01**	6.95E-01	9.46E+00	4.77E-01	5.55E-01	2.50E+00
$f_4(x)$	Best	**0**	4.81E-10	2.03E-01	**0**	1.30E+01	1.99E+00
	Mean	**1.32E-04**	5.73E-01	7.90E+00	2.89E-03	2.90E+01	5.87E+00
	Median	**5.51E-08**	7.90E-03	6.09E+00	3.13E-06	2.92E+01	5.47E+00
$f_5(x)$	Best	**1.09E-02**	1.18E+00	1.04E+01	1.99E-01	5.44E-01	2.44E-02
	Mean	3.53E+00	2.00E+02	1.38E+02	2.83E+00	5.85E+00	**4.85E-02**
	Median	1.45E+00	5.33E+00	7.04E+01	2.18E+00	4.30E+00	**4.78E-02**
$f_6(x)$	Best	**2.66E-15**	4.82E-05	8.44E-05	2.78E-13	1.17E-01	1.45E-03
	Mean	**3.8E-04**	4.97E-02	4.43E-03	3065E-03	3.23E-01	1.89E-01
	Median	**1.32E-06**	1.22E-02	2.68E-03	4.53E-05	2.72E-01	1.95E-03
$f_7(x)$	Best	**0**	7.33E-08	1.22E-01	**0**	5.19E-02	7.40E-03
	Mean	**5.00E-04**	4.29E-02	5.48E-01	6.99E-03	1.51E-01	7.12E-02
	Median	**8.05E-13**	3.43E-02	5.61E-01	1.95E-05	1.61E-01	6.02E-02
$f_8(x)$	Best	**9.99E-02**	**9.99E-02**	2.11E+00	**9.99E-0**2	1.41E+00	**9.99E-02**
	Mean	**1.07E-01**	**1.53E-01**	3.66E+00	1.03E-01	2.30E+00	2.53E-01
	Median	**9.99E-02**	**9.99E-02**	3.77E+00	**9.99E-02**	2.30E+00	2.00E-01

4.6 Design of ChBWO-FOPID controller for AVR system

To optimize the AVR system response the chaotic black widow optimization algorithm is used. The tuning procedure is similar to the procedure mentioned in the section 4.4.1. The corresponding block diagram is shown in the figure 4.15.

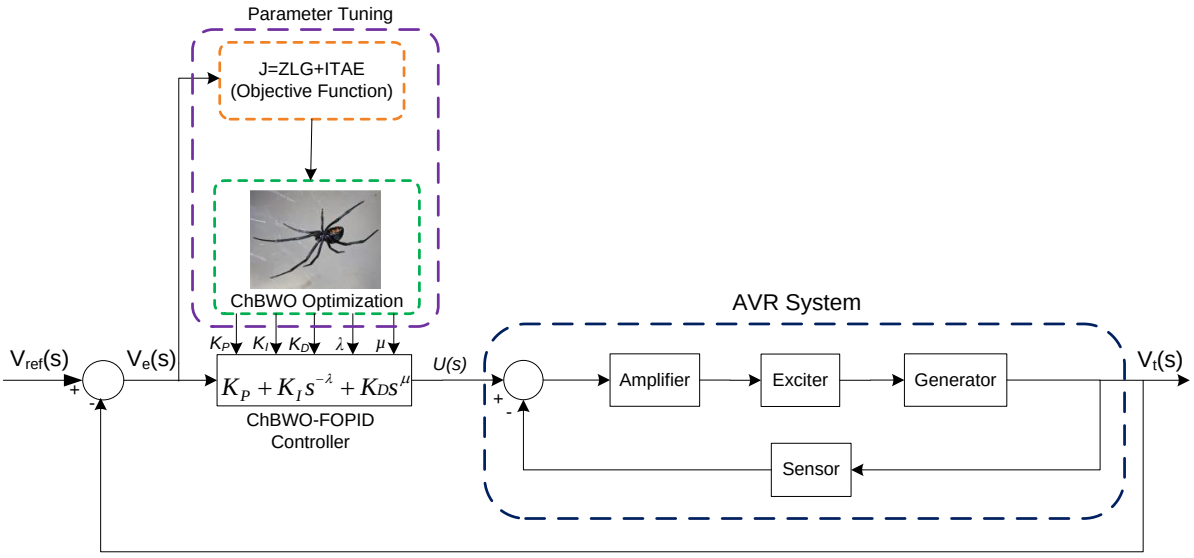

Figure 4.15 ChBWO tuned FOPID controller for AVR system

4.6.1 Parameter selection

The optimization process uses various parameters at different stages. All the parameters are categorized as AVR system, FOPID controller, and ChBWO algorithm parameters. The range of parameter values related to FOPID controller are given as $K_p = (0.1, 3), K_i = (0.1, 1), K_d = (0.1, 1.5), \lambda = (0.5, 1.5)$ and $\mu = (0.5, 1.5)$. The range for the parameters is identified from the observation of different FOPID controllers [63–65, 67], and [69].

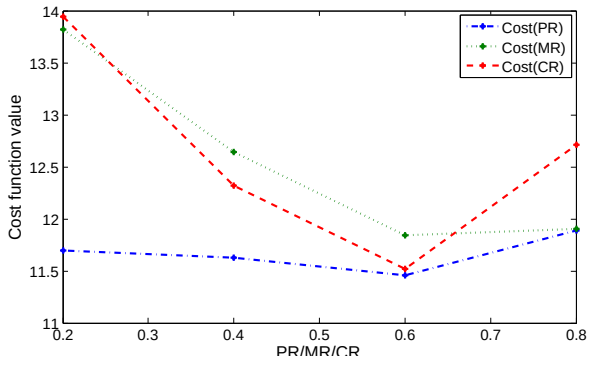

Figure 4.16 Super parameter effect on cost function

To identify the ChBWO algorithm super parameters (PR,CR, and MR), each parameter is varied in the range (0.2, 0.8) while keeping other parameters constant. The corresponding cost function curves related to the individual parameters are given in figure

4.16. The lowest cost value is produced when the procreation rate (PR), Cannibalism rate (CR), and Mutation rate (MR) are 0.6, 0.6, and 0.8 respectively.

4.6.2 Fitness function

To tune the parameters of FOPID controller, the state-of-the art methods used various types of objective functions as listed in the table 4.10. In addition to the general

Table 4.10 Cost functions for AVR system FOPID controller

Reference	Cost function
[68,69]	$ZLG = (1 - e^{-\beta}) \cdot (OS + e_{ss}) + e^{-\beta} \cdot (t_s - t_r)$
[70]	$IAE = \int \lvert e(t) \rvert dt$
[66]	$ITAE = \int t \lvert e(t) \rvert dt$
[67]	$OF_1 = \int te^2(t)dt, OF_2 = \int \Delta u^2(t)dt, OF_3 = \int te^2_{load}(t)dt$
[64]	$OF = w_1 \cdot OS + w_2 \cdot t_s + w_3 \cdot E_{ss} + w_4 \int \lvert e(t) \rvert dt + w_5 \int u^2(t)dt$
	$OF = (w_1 \cdot OS)^2 + w_2 t_s^2 + \frac{w_3}{(max\_dv)^2}$
[67]	$OF_1 = IAE, OF_2 = 1000 \lvert E_{ss} \rvert, OF_3 = t_s$
[65]	$OF_1 = \omega_{gc}, OF_2 = P_m$
[63]	$OF = w_1 \cdot OS + w_2 \cdot t_r + w_3 \cdot t_s + w_4 \cdot E_{ss} + \int (w_5 \cdot \lvert e(t) \rvert + w_6 \cdot V_f(t)^2)dt + \frac{w_7}{P_m} + \frac{w_8}{G_m}$

objective functions like integral of squared error (ISE), ITAE, and integral time squared error (ITSE), most of the objective functions are formed by either taking the weighted sum of different performance metrics of AVR system or by considering the total error of the system. Moreover, the weighted combination of performance metrics increases the dimensionality of the fitness function. Therefore, a new objective function is defined in the proposed method to improve performance metrics and minimize the total error of the system without increasing the dimensionality of the cost function.

In the proposed method, ITAE and ZLG objective functions are combined into a single function. After several trials of simulations with different objective functions, it is found that the combination of ITAE and ZLG gives best tuned parameters of the FOPID controller. The corresponding mathematical formula is given by equation (4.10).

$$J = (1 - e^{-\beta}) * (M_p + E_{ss}) + e^{-\beta} * (T_s - T_r) + \int t.|e|dt \tag{4.10}$$

The proposed fitness function has two advantages. The first one is getting optimized overshoot, rise time and settling time of the system using the ZLG function. The second one is reducing the overall output error of the system using the ITAE function.

4.6.3 Convergence curve

To identify the optimum parameters of the ChBWO-FOPID controller for AVR system, we take the fitness function mentioned in Section 4.7.2. The ChBWO algorithm is initialized with 50 spiders (population) and run for 50 iterations. A comparison of convergence curves of the proposed ChBWO and existing BWO algorithms are shown in figure 4.17. The convergence curve shows that the ChBWO algorithm starts converging after

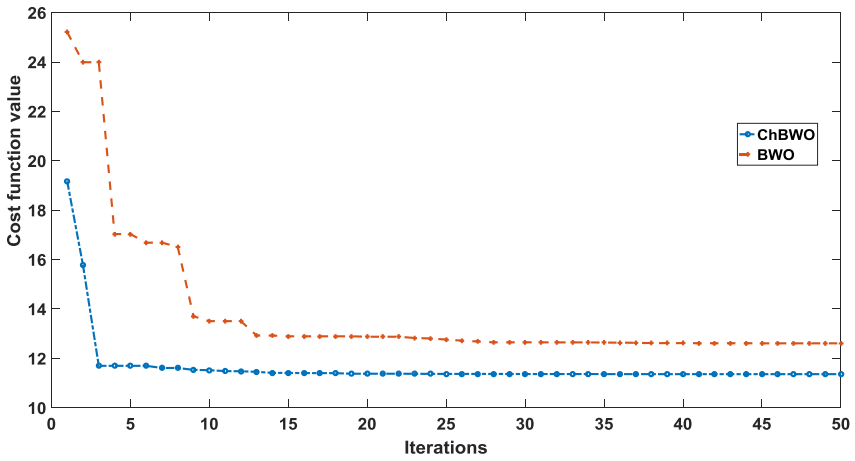

Figure 4.17 Comparison of convergence curves for BWO and ChBWO algorithms

iteration 5, whereas the BWO algorithm took 15 iterations for convergence. Moreover, the proposed algorithm has first iteration cost function value of 19, whereas it is 25 for the BWO algorithm. The reason for improvement is the use of the logistic map at the initialization stage of the proposed algorithm.

4.7 Results and Discussion

The proposed ChBWO-FOPID controller is tested for stability and reliability under different conditions. Various types of analyses (Step, load, robust, and Bode) are carried out on the controller. To improve the performance of AVR system, the proposed controller

is added in the control loop of the AVR system.

4.7.1 Step response

The step response of different FOPID controllers are compared using the optimally tuned parameters K_p, K_i, K_d, λ, and μ of the FOPID controllers as depicted in figure 4.18. Table 4.11 lists the tuned values of various FOPID/PID controllers.

Table 4.11 also shows the comparison of performance metrics related to the step response of different controllers. From the results in table 4.11, it is found that the the proposed controller produces good rise time and settling time values with acceptable overshoot (below 2%). When compared with other state-of-the-art methods, CHBWO-FOPID has improved the rise time 0.02s, settling time 0.03s, overshoot 0.7%, and phase margin 2^0. When compared to SA-MRFO-FOPID [73] and C-YSGA-FOPID [75]. The PSO-FOPID [64] and produced good rise time but suffered from high settling time and large overshoot values. The proposed controller reduced the overshoot 21.4% and settling time 1.16s, and phase margin 92.9^0. As compared to CS-FOPID controller [120] the proposed controller improved the settling time 0.808s and overshoot 1.72%. As compared to GA-FOPID [67] the proposed controller improved rise time 1.19s, settling time 1.53s, overshoot 5.816%, and phase margin 14^0. As compared to SCA-FOPID [72] and SA-FOPID [84] controllers the proposed controller improved Phase margin by 93.18^0 and 95.6^0 respectively. It is also informed that for various methods multiple runs of simulation showed a trade-off between overshoot, rise time, and settling time.

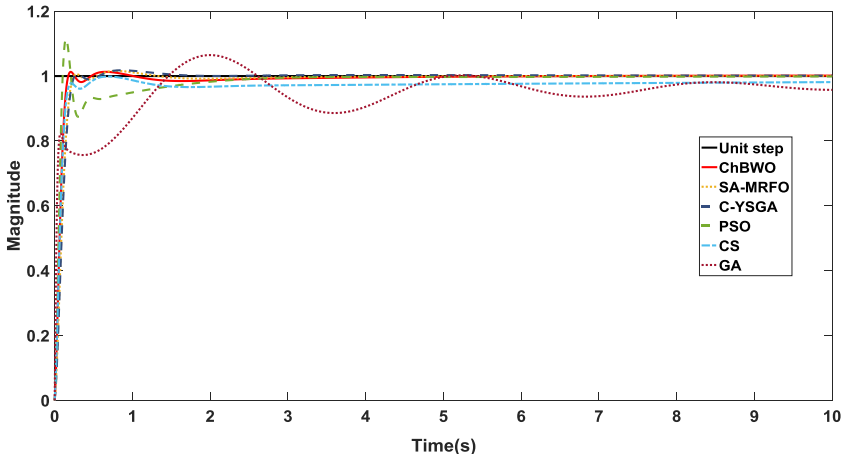

Figure 4.18 Step response of FOPID controllers for AVR system

Table 4.11 Tuned FOPID/PID controller parameters

Algorithm	K_p	K_i	K_d	λ	μ	Tr(s)	Ts(s)	Mp(%)	Ess	GM(Db)	PM(Degrees)
ChBWO - FOPID(Proposed)	2.8204	0.7387	0.428	1.1294	1.3558	0.1103	0.169	1.1838	0.0038	∞	164.1882
SA-MRFO - FOPID [75]	1.8931	0.8699	0.3595	1.0408	1.278	0.1309	0.1909	1.9765	-	∞	162.2232
C-YSGA - FOPID [73]	1.7775	0.9463	0.3525	1.206	1.1273	0.1347	0.2	1.89	0.009	∞	163.3101
PSO - FOPID [64]	1.5338	0.6523	0.9722	1.209	0.9702	0.0614	1.3313	22.58	0.0175	∞	71.2704
CS - FOPID [120]	2.549	0.1759	0.3904	1.38	0.97	0.0963	0.9774	3.56	0.0321	∞	166.5284
GA - FOPID [67]	0.9632	0.3599	0.2816	1.8307	0.5491	1.3008	1.6967	6.99	0.0677	∞	150.111
SCA - FOPID [72]	1.5447	0.5691	0.2389	1.3522	1.3195	0.1549	0.2183	0.01	1.49E-05	4.65	71.01
SA - FOPID [84]	0.7837	0.5027	0.2307	1.0103	1.0727	0.2653	0.4057	0.52	5.99E-10	16.37	68.59

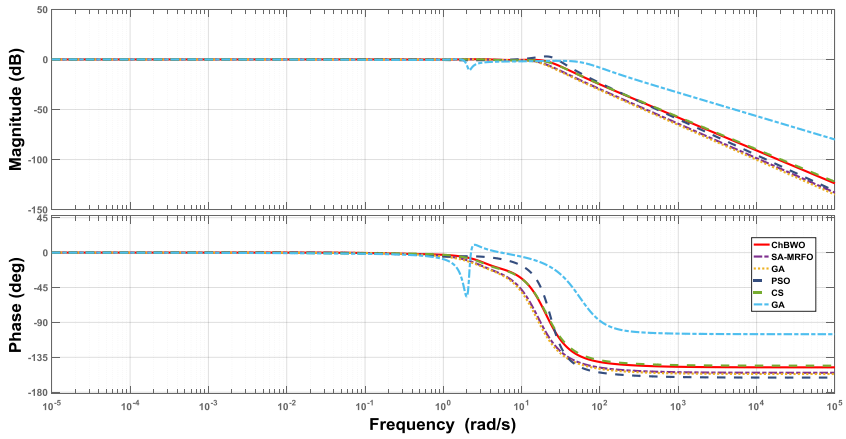

Figure 4.19 Comparison of Bode plots

The Bode plots of different controllers are compared as shown in figure 4.19. The analysis of plots indicates that all the controllers produce stable responses. The ChBWO-FOPID controller has infinite gain margin with phase margin of 164.19 degrees.

4.7.2 Load response analysis

During the simulation, the proposed controller is tested with two types of load conditions. Initially, different step changes are applied to the controller continuously. Then the response is compared with respect to other controllers. The results in figure 4.20 show that the ChBWO-FOPID controller is able to track different set points more efficiently than the other controllers. Later, the controller is fed with sudden impulses

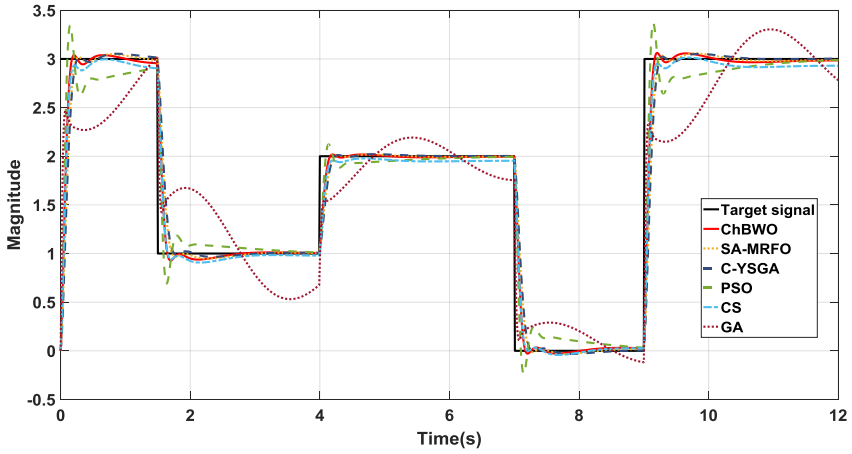

Figure 4.20 Load response of AVR system for different FOPID controllers

in the set point. The impulses act as disturbances and help to identify the disturbance

response of the controller. From figure 4.21, we can conclude that the proposed controller

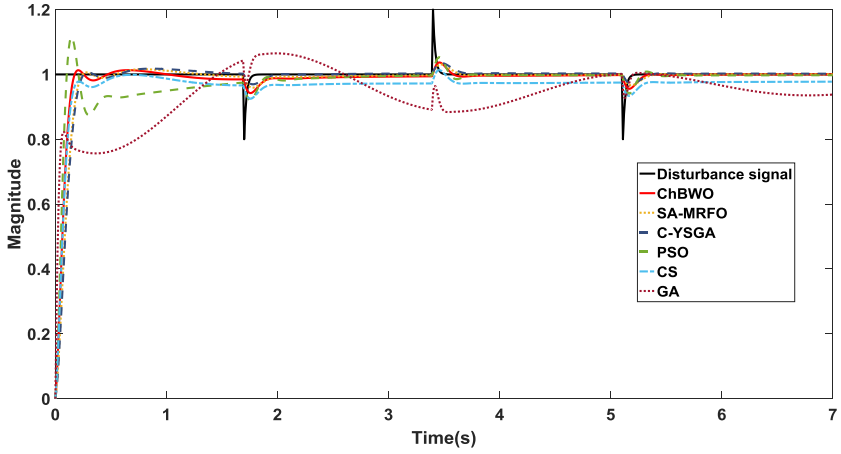

Figure 4.21 Disturbance response of AVR system for ChBWO FOPID controller

can withstand sudden impulses of load and responds optimally.

4.7.3 Robust analysis

The controller is tested for parameter variations of the AVR system ranging from -50% to 50% uncertainty. The parameters considered are time constants of amplifier(τ_a), exciter(τ_e), generator(τ_g) and sensor(τ_s) as listed in table 4.12. The corresponding results are shown in figures 4.22, 4.23, 4.24, and 4.25, respectively. From the graphs, it is observed that the proposed controller performance is satisfactory under parameter variations.

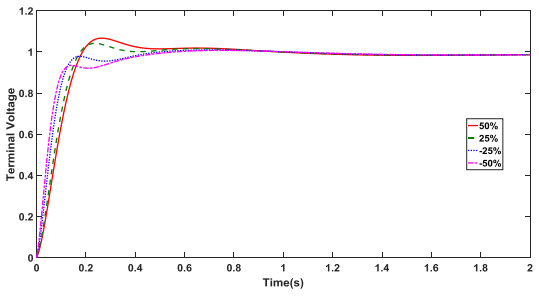

Figure 4.22 Robust analysis for τ_a

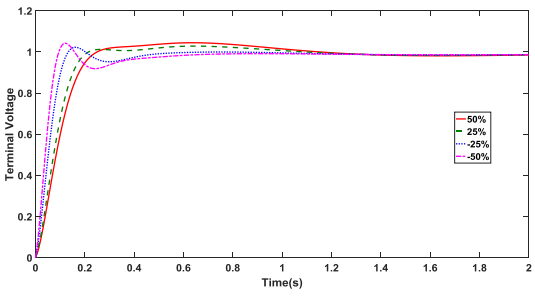

Figure 4.23 Robust analysis for τ_e

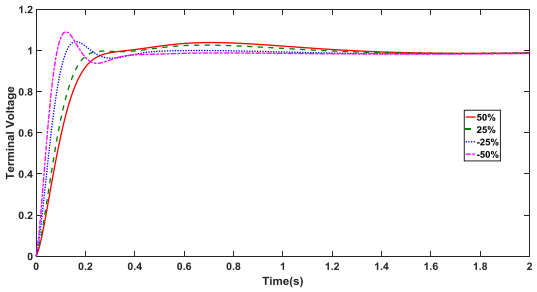

Figure 4.24 Robust analysis for τ_g

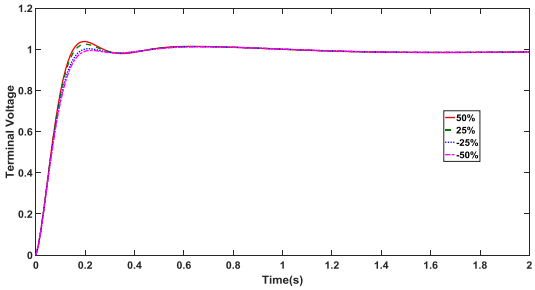

Figure 4.25 Robust analysis for τ_s

Table 4.12 Robust analysis performance of proposed controller

Parameter	Deviation(%)	Value	Rise time(s)	Settling time(s)	Overshoot(%)
τ_a	50	0.15	0.1424	0.4404	3.9027
	25	0.125	0.1347	0.2068	1.4395
	-25	0.075	0.1234	0.4211	0.7976
	-50	0.05	0.1432	0.455	0.5819
τ_e	50	0.6	0.1722	0.9683	4.2137
	25	0.5	0.1502	0.8313	2.5919
	-25	0.3	0.1043	0.4601	0.0222
	-50	0.2	0.08	0.5967	0.0209
τ_g	50	1.5	0.1879	1.0231	3.5755
	25	1.25	0.1571	0.8444	2.3547
	-25	0.75	0.1	0.4164	0.019
	-50	0.5	0.0732	0.5007	1.7496
τ_s	50	0.015	0.1003	0.8765	5.1407
	25	0.0125	0.1033	0.8813	3.8387
	-25	0.0075	0.111	0.8906	2.6313
	-50	0.005	0.1154	0.8952	2.6104

4.7.4 Summary

In this chapter, efforts are put to improve the FOPID controller performance for AVR system. An alternative method was proposed for optimal tuning of FOPID controller parameters based on chaotic maps and black widow optimization (BWO) algorithm [119]. To improve the convergence speed and search capability of BWO algorithm, effect of chaotic maps is studied and found that logistic map [121] improves the performance of the existing. The cost function plays a crucial role during optimization of any controller.

So, after working on different cost functions for the ChBWO algorithm, it is found that the combination of ZLG and ITAE functions produces optimum values for the FOPID controller. The advantage of ChBWO algorithm is early convergence and produced better

optimum values when compared with other state of the art algorithms like PSO, GA, ABC, and bio-geography-based optimization (BBO). Even though various cost functions have been proposed in the state-of-the-art approaches, most of them are weighted combinations of controller parameters and system error. As a result, the dimensionality of optimization problems increases. Since the proposed cost function does not contain these weights, it is simple to implement and effective.

4.8 Conclusion

A novel optimization algorithm has been proposed by hybridizing chaotic maps with the black widow optimization (BWO) algorithm. The effects of 10 chaotic maps have been studied and found that logistic map generates best results for most of the uni-modal and multi-modal benchmark functions. The proposed Chaotic BWO (ChBWO) algorithm has been used in the optimization of FOPID controller parameters for AVR system. To identify the finest controller parameters, a novel and efficient objective criteria with a combination of ITAE and ZLG functions is developed. Using simulation, it has been shown that the ChBWO-FOPID controller is able to produce better step response in terms of rise time and settling time than the existing state-of the-art techniques. To study the behavior of the controller, it is subjected to set point changes, sudden disturbances and parameter variations. In all the cases, the proposed controller has incremental improvement and produces stable and robust response.

Chapter 5

BR-NARXnets for Identification and Control of Automatic Voltage Regulator System

This chapter presents a novel design method for identification and control of automatic voltage regulator system. The method uses recurrent neural networks which have the capability to model time delayed systems. To avoid over fitting during the training of neural networks Bayesian regularization is used.

5.1 Introduction

Modeling and control of dynamic systems requires good understanding of parameters which govern the system operation. Moreover, for complex systems, it is difficult to identify the model. For example, in chapter 4, design of FOPID controller requires the mathematical model of the system. For complex systems these models may not be accurate and requires lot of time and effort to develop a good model of the system. Therefore in this chapter a data driven approach is proposed for the system identification as well as controller design. On the other hand neural networks have universal approximation capabilities which require only excitation and response data. Therefore, to model complex systems, instead of mathematical model the neural networks can be used to find the system behavior. The advantage is it reduces the modeling time and requires only input and output data of the system. Even though different types of neural networks are available, NARXnets (Non-linear Auto regressive with eXogenous input networks) are better suited

for the problems where inherent delays exist in the system.

Even though various types of PID and FOPID controllers are developed for AVR system, there are very few studies in which neural networks are used. In the proposed method, an attempt has been put for the design of NARXnet based AVR system model and corresponding controller development.

5.2 Automatic Voltage Regulator (AVR) System

5.2.1 Overview of AVR system

The industrial power generation units are often employed with synchronous generators. In order to regulate the output voltage of synchronous generator and to stabilize the overall system synchronous generators are often integrated with automatic voltage regulator (AVR) systems. The AVR system internally composed of four modules namely, amplifier, exciter, generator, and sensor. The standard model of the AVR system is mentioned in the literature [66,68,73,75]. In the model, the individual modules are represented as first order systems and shown in the figure 5.1.

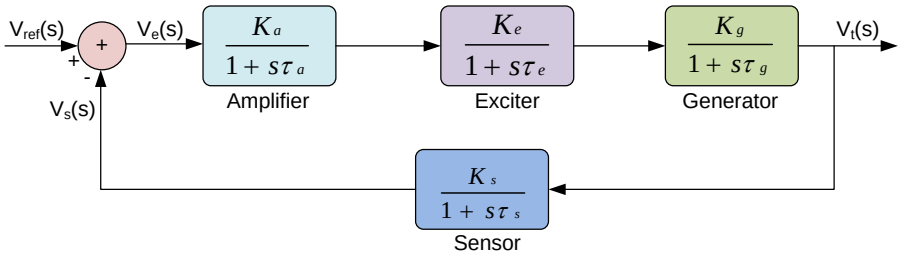

Figure 5.1 AVR system block diagram

Correspondingly, various AVR system parameters used in the modeling are mentioned in table 5.1. The parameter values are identified from the literature [66, 68, 73, 75].

In the table, K represents gain (unit-less) and T represents time constant (in seconds) of different modules. The suffixes a, e, g, and s indicates the modules amplifier, exciter, generator, and sensor respectively.

Using the parameters mentioned in the section 5.1, the AVR system model is de-

Table 5.1 Parameter values of AVR system

Parameter	Range	Value
K_a	[10,400]	10
τ_a	[0.02,1]	0.1
K_e	[1,10]	1
τ_e	[0.4,1]	0.4
K_g	[0.7,1]	1
τ_g	[1,2]	1
K_s	[1,2]	1
τ_s	[0.001,0.06]	0.01

veloped and simulated. The corresponding unit step response is shown in the figure 5.2. From the response, it is identified that the system has 60.52% overshoot, 0.26s rise time, and 6.22s settling time. Moreover, the system produced a steady state error of 0.091. The system response can be optimized by including a suitable controller in the closed loop form.

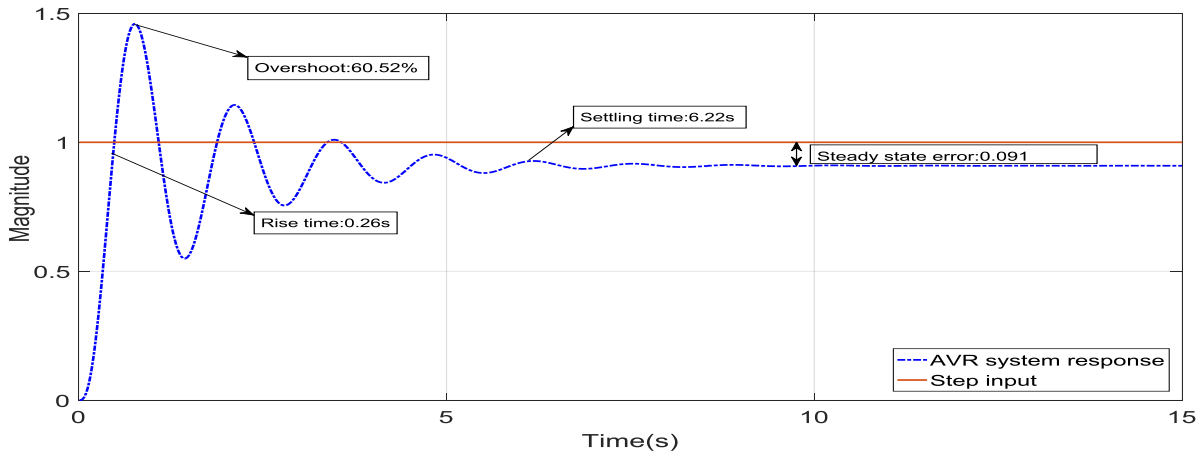

Figure 5.2 Open loop unit step response of AVR system

5.3 Bayesian regularization back-propagation algorithm

The NARXnets can be trained to identify the input-output mapping of dynamic systems. The internal weights and biases are updated based on the error between actual and target values. In the section, working of Bayesian regularization back-propagation training algorithm is discussed in detail.

The traditional back-propagation uses gradient descent algorithm to update the neural network weights. But the draw back is the network may over-fit the training data. This produces very good performance during training and poor performance when new data set or testing data is applied. To overcome the problem, regularization mechanisms should be incorporated into the training algorithm. In the proposed method, Bayesian regularization (BR) back-propagation is used to train the NARXnets. BR avoids the over fitting by including regularization parameters into the cost function. It also has other advantages like no need for lengthy cross-validation and omits the non-performing parameters in the network during training phase. Mathematically it converts non-linear regression problem to statistical L2 (Ridge) type regression. A brief overview of Bayesian regularization algorithm is discussed as follows.

Consider for an unknown system with the input samples $x_i \epsilon \mathbb{R}$ and the corresponding response $y_i \epsilon \mathbb{R}$. Let the set C represents input and response pairs $(x_i, y_i) \epsilon \mathbb{R}$ indicated in the equation (5.1)

$$C = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), ..., (x_p, y_p)\} \tag{5.1}$$

where p indicates number of samples in the data set. Conveniently, the above equation can be represented as $C = \{X, Y\}$, where X and Y indicates input and output values of the system. To identify the relation between x_i and y_i a neural network architecture M is developed. Let $W \epsilon \mathbb{R}$ represents the connection weights, then the mapping between inputs and responses is given by equation (5.2)

$$\hat{Y} = M(X, W) \tag{5.2}$$

The difference between actual response (Y) and estimated response ($\hat{Y}$) is denoted as error energy E_C and represented in equation (5.3)

$$E_C = \frac{1}{2}\sum_{i=0}^{p}(Y - \hat{Y})^2 \tag{5.3}$$

An additional regularization term called energy of weights indicated by E_w is added to the actual error as shown in equation (5.4)

$$E_w = \sum_{i=0}^{k}(\bar{w_i})^2 \tag{5.4}$$

where $\bar{w}_i$ indicates vector of all the weights in the neural network. Therefore, the final objective function is given by equation (5.5)

$$E = \mu * E_w + \eta * E_C \tag{5.5}$$

where the variables η and μ are objective function parameters. If $\mu << \eta$ then it leads to larger weights and smaller errors, can cause over fitting problem. If $\mu >> \eta$ then it leads to smaller weights at the cost of overall network error. Therefore, the optimum values of μ, η, and w can be found using Bayesian probability theory [122]. In Bayesian terminology, the network weights W are considered as random numbers having probability density function (5.6).

$$P(w|C, \mu, \eta, M) = \frac{P(C|w, \eta, M)P(w|\mu, M)}{P(C|\mu, \eta, M)} \tag{5.6}$$

In equation (5.6), $P(w|\mu, M)$ indicates prior density of network weights, $P(C|w, \eta, M)$ is likelihood function of data for given weights and $P(C|\mu, \eta, M)$ represents normalizing factor which limits the total probability to 1.

Assuming the noise in data and prior distribution of weights are Gaussian then the corresponding probability densities are written as shown in equation (5.7)

$$P(C|w, \eta, M) = \frac{1}{Z_C(\eta)} exp(-\eta E_C) \text{ and } P(w|\mu, M) = \frac{1}{Z_w(\eta)} exp(-\mu E_w) \tag{5.7}$$

where $Z_C(\eta) = (\pi/\eta)^{k/2}$ and $Z_w(\mu) = (\pi/\mu)^{N/2}$. Where k indicates the number of neural network weights and N indicates total number of parameters. Substituting (5.7), Z_C, and Z_w into equation (5.5)

$$P(w|C, \mu, \eta, M) = \frac{\frac{1}{Z_w(\mu)} \frac{1}{Z_C(\eta)} exp(-(\eta E_C + \mu E_w))}{P(C|\mu, \eta, M)} = \frac{P}{exp(-F(w))} Z_F(\mu, \eta) \tag{5.8}$$

where

$$Z_F(\mu, \eta) = Z_C(\eta) Z_w(\mu) \tag{5.9}$$

Applying Bayes rule [122] to optimize the parameters μ and η

$$P(\mu, \eta|C, M) = \frac{P(C|\mu, \eta, M)P(\mu, \eta, M)}{P(C|M)} \tag{5.10}$$

Assuming uniform prior density $P(\mu, \eta, M)$ for parameters μ and η then minimizing posterior density is achieved by maximizing $P(C|\mu, \eta, M)$. Since all the probabilities have Gaussian form, the corresponding posterior density is mentioned in equation (5.8). Solving equation (5.5) for normalization factor,

$$\begin{aligned} P(C|\mu,\eta,M) &= \frac{P(C|w,\eta,M)P(w|\mu,M)}{P(w|C,\mu,\eta,M)} \\ &= \frac{[\frac{1}{Z_C(\eta)}exp(-\eta E_C)][\frac{1}{Z_w(\eta)}exp(-\mu E_w)]}{Z_F(\mu,\eta)exp(-F(w))} \\ &= \frac{Z_F(\mu,\eta)}{Z_C(\eta)Z_W(\mu)}\frac{exp(-\eta E_C - \mu E_W)}{exp(-F(w))} \\ &= \frac{Z_F(\mu,\eta)}{Z_C(\eta)Z_W(\mu)} \end{aligned}$$

Expanding $F(w)$ around minimum point w^* using Taylor series, the optimum parameters are estimated as

$$\mu^* = \frac{\gamma}{2E_w(w^*)} \text{ and } \eta^* = \frac{k-\gamma}{2E_C(w^*)} \tag{5.11}$$

where

$$\gamma = k - 2\mu^* tr(H^*)^{-1} \tag{5.12}$$

The variable γ represents effective number of parameters in the network. H^* is the Hessian matrix of objective function evaluated at w^* and is calculated using Gauss - Newton approximation [123] as shown in equation (5.13).

$$H^* \approx J^T.J \tag{5.13}$$

where J is jacobian matrix. Then, the approximation factor $Z_F(\mu, \eta)$ is given by [124] equation (5.14)

$$Z_F(\mu,\eta) \approx (2\pi)^{N/2}(det(H^*))^{1/2}exp(-F(w^*)) \tag{5.14}$$

Computing H matrix at minimum point w^* is solved according to LM algorithm [113]. The weights and biases are adjusted using [122, 123] equation (5.15).

$$w^{k+1} = w^k - [J^T J + \lambda I]^{-1} J^T e \tag{5.15}$$

where λ denotes the LM damping factor and $J^T e$ is the error gradient.

5.4 Proposed BR-NARXnet System architecture

The proposed system architecture consists of two neural networks known as plant network and controller network. Initially, the behavior of the plant is identified by training the plant neural network on the AVR system data. Later, the controller network trained using the plant network and reference data. Both the networks uses Bayesian regularization algorithm to update the weights during training process.

The plant network is designed with 15 hidden layer neurons and one output neuron. The feedback in actual AVR system (figure 5.3) indicates that the current response of AVR system depends on past outputs. Therefore, the plant network also consists of two inputs one is the actual reference signal and the other one is the feed back signal. The network has two delay units for reference input and three units for the feedback signal. In figure 5.3, $u(k)$ and $\hat{y}$ are inputs for the plant and the outputs of delay blocks are $u(k), u(k-1)$, $u(k-2)$, $\hat{y}(k)$, $\hat{y}(k-1)$, and $\hat{y}(k-2)$ respectively.

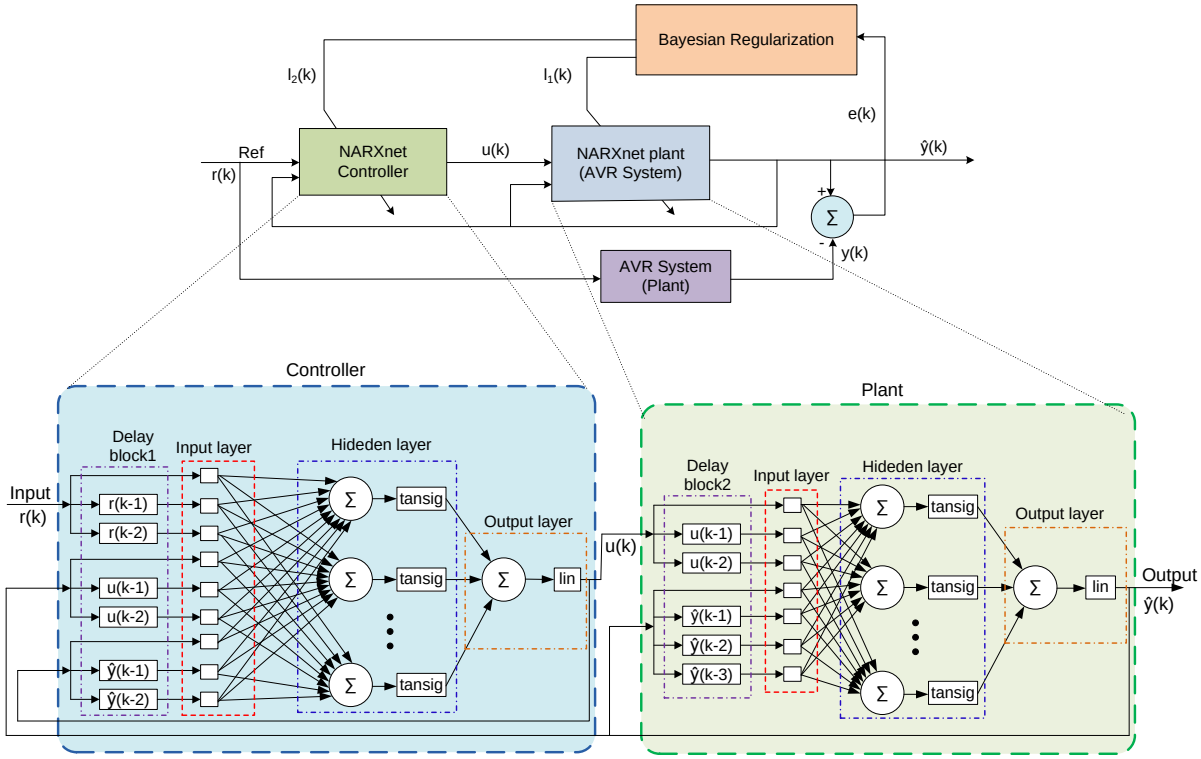

Figure 5.3 Architecture of proposed system

Similarly, the controller network is also created with 15 hidden neurons and one output neuron. For better training and to learn past dependencies the controller has

feedback signals from its own output as well as system output. Therefore, the controller has totally three inputs known as reference signal ($r(k)$), controller output feedback ($u(k)$), and actual system response ($\hat{y}(k)$). All these inputs are delayed by two units as discussed in plant network to optimize the overall system response.

The commonly used activation functions are linear, sigmoid, and tan sigmoid functions whose definitions are given in equations (5.16), (5.17), and (5.18). The plant and controller network hidden layer neurons are activated using tansigmoid function and output neurons are activated using linear function.

$$Sigmoid : f(x) = \frac{1}{1+e^{-x}} \ \epsilon \ [0,1] \tag{5.16}$$

$$Tansigmoid : f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \ \epsilon \ [-1,1] \tag{5.17}$$

$$Linear : f(x) = x, \text{ and } x \ \epsilon \ \mathbb{R} \tag{5.18}$$

Even though the activation functions mentioned are equally good for general applications, we have chosen tansigmoid activation function for the NARXnets. Since tansigmoid function generates the output in the range [-1,1] which better suits for controlling applications.

5.5 BR-NARXnet Training and Simulation

The total training part is divided into two phases. In the first phase, the dynamics of plant neural network are identified using the data generated using the Simulink model of AVR system shown in figure 5.4. In the second phase, the identified plant dynamics along with controller training data are used to train the controller network.

5.5.1 Training procedure for BR-NARXnet

The procedure to train the BR-NARXnet controller for the AVR system involves the following steps. It is mentioned that the proposed method can also be used as a generalized procedure to design BR-NARXnet controllers for various dynamic systems.

Plant training:

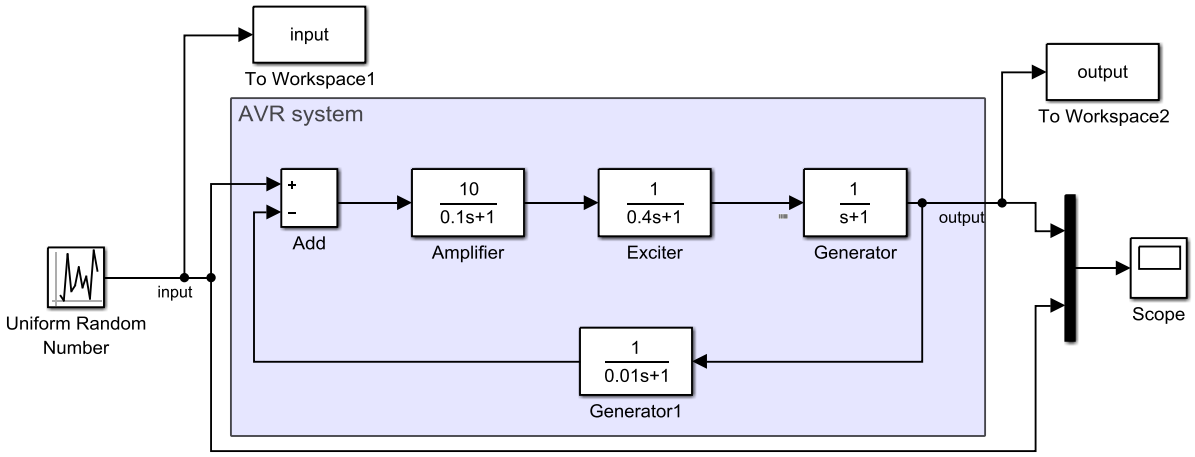

Figure 5.4 Simulink model for training data generation

1. Collect the training data i.e, excitation (input) and response (output), for the system (AVR system) with N samples.

2. Initialize the plant network with Hp hidden neurons, I_p input neurons, and O_p output neurons. The number of input neurons is equal to number of features and output neurons is equal to plant outputs.

3. Set the number of epochs to 1500, learning method as Bayesian regularization and chose the performance criteria for training as MSE.

4. Train the neural network using the procedure mentioned in Section.3 as described in the following

   (a) Calulate the regularization parameters using equations (5.11) and (5.12)

   (b) Apply the back propagation algorithm for the calculated sensitivities using equation (5.5)

   (c) With the help of the jacobian matrix J update the weights using equation (5.15)

5. Calculate the MSE and statistics during training and plot the response.

Controller training:

1. Collect the training data for the controller with M samples.

2. Initialize the controller network with Hc hidden neurons, I_c input neurons and O_c output neurons and connect the controller network to plant network as shown in figure 5.3.

3. Disable the learning for plant network, and train the controller using the procedure mentioned in steps 3 and 4.

4. Save the network parameters, weights, and biases.

5. Using the saved model predict the output from the testing data set and evaluate the metrics.

5.5.2 Training plant neural network

The plant network training data is generated from the Simulink model of AVR system as shown in figure 5.4. A total of 50000 excitation and response samples are generated. The figures 5.5 and 5.6 represents the input and response data of the actual AVR system respectively. During plant network training, the controller network weights are deactivated and the reference signal $r(t)$ is directly fed to the input of the plant network along with feedback.

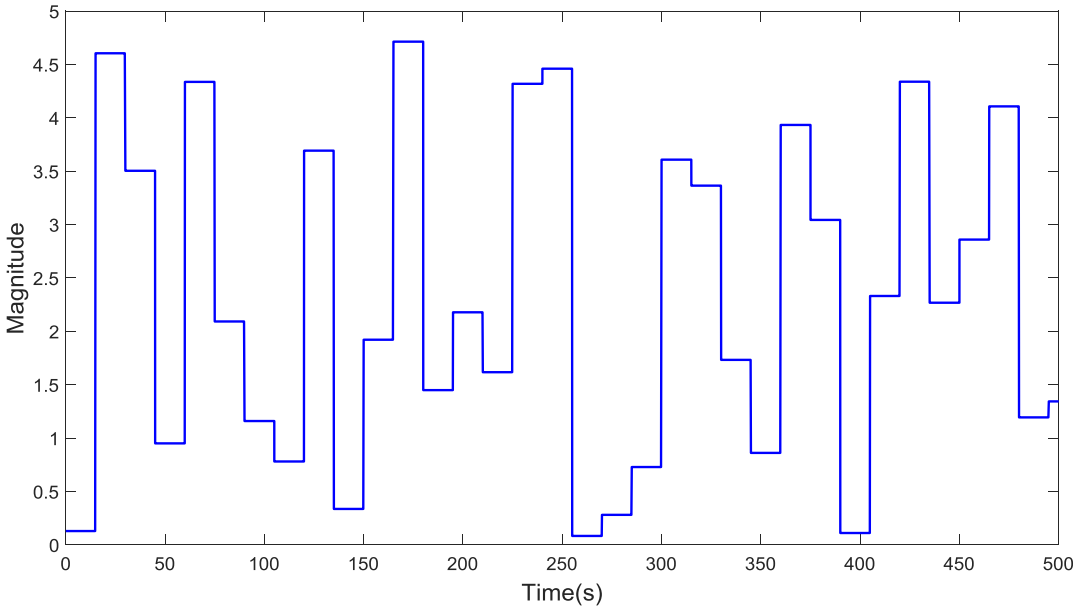

Figure 5.5 Training data (input) for plant neural network

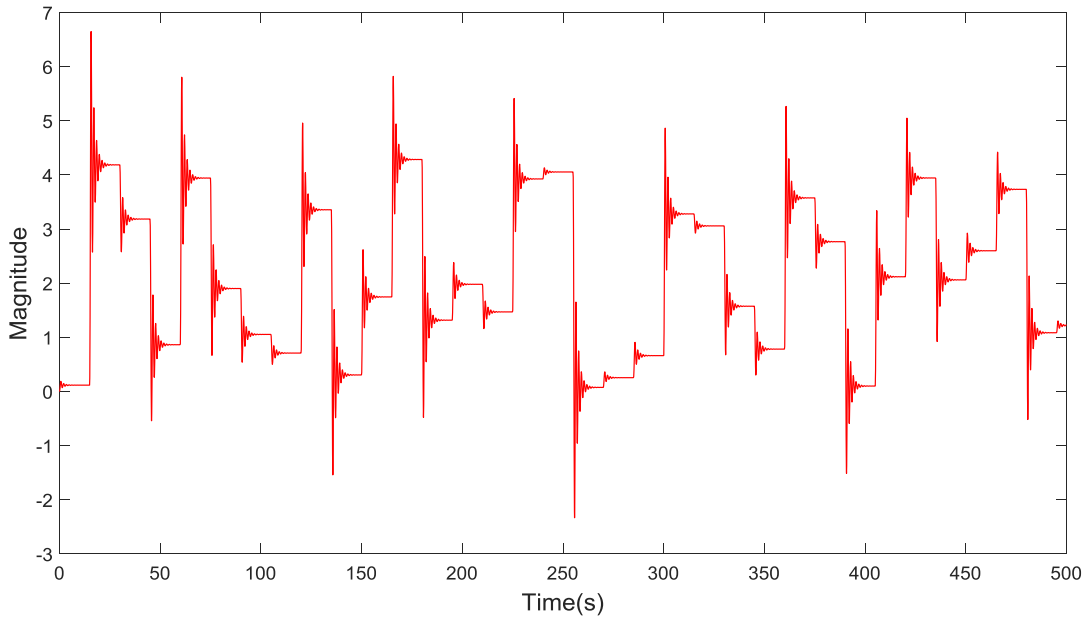

Figure 5.6 Training data (output) for plant neural network

Bayesian regularization backpropagation algorithm is used to train the neural network with mean square error (MSE) evaluation criteria. The detailed analysis of training performances are given in section 5.6. The performance curve of the plant network during the training is shown in figure 5.7. The network produced minimum cost value 1.688e-10 for 1500 epochs.

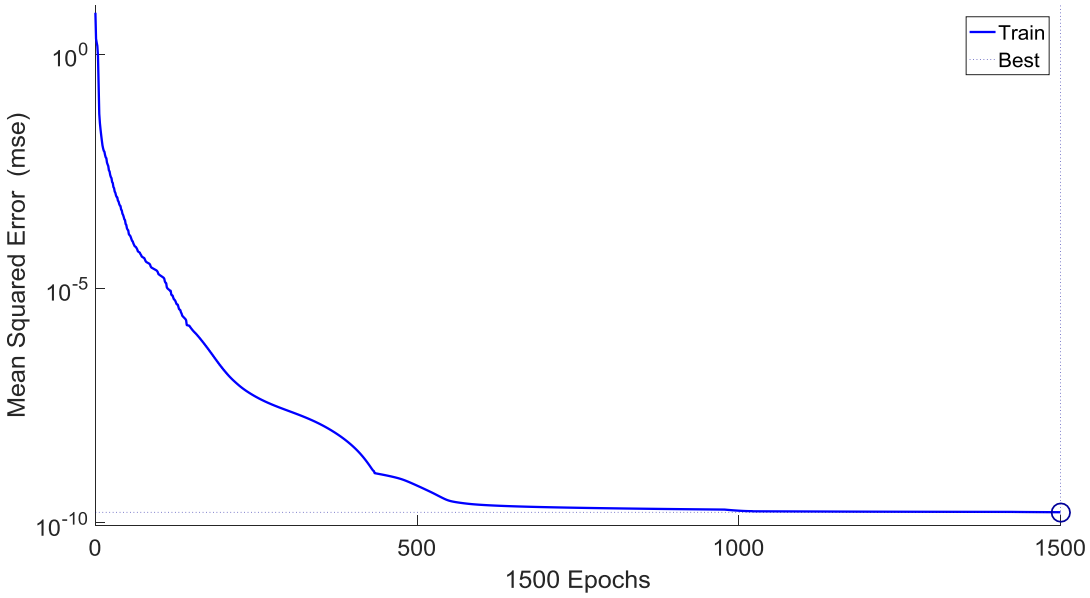

Figure 5.7 Training performance of the plant network

The error distribution is given in the figure 5.8. The distribution plot is drawn by

taking error values on one axis and instances of error are taken on another axis. From the figure, it is observed that most of the error samples are located near the zero line (indicated as red line).

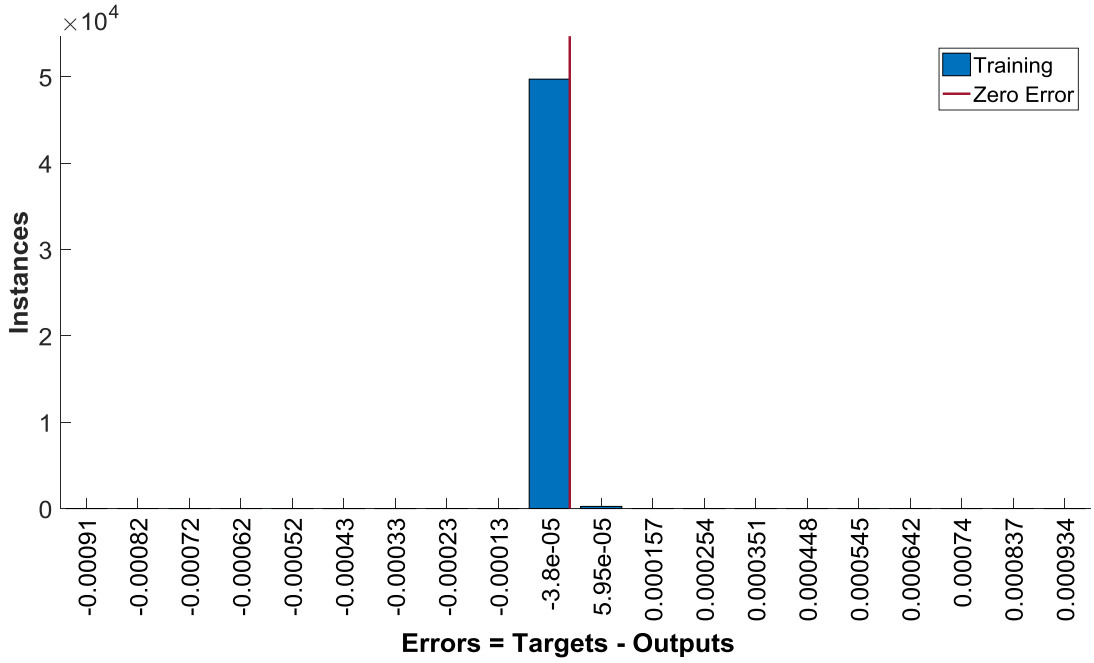

Figure 5.8 Training error histogram for plant

5.5.3 Training controller neural network

The controller network is also trained similar to the plant neural network. During the controller training, the trained plant network is added to the controller network whose output can be used to compute the error. The weights of the controller network are updated using the error values. Since the plant network is already trained, its weights are not updated during controller network training. The training data for the controller network is shown in figure 5.9 To train the controller Bayesian regularization algorithm with MSE criteria is used. The performance curve of the controller network during the training is shown in figure 5.10. The controller network took 1500 epochs to produce the minimum MSE value 0.003021. Various parameters used during the plant and controller network training are summarized as shown in table 5.2.

The error distribution plot after controller training was completed is shown in the figure 5.11. The plot is generated similar to the plant error distribution. From the figure,

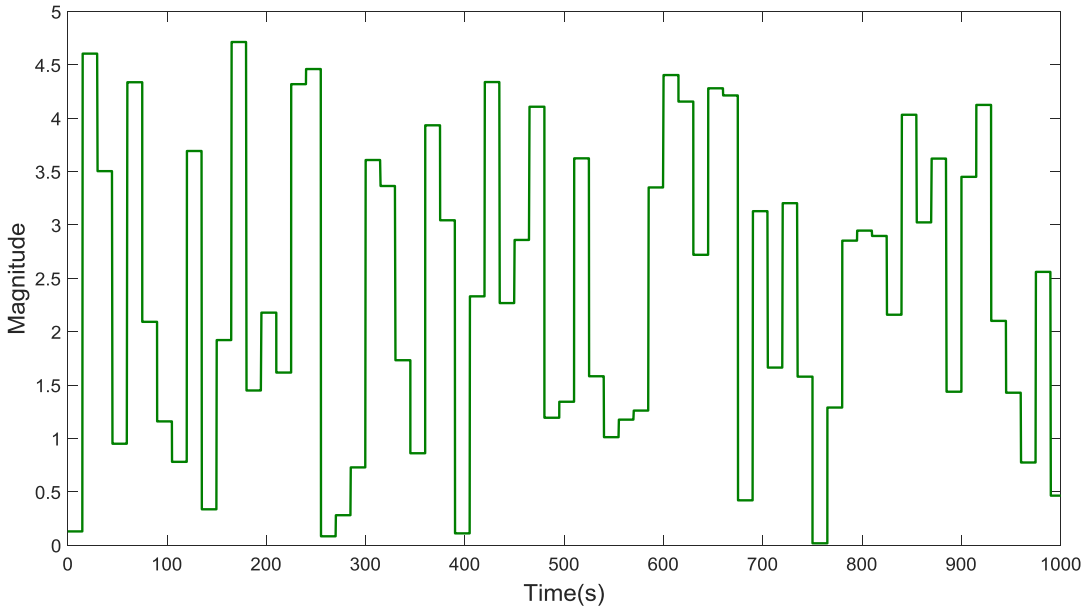

Figure 5.9 Training data for controller neural network

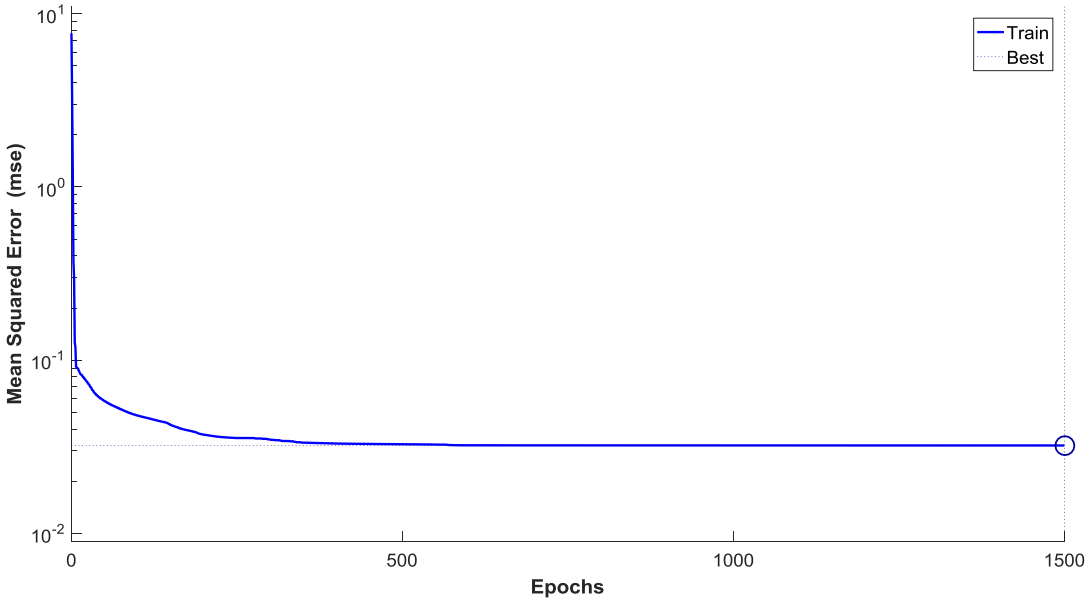

Figure 5.10 Training performance of the controller network

it is found that most of the errors are located near the zero value which is indicated as a solid red line in the figure 5.11.

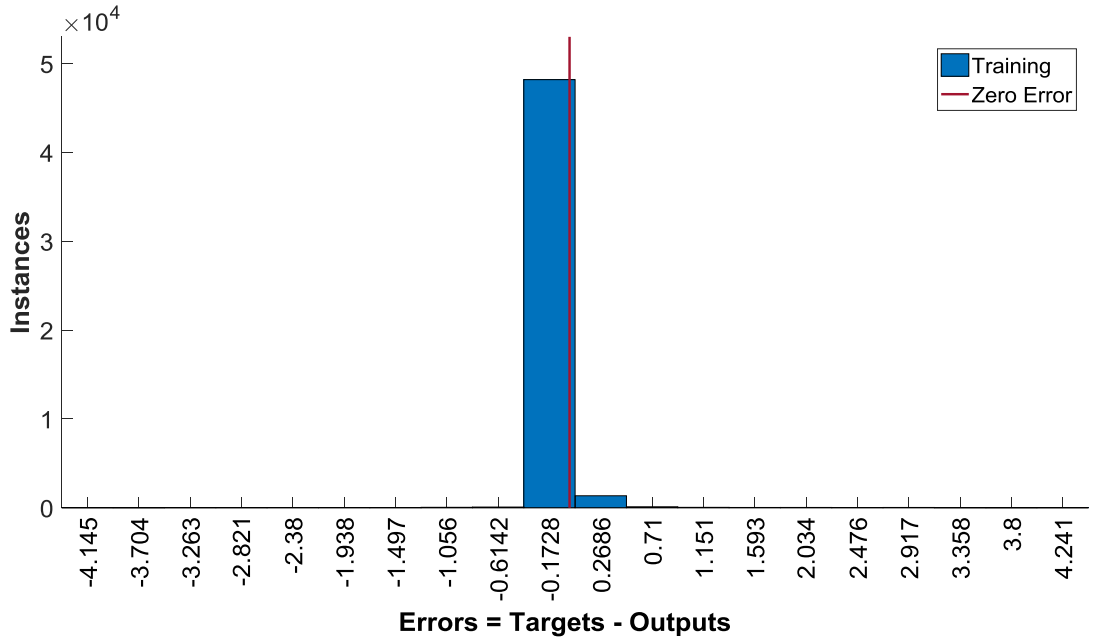

Figure 5.11 Training error histogram for controller

5.6 Results and discussions

All the training and simulations are performed using MATLAB 2016b software on a computer with intel core i5 processor with 8GB RAM. The simulation results are divided into two parts containing plant and controller performances. Various analysis are performed on the plant and controller networks during the training process.

5.6.1 Analysis of plant training

To study the effect of hidden number of neurons on plant identification the plant network is trained with 5, 10, 15 and 20 hidden neurons. Moreover, the training is performed by considering Bayesian regularization [122] and Livenberg Marquardt (LM) algorithm [116]. To compare the performance of the neural network various measures like MSE (Mean Square Error), SSW (Sum of Squared weights), gradient, and RMSE (Root Mean Square Error) are considered. The formulas used for evaluation of MSE, SSW, and RMSE are given by equations (5.19), (5.20), and (5.21).

$$MSE = \frac{1}{m} E_D \tag{5.19}$$

Table 5.2 Parameters for plant and controller neural networks

Parameter	Plant network	Controller network
Inputs	2	3
Unit delay blocks	5	6
Hidden layer size	15 neurons	15 neurons
Output layer size	1 neuron	1 neuron
Epochs	1500	1500
Training samples	50000	50000

$$SSE = E_D \tag{5.20}$$

$$RMSE = \sqrt{\frac{1}{m} E_D} \tag{5.21}$$

The performance measures of plant NARXnet, trained using LM algorithm with different hidden neurons are given in table 5.3. Similarly, the performance measures when BR algorithm is used are given in table 5.4. Table 5.5 lists the MRE values of LM and BR algorithms.

The relative error (RE) is used to measure the deviation of the actual output from the target values. The formula for mean relative error (MRE) is given by equation 5.22. Where Y_i represents the target and $\hat{y}_i$ represents the actual neural network response.

$$MRE = \frac{1}{N} \sum_{i=1}^{N} \frac{Y_i - \hat{y}_i}{Y_i} \tag{5.22}$$

Table 5 lists the MRE values of LM and BR algorithms. The comparison of tables indicate that both LM and BR algorithms produced MSE of 1.63E-10 for 15 hidden neurons. The SSE and RMSE values are also similar, but the BR algorithm produced lowest gradient and MRE values 1.05E-07 and 3.82E-04 respectively. Both algorithms did not show early convergence behavior during the training. Based on the comparison, finally for the plant network BR algorithm with 15 hidden neurons are considered.

The comparison of step, load, and disturbance responses of proposed BR-Narxnet plant model and actual model of AVR system are shown in figures 5.12, 5.13, 5.14, and 5.15. From the figures, it is observed that the response produced by Narxnet model is almost identical to actual model of the AVR sytem.

Table 5.3 Performance of LM algorithm for plant network

Hidden neurons	MSE	SSE	SSW	Gradient	Mu
5	2.13E-10	1.07E-05	53.1069	1.23E-06	1.00E-08
10	1.64E-10	8.20E-06	24.465	2.15E-06	1.00E-09
15	1.63E-10	8.15E-06	25.1	1.99E-07	1.00E-09
20	1.66E-10	8.30E-06	24.1688	1.53E-06	1.00E-09

Table 5.4 Performance of Bayesian regularization algorithm for plant network

Hidden neurons	MSE	SSE	SSW	Gradient	Epochs	Mu	Effective parameters
5	1.97E-10	9.85E-06	111.623	7.87E-07	1500	5.00E+05	29.6
10	1.69E-10	8.45E-06	26.7	3.39E-07	1500	5.00E+05	56.5
15	1.63E-10	8.15E-06	22.68	1.05E-07	1500	5.00E+05	80
20	1.64E-10	8.20E-06	24.2482	1.18E-07	1500	5.00E+05	103

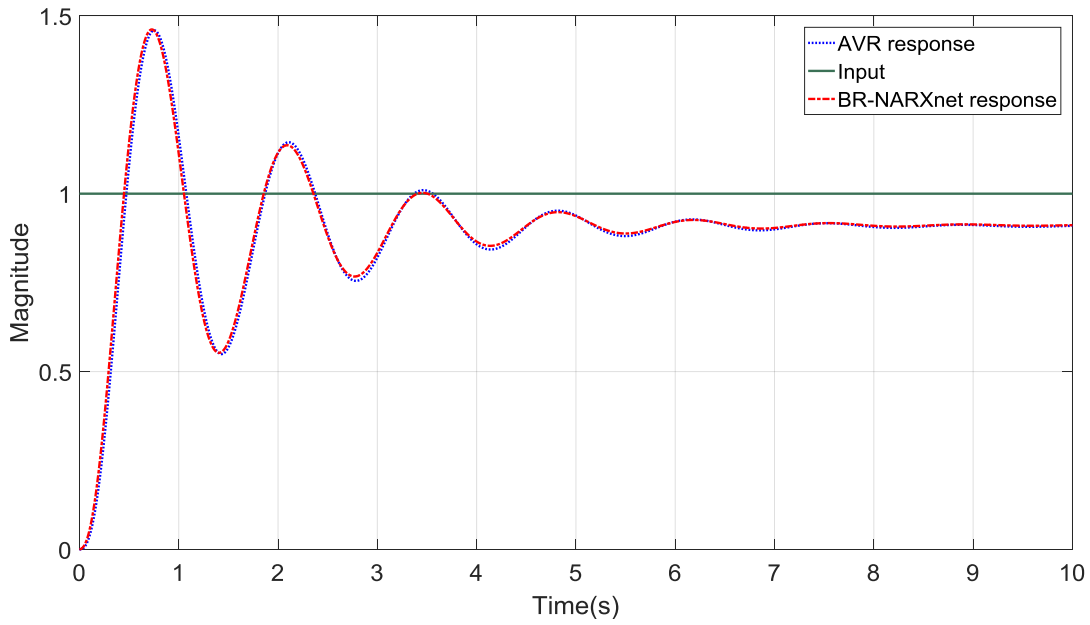

Figure 5.12 Comparison of step response of Narxnet model and AVR transfer function model

To further analyze the identified plant behavior various performance metrics like rise time, settling time, overshoot, undershoot, steady state error, and peak values are compared with actual model of the AVR system. The corresponding results are mentioned

Table 5.5 MRE and RMSE comparison for plant network

Hidden Neurons	BR (MRE)	LM (MRE)	BR (RMSE)	LM (RMSE)
5	0.0089	0.0078	1.40357E-05	1.4595E-05
10	0.0031	0.0019	1.3E-05	1.2806E-05
15	3.82E-04	0.0018	1.27671E-05	1.2767E-05
20	0.0048	8.52E-04	1.28062E-05	1.2884E-05

in the table 5.6 and figure 5.12. From the comparison it is concluded that the BR-NARXnet model of the plant produced more than 95% accuracy.

Table 5.6 Copmarison of performance metrics for plant

Parameter	AVR Model	BR-NARXnet Model
Rise time(s)	0.2687	0.2635
Settling time(s)	5.7055	5.6636
Overshoot(%)	60.4151	60.3395
Steady state error	0.09	0.089
Peak value	1.4592	1.4616
Undershoot(%)	0	0

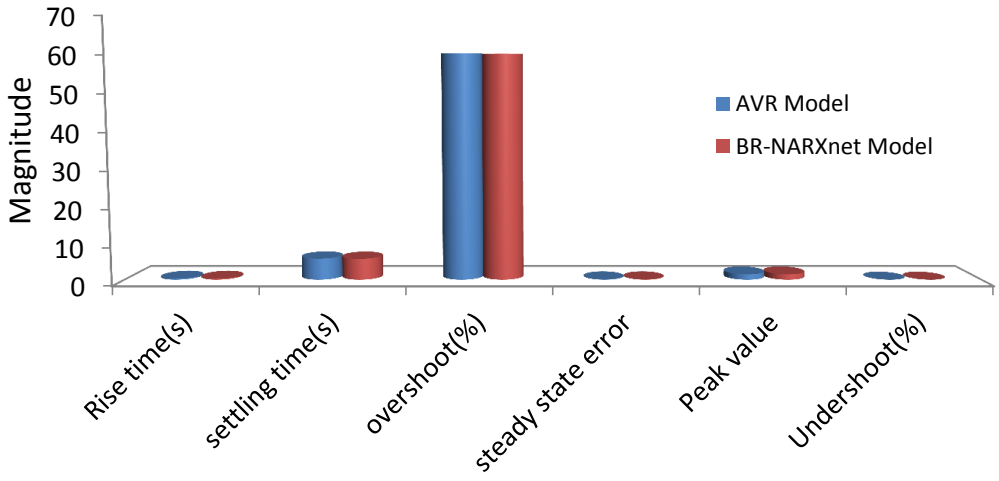

Figure 5.13 Comparison of performance metrics of actual AVR model and BR-NARXnet model

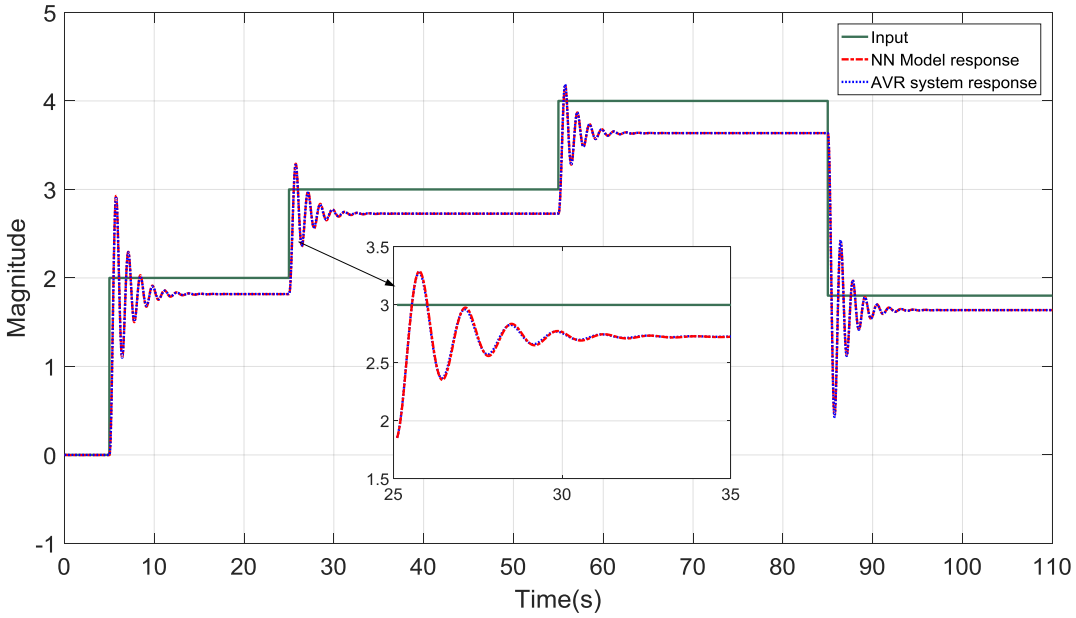

Figure 5.14 Comparison of load response of Narxnet model and AVR transfer function model

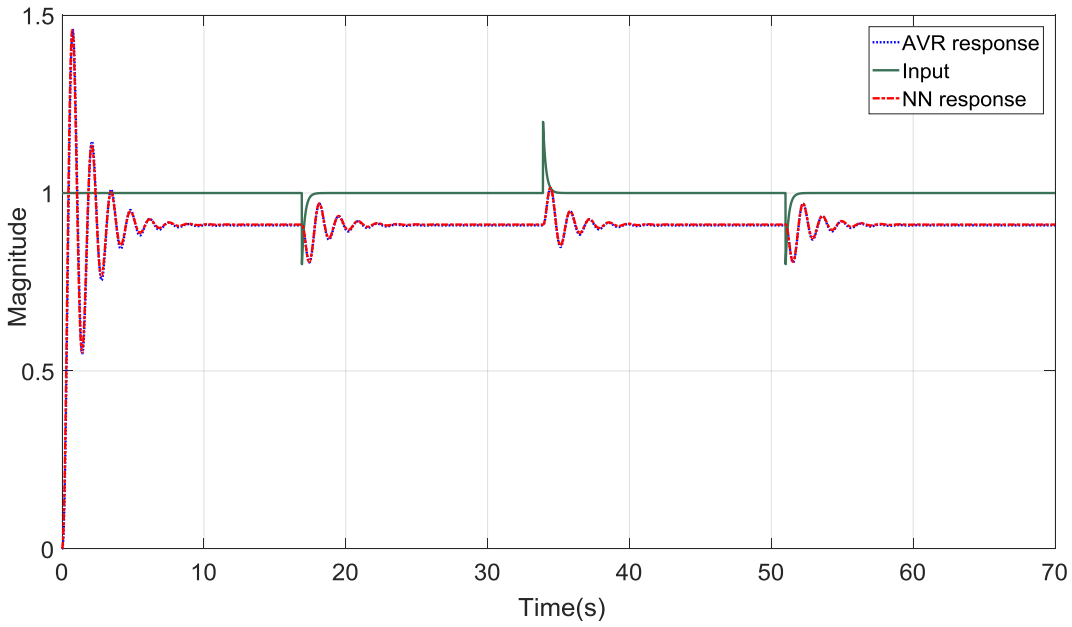

Figure 5.15 Comparison of disturbance response of NARXnet and AVR transfer function model

5.6.2 Analysis of controller training

A similar analysis discussed in section 7.2 is carried out on the controller during training. To study the effect of hidden neurons, the controller network is designed with 5, 10, 15 and 20 hidden neurons. Various metrics like MSE, SSE, RMSE and MRE are used to describe the controller efficiency. The controller network is trained using LM

and Bayesian regularization algorithms. The corresponding results obtained during the training process are tabulated as shown in tables 5.7, 5.8, and 5.9. On comparison of the tables it is found that BR algorithm produced the least mean square error of 3.22E-02 for 15 hidden neurons. It is also mentioned that the sum of squared error (SSE), gradient and RMSE values are better for BR algorithm than lm algorithm. The LM algorithm shown early convergence for 15 and 20 hidden neurons. Similarly BR algorithm shown early convergence for 5, 10, and 15 hidden neurons. Therefore, considering all these metrics it is decided to use BR algorithm to train the controller network with 15 hidden neurons.

Table 5.7 Performance of LM algorithm for controller network

Hidden neurons	MSE	SSE	SSW	Gradient	Epochs	Mu
5	1.05E+00	5.25E+04	27.099	1.33E+02	1500	1.00E+05
10	6.45E-02	3.22E+03	40.9933	6.10E-02	1500	1.00E+02
15	1.96E-01	9.80E+03	43.765	6.14E+08	7	5.00E+10
20	1.28E-01	6.38E+03	70.14	2.25E+00	17	5.00E+10

Table 5.8 Performance of Bayesian regularization algorithm for controller network

Hidden neurons	MSE	SSE	SSW	Gradient	Epochs	Mu	Effective parameters
5	5.63E-02	2.82E+03	32.9172	7.35E+00	148	5.00E+10	32.3
10	3.54E-02	1.77E+03	47.798	1.27E+00	614	5.00E+10	67.3
15	3.22E-02	1.61E+03	71.405	2.17E-02	1500	5.00E+06	78.6
20	6.86E-02	3.43E+03	21.159	1.52E+01	272	5.00E+10	32.2

5.6.3 Controller response for step input and set point changes

To investigate the efficiency of the proposed neural controller it is correlated with other meta-heuristic and stochastic optimization algorithm tuned controllers. Primarily the step input response is plotted for all the controllers as shown in figure 5.16. Cor-

Table 5.9 MRE and RMSE comparison of proposed controller

Hidden Neurons	BR (MRE)	LM (MRE)	BR (RMSE)	LM (RMSE)
5	7.10E-03	4.30E-02	0.2373	2.236068
10	5.25E-04	1.39E-02	0.1881	3.162278
15	9.56E-05	6.28E-04	0.1794	3.872983
20	2.38E-02	1.64E-02	0.2619	4.472136

respondingly, the controller parameter values for the FOPID/PID/PIDA controllers are given in table 5.10

From the step response, it can be observed that the NARXnet based controller shown improved performance than other FOPID/PID/PIDA controllers in terms of settling time, rise time, and overshoot. From the step response various performance measures are calculated and listed in table 5.10.

From the results, it is found that the proposed controller produced the rise time of 0.1052 seconds and took 0.4108 seconds to settle. Moreover, the proposed controller produced minimum overshoot value 2.9116%. The overall performance of each controller is calculated using the equation 5.23.

$$Overall\ Performance = Rise\ Time\ +\ Settling\ Time\ +\ Overshoot \tag{5.23}$$

From the table 5.10 it is concluded that the neural controller is performing better than the traditional optimization based controllers.

To further investigate, all the controllers are subjected to track different set points within a single run. The corresponding results are shown in the figure 5.17. The response curves show exactly how the proposed neural controller is able to track the load variations over the time. The results indicate the NARXnet controller is quickly tracking the set point changes than the traditional FOPID/PID/ PIDA controllers. The pictorial representation for the data mentioned in table **??** is shown in figures 5.18, 5.19, 5.20, and 5.21.

Finally, analysis of results indicates that the proposed BR-NARXnets can be used

Table 5.10 FOPID/PID/PIDA controller parameters

Algorithm	Kp	Ki	Kd	N	Lambda	Mu	Rise time	Settling time	Overshoot	Overall Performance
KIA-PID [86]	1.0426	1.0093	0.5999	NA	NA	NA	0.1276	0.7531	15.0041	15.8848
WOA-PIDA [85]	0.7847	0.9961	0.3061	NA	NA	NA	0.2151	2.1291	7.2357	9.5799
TLBO-PIDA [71]	0.9685	1	0.8983	1000	NA	NA	0.0957	1.3834	22.2811	23.7602
HAS-PIDA [71]	0.9519	0.9997	0.8994	NA	NA	NA	0.0957	1.3902	22.1461	23.632
PSO-PID [125]	0.708	0.656	0.282	NA	NA	NA	0.2377	0.8302	2.9905	4.0584
MOEO-FOPID [67]	2.9737	0.9089	0.5383	1000	NA	NA	**0.0782**	**0.4108**	7.5121	8.0011
LUS-PID [82]	1.2012	0.9096	0.4593	NA	NA	NA	0.149	0.7997	15.5623	16.511
IABC-PID [70]	1.9605	0.4922	0.2355	NA	1.4331	1.55	0.2113	1.8759	3.1923	5.2795
ChMO-FOPID [65]	0.408	0.374	0.1773	NA	1.3336	0.68	0.8867	4.6227	3.6933	9.2027
ChAS-FOPID [68]	1.0537	0.4418	0.251	NA	1.1122	1.06	0.2201	1.6487	3.4577	5.3265

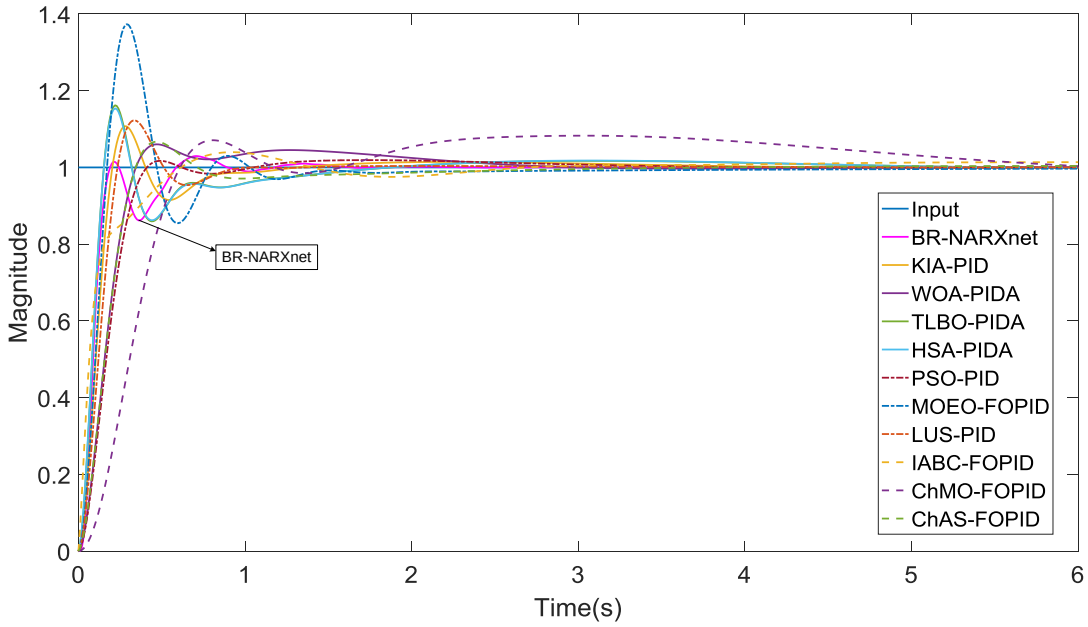

Figure 5.16 Comparison of step response

for identification and control of AVR system. From the comparison it has been shown that NARXnets perform very good for system identification and satisfactory performance for controlling. To further improve the BR-NARXnet performance, specifically for controlling applications, it is suggested to use hybrid cost functions that include performance metrics like overshoot, settling time, rise time, ITSE, and ITAE in addition to MSE.

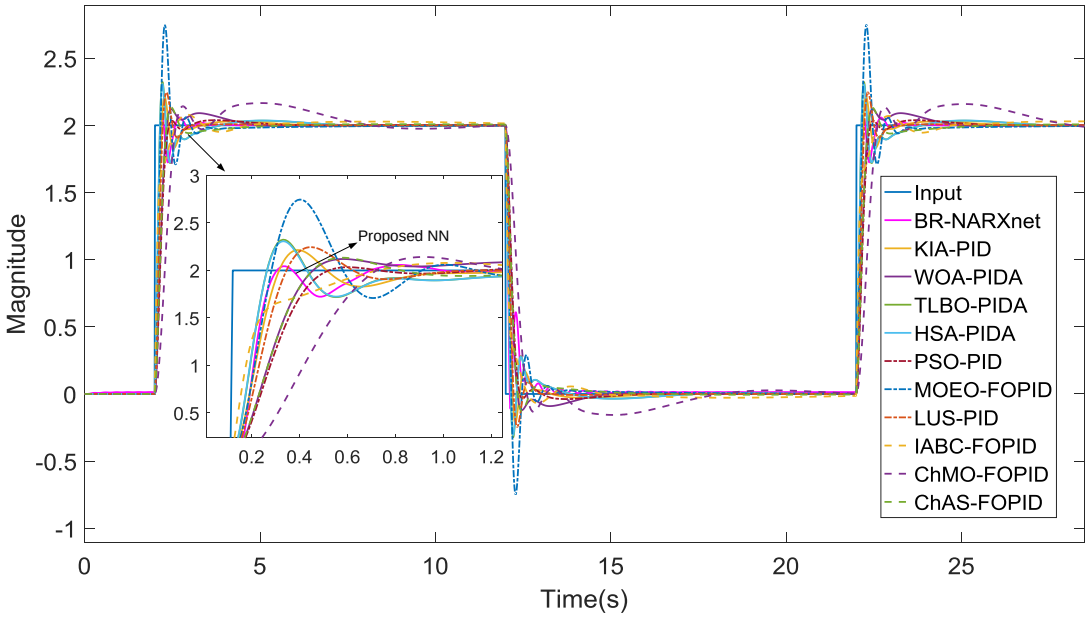

Figure 5.17 Load response comparison for different controllers

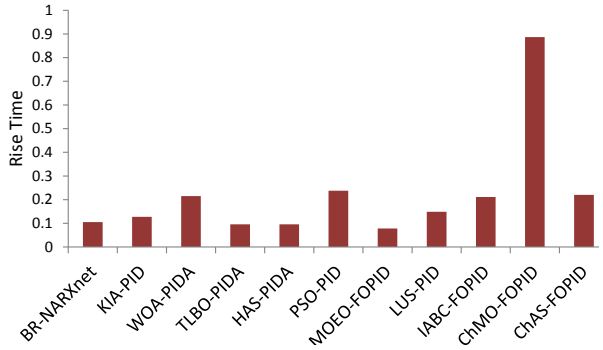

Figure 5.18 Comparison of rise time

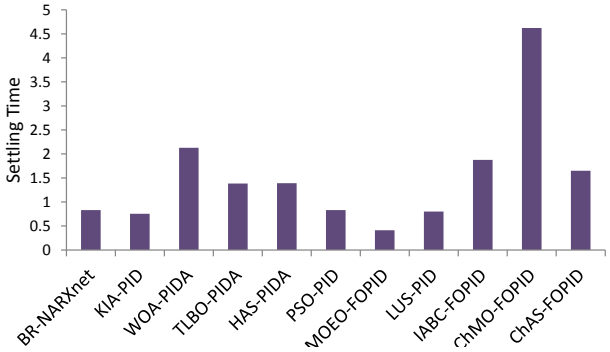

Figure 5.19 Comparison of settling time

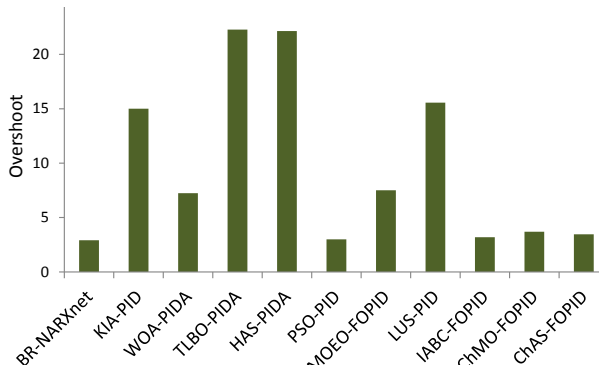

Figure 5.20 Comparison of overshoot

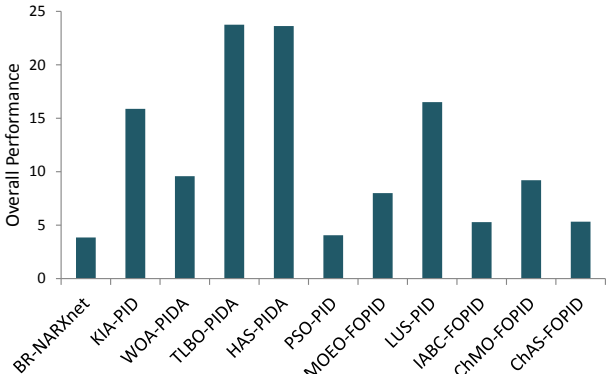

Figure 5.21 Comparison of overall performance

5.7 Conclusion

For the modeling and control of dynamic systems, a neural network-based solution employing BR-NARXnets has been presented. To verify the proposed methodology AVR system is considered and its response is optimized using NARXnets. The neural network training data is generated using the Simulink model of the actual AVR system. The BR-NARXnets are used to identify the AVR system dynamics. The comparison of actual model response and NARXnet response has shown very good correlation. Later the trained plant network is utilized to identify the controller. During the training Bayesian regularization back propagation algorithm and Levenberg-Marquardt algorithms are used. On comparison of performance metrics like MSE, MRE and gradient values it is found that BR algorithm performs better. The influence of the hidden layer neurons on the system's performance is investigated. The performance of the controller is compared to existing state-of-the-art approaches. The suggested BR-NARXnet controller outperformed existing approaches in terms of overshoot and combined performance metric values. Moreover,

the developed procedure can be used as a generalized method to design NARXnet controllers for various types of dynamic systems.

Chapter 6

A novel sigmoid PID (SPID) Controller for AVR System using Jellyfish Search Optimization algorithm

6.1 Sigmoid PID controller

The proportional integral derivative controllers are famously known for their simple architecture, robust operation, and ease of implementation properties. The PID controller consists of fixed values of controller parameters known as proportional gain (K_p), integral gain (K_i) and derivative gain (K_d). Although these fixed gains are sufficient to give the required control action, they are not precisely sensitive to the error, limiting the controller response. Therefore, variable gain PID controllers are used in many control applications [126,127] to improve the sensitivity of the PID controller and to enhance the controller reaction. On the other hand FOPID controllers will give additional degrees of freedom to the existing PID controllers whose gain values are fixed during the course of operation. Therefore these controllers cannot improve the sensitivity of the parameters. Therefore efforts have been put on a new type of controller called sigmoid PID controller whose gains can be varied during the operation according to the error.

The sigmoid PID controller which is a modified version of variable gain PID controller was initially mentioned in [126]. The sigmoid PID(SPID) controller continuously varies the gains of PID controller (K_p, K_i, and K_d) according to the magnitude of the error signal. The controller internally uses sigmoid activation function to limit the gain parameters within the specified range. The sigmoid function is famous activation function

used in artificial neural networks. A mathematical description governing the operation of SPID controller is given by the following equations.

The proportional gain is varied according to equation (6.1)

$$K_{pv}(t) = K_{plo} - \left| \frac{K_{phi} - K_{plo}}{1 + exp(-\sigma_p |e(t)|)} \right| \tag{6.1}$$

considering $\delta_p = K_{phi} - K_{plo}$, then

$$K_{pv}(t) = K_{plo} - \left| \frac{\delta_p}{1 + exp(-\sigma_p |e(t)|)} \right| \tag{6.2}$$

The integral gain equation is given by

$$K_{iv}(t) = K_{ilo} - \left| \frac{K_{ihi} - K_{ilo}}{1 + exp(-\sigma_i |e(t)|)} \right| \tag{6.3}$$

considering $\delta_i = K_{ihi} - K_{ilo}$, then

$$K_{iv}(t) = K_{ilo} - \left| \frac{\delta_i}{1 + exp(-\sigma_i |e(t)|)} \right| \tag{6.4}$$

Similarly the equation for derivative gain is given by

$$K_{dv}(t) = K_{dlo} - \left| \frac{K_{dhi} - K_{dlo}}{1 + exp(-\sigma_d |e(t)|)} \right| \tag{6.5}$$

considering $\delta_d = K_{ihi} - K_{ilo}$, then

$$K_{dv}(t) = K_{dlo} - \left| \frac{\delta_d}{1 + exp(-\sigma_d |e(t)|)} \right| \tag{6.6}$$

In the equations (6.1)-(6.6), the parameters K_{phi}, K_{ihi}, and K_{dhi} represents the higher bounds and K_{plo}, K_{ilo}, and K_{dlo} represents lower bounds of proportional, integral and derivative gains respectively. The coefficients σ_p, σ_i, and σ_d are used to adjust the sharpness of sigmoid curve as shown in figure 6.1.

The mathematical representation for SPID controller is obtained by summing all the gains as mentioned in equation (6.7).

$$SPID = K_{pv}(t) + \frac{K_{iv}(t)}{s} + K_{dv}(t) * s \tag{6.7}$$

The advantage of sigmoid PID controller is it adjusts the controller gains within the limited boundary according to the variations in the error signal.In the case of traditional PID controller these gains are constant irrespective of changes in the error value.

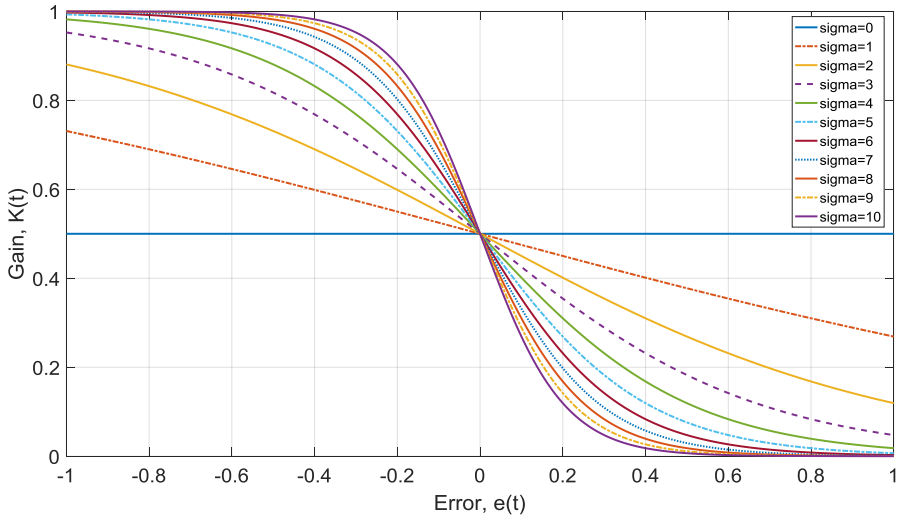

Figure 6.1 Variation in sigmoid function with σ

6.2 Jellyfish Search Optimization (JSO) Algorithm

Artificial jellyfish search optimization is a meta-heuristic algorithm proposed in [128]. The algorithm is developed according to the food searching nature of jellyfish in the ocean. The jellyfish exhibits different types of motions depending on the availability of food sources and ocean currents. In general the ocean currents carry good amount of nutrients. Therefore while moving along the ocean currents the jellyfish swarms are formed. Inside the swarm they exhibit two varieties of motions known as passive and active motions. During the passive motion the jelly fish move nearby to its present location. In the active motion they will move inside the swarm as a whole searching for better food locations. The jellyfish blooms are formed when the availability of food is maximum.

A mathematical model is developed simulating the ocean currents and different types of jellyfish motions. The switching between various motions is governed by a time control mechanism. The quantity of available food is mapped to the cost function. The algorithm tries to find the optimum solution using the objective function value similar to the jellyfish movement.

The JSO algorithm is presented in the form of series of steps as mentioned below.

Step 1: Set the initial parameters like population size, number of iterations and boundaries of search space.

Step 2: Initialize the jellyfish population (ξ_i) using logistic map

$$\xi_{i+1} = \mu * \xi_i * (1 - \xi_i) \tag{6.8}$$

Since the random numbers should be in the range of $(0, 1)$, in the algorithm the value of μ is set to 4.

Step 3: Calculate the quantity of food at each location by evaluating the objective function $f(\xi_i)$ and find the current best location ξ^*.

Step 4: Calculate the control time $\nu(t)$ using the equation

$$\nu(t) = \left| \left(1 - \frac{t}{max\_iter}\right) * (2 * rnd(0,1) - 1) \right| \tag{6.9}$$

Step 5: When the control time $\nu(t) \geq 0.5$, the jellyfish follows ocean current. The ocean current direction and the jellyfish positions are updated using the equations

$$\overrightarrow{current} = \xi^* - \beta * rnd(0,1) * \mu \tag{6.10}$$

where β is called distribution coefficient and its value is considered as 3 [128].

$$\xi_i(t+1) = \xi_i(t) + rnd(0,1) * \overrightarrow{current} \tag{6.11}$$

Step 6: When the control time $\nu(t) < 0.5$ the jellyfish moves inside swarm and exhibit passive and active motions When $rand(0,1) < 1 - \nu(t)$, then passive motion is exhibited and the new positions are given by

$$\xi_i(t+1) = \xi_i(t) + \gamma * rnd(0,1) * (U_b - L_b) \tag{6.12}$$

Where γ is called motion coefficient and its value is identified as 0.1 from the sensitivity analysis.

When $rand(0,1) > (1-\nu(t))$, then active motion is exhibited and the new current direction and positions are given by

$$\overrightarrow{Direction} = \begin{cases} \xi_i(t+1) = \xi_j(t) - \xi_i(t) \; if \; f(\xi_i) \geq f(\xi_j) \\ \xi_i(t+1) = \xi_i(t) - \xi_j(t) \; if \; f(\xi_i) < f(\xi_j) \end{cases} \tag{6.13}$$

$$\xi_i(t+1) = \xi_i(t) + rnd(0,1) * \overrightarrow{Direction} \tag{6.14}$$

Step 7: In the steps 5 and 6 check the boundary conditions and evaluate the quantity of food using $f(\xi_i)$ and update the best position ξ^*.

Step 8: Repeat the steps 3 to 7 until the maximum number of iterations reached or the termination criteria is met. The detailed description of the algorithm along with convergence profiles can be found in [128]. For the implementation of JSO algorithm the hyper parameters spatial distribution (β) and motion coefficient (γ) are taken as 3 and 0.1 respectively. These values are identified by varying the values of β and γ in the range [0.510] and [0.051] respectively.

6.3 Implementation

6.3.1 Proposed JSO-Sigmoid PID controller

In the proposed method, the JSO algorithm is used to identify the optimum values of the SPID controller parameters. Initially, the error value $V_e(s)$ is calculated from the AVR system response. The error is then used to compute the objective function value. The JSO controller generates the initial controller parameters. Based on these parameters, controller generates the control signal $U(s)$ to minimize the difference between the set point $V_{Ref}(s)$ and the AVR system output. Then the error is again calculated from the AVR system response, and the process is repeated until the error is minimized. The aforementioned process is depicted in the figure 6.2.

To perform the simulations, the JSO algorithm is implemented in MATLAB, SPID controller and plant are developed using Simulink. When parameters generated by JSO algorithm are sent to Simulink and model response is calculated. Using the simulink model response, the objective function calculates the cost value and new set of parameters are generated and the process is repeated until maximum number of iterations.

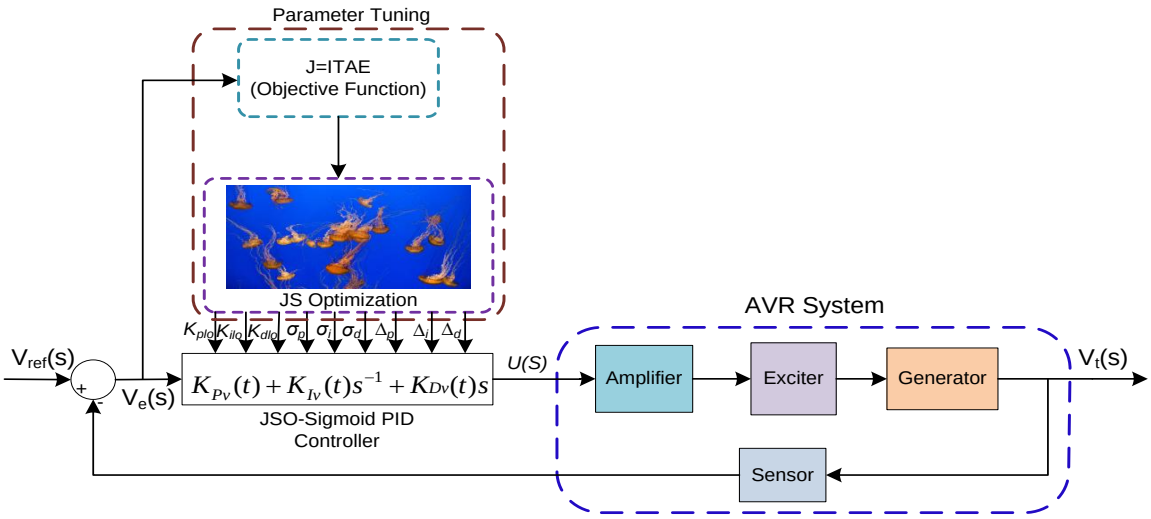

Figure 6.2 Proposed JSO-SPID controller block diagram

6.3.2 Objective function

Various objective functions [73] are used in the literature to identify the optimum controller parameters for the AVR system. These functions vary from simple IAE (integral of absolute error) function to functions involving weighted combinations of performance metrics. Even though the objective functions used in the existing methods are able to optimize individual performance metrics and overall system error they suffer from high computational complexity. To reduce number computations without compromising the controller performance, ITAE (Integral of time multiplied absolute error) is used in the proposed method. Moreover, it is mentioned that after several simulations on various objective functions such as IAE, ISE (integral of squared error), ITAE (integral of time multiplied absolute error), ITSE (integral of time multiplied squared error), and FOD (figure of demerit), it is found that the ITAE has given better controller performance.

$$ITAE = \int_0^\infty t * |e(t)|dt \tag{6.15}$$

The JSO algorithm is run with the population of 100 for 300 iterations to produce the optimum SPID controller parameters. The convergence behavior of the JSO algorithm during the optimization of SPID controller parameters is given in figure 6.3. The convergence curve shows that the algorithm converged in 100 iterations, after that the variation in cost

function is very less. The detailed block diagram describing the complete optimization procedure is shown in figure 6.4.

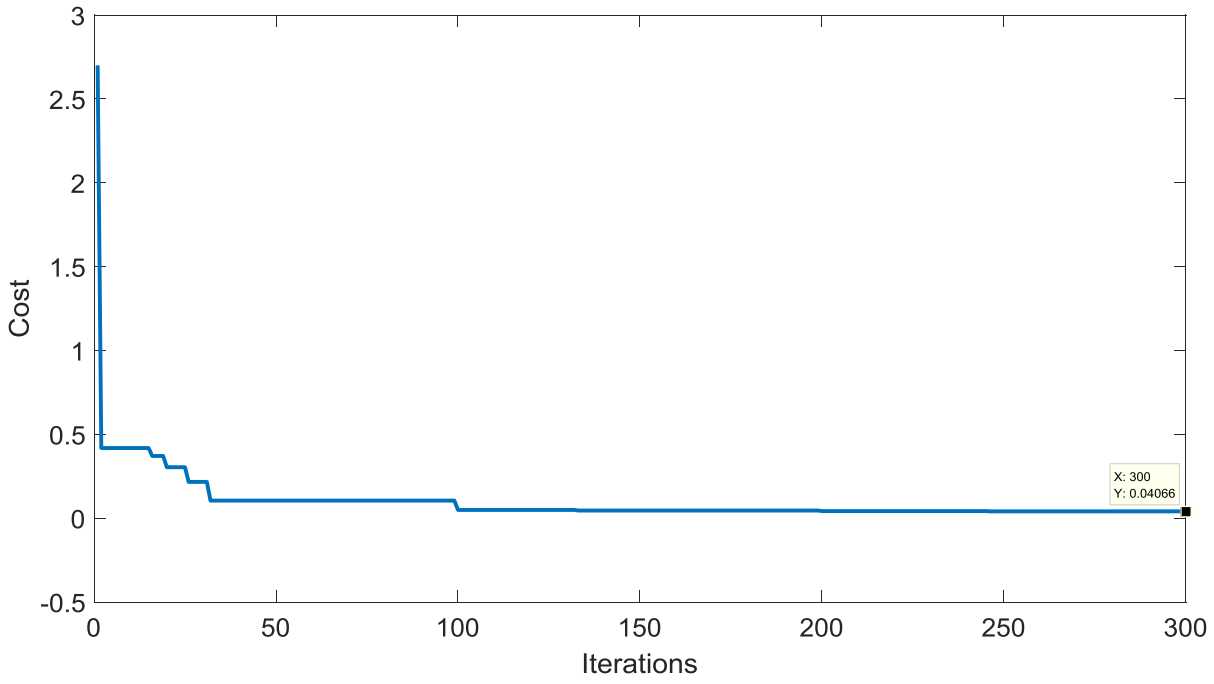

Figure 6.3 Convergence curve of JSO-SPID controller

The range of SPID controller parameters and their optimum values obtained using JSO algorithm are listed in table 6.1. For all the controller gains the lower bound is taken as 3 and upper bound is taken as 20. The upper and lower bounds of the SPID controller parameters are selected by analyzing the existing literature related the FOPID/PID controller parameter values.

Substituting the values mentioned in table 6.1 into the equations (6.2), (6.4), and (6.6) results the equations (6.16), (6.17), and (6.18) respectively.

$$K_{pv}(t) = 1.95 - \left| \frac{5.77}{1 + exp(-2.18 * |e(t)|)} \right| \tag{6.16}$$

$$K_{iv}(t) = 1.95 - \left| \frac{1.46}{1 + exp(-2.12 * |e(t)|)} \right| \tag{6.17}$$

$$K_{dv}(t) = 0.56 - \left| \frac{18.04}{1 + exp(-0.33|e(t)|)} \right| \tag{6.18}$$

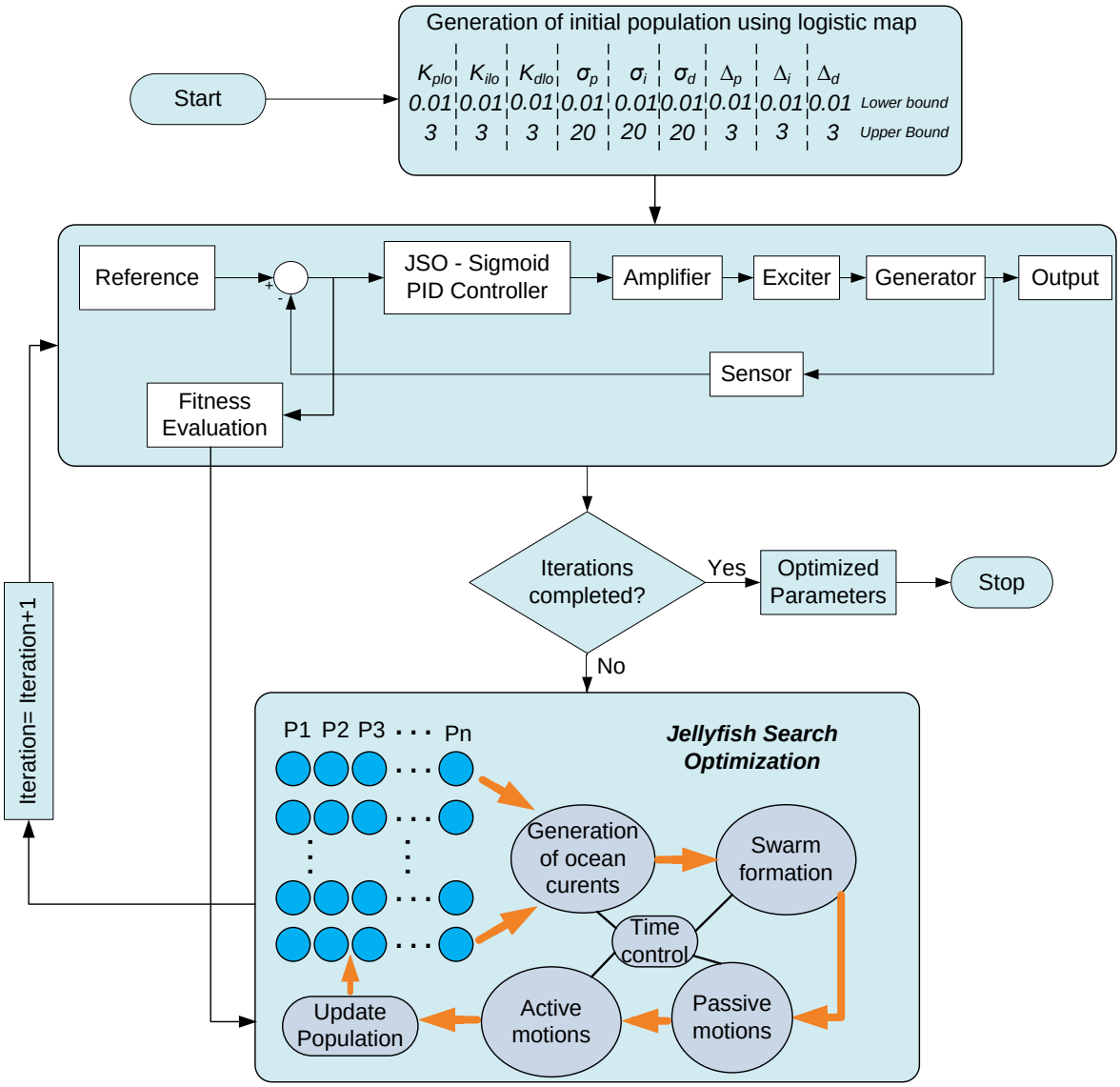

Figure 6.4 Optimization of SPID controller parameters using JSO algorithm

6.4 Results & Discusion

To inspect the efficiency of the proposed sigmoid PID controller different types of analysis like step, load, disturbance, and tracking a reference signal are carried out. The proposed JSO-SPID controller is compared with the existing state-of-the-art PID controllers. All the simulations are performed using Matlab®version 2016b on a system with intel®core i5 processor and 8GB RAM.

Table 6.1 Optimized JSO-SPID controller parameter values

Parameter	Lower bound	Upper bound	Identified value
K_{plo}	0.01	3	2.6913741
K_{ilo}	0.01	3	1.9514011
K_{dlo}	0.01	3	0.5649716
δ_p	0.01	20	5.7705772
δ_i	0.01	20	1.4594253
δ_d	0.01	20	18.039579
σ_p	0.01	3	2.1806829
σ_i	0.01	3	2.117246
σ_d	0.01	3	0.331519

6.4.1 Step response analysis

The controller behavior is analyzed by applying step input as the reference signal. The step response of the AVR system with JSO-SPID controller is calculated. From the response various performance metrics like rise time, settling time, overshoot, and steady state error values are computed. The proposed system performance is compared with the existing NSCA-SPID and PID controllers.

The step response comparison plots are shown in figure 6.5 and the corresponding performance metrics are mentioned in the table 6.2.

For better visualization and comparison purpose the performance metrics of various SPID/PID controllers are represented as bar charts and shown in figure 6.6, 6.7 and 6.8. From the comparison of performances, it is found that the proposed JSO-SPID controller produced lowest value of overshoot, peak error and settling time values with little increase in rise time. When compared with existing NSCA-SPID controller the JSO-SPID controller produced better performance metrics. Moreover it is also identified that the oscillations are minimum in the proposed controller response when compared with all the existing state of the art SPID/PID controllers. If we look at the SPID and PID controllers as a whole it is interesting to note that the SPID controllers produced very low overshoot values than the existing PID controllers.

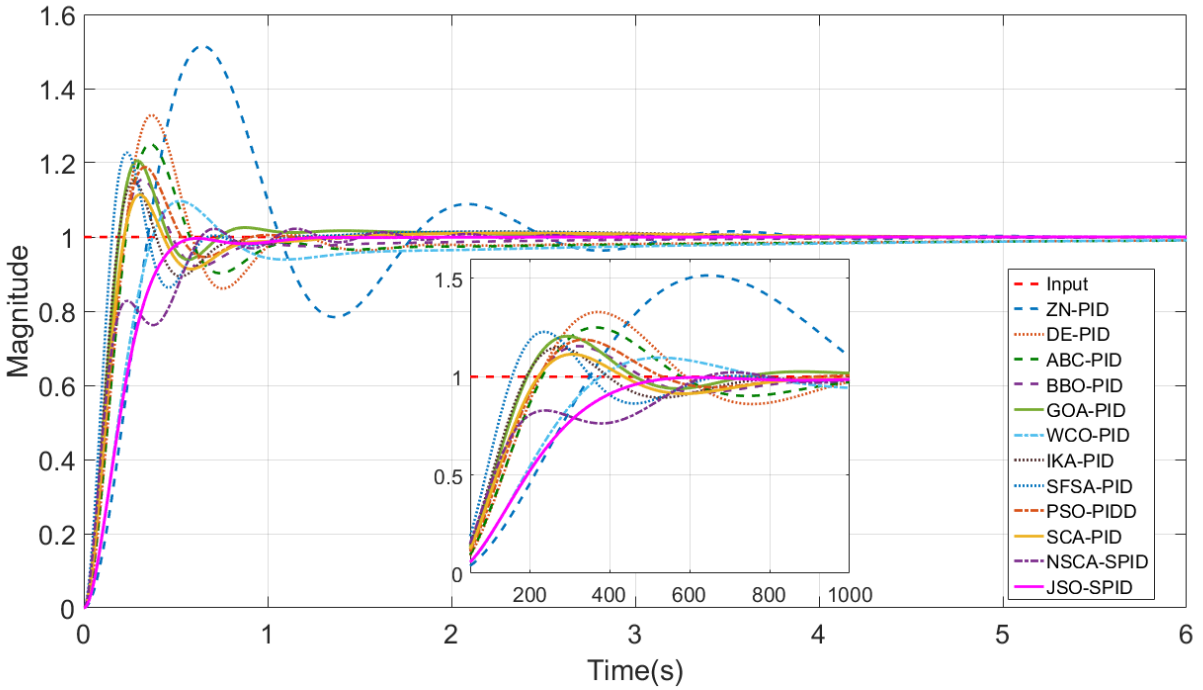

Figure 6.5 Comparison of step response of different controllers

Table 6.2 Step response comparison of different PID/SPID controllers

Controller	Rise time	Settling time	Overshoot	Peak value	Peak error
ZN-PID [129]	0.236487	3.0526	51.4973	1.515	0.515
DE-PID [80]	0.150608	1.7835	33.3775	1.3287	0.3287
ABC-PID [80]	0.15489	2.4597	25.4783	1.2504	0.2504
BBO-PID [130]	0.148284	1.4417	15.5482	1.1553	0.1553
GOA-PID [87]	0.129555	0.9716323	20.5827	1.2058	0.2058
WCO-PID [131]	0.250857	3.3705	9.9123	1.0967	0.0967
IKA-PID [86]	0.127554	0.7541113	15.0038	1.1501	0.1501
SFSA-PID [62]	**0.103659**	0.9545286	22.7783	1.2278	0.2278
PSO-PID [132]	0.149089	0.8155078	18.8382	1.1884	0.1884
SCA-PID [72]	0.149089	0.8155078	18.8382	1.1884	0.1884
NSCA-SPID [133]	0.499462	1.185	2.1668	1.0221	0.0221
JSO-SPID (Proposed)	0.319407	**0.507679**	**0**	**1**	**0**

6.4.2 Analysis of JSO-SPID parameters

When compared with the traditional PID controllers, the SPID controllers consists of variable controller gains represented as $K_p(t)$, $K_i(t)$, and $K_d(t)$. The controller gains vary according to the error value. The system error $e(t)$ versus gain plots are given in

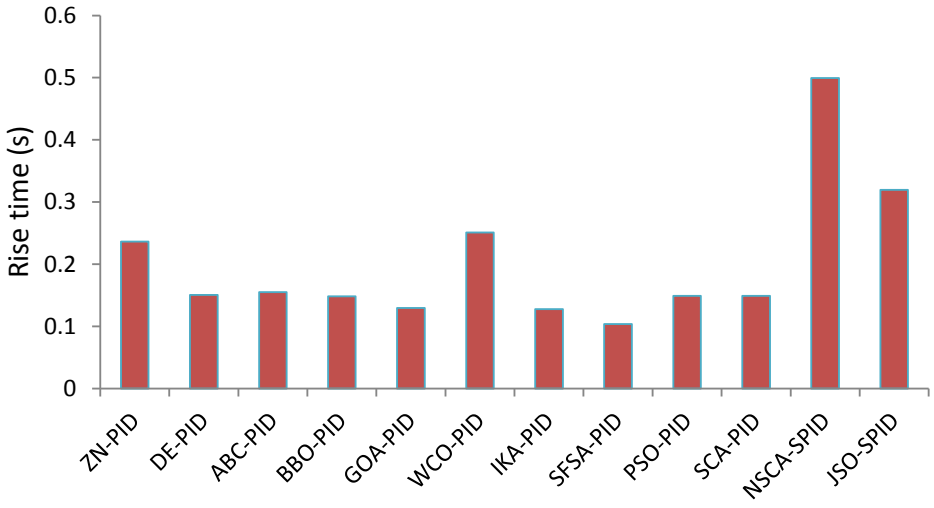

Figure 6.6 Comparison of rise time

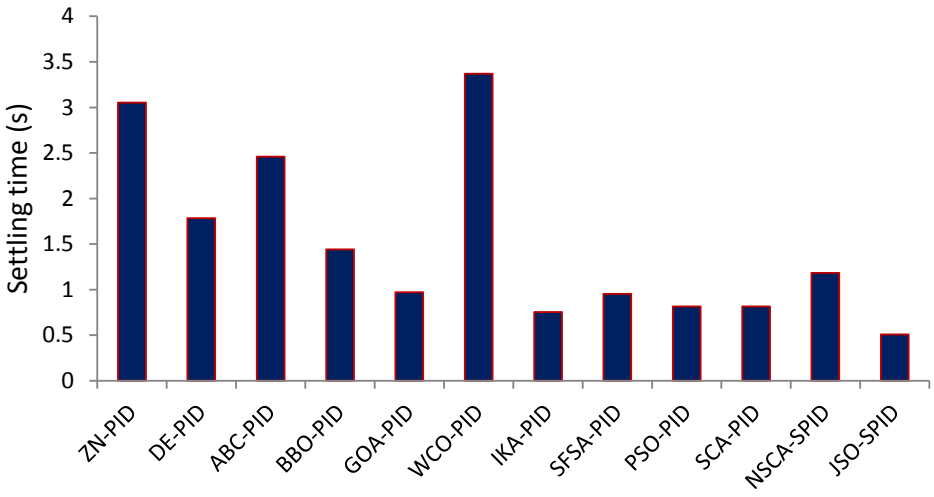

Figure 6.7 Comparison of settling time

figures 6.9, 6.10, and 6.11. For the step response considered in section 6.1, the variation of controller gain values ($K_{pv}(t), K_{iv}(t)$, and $K_{dv}(t)$) and error value are given in figure 6.12, 6.13, and 6.14.

6.4.3 Robust analysis

The JSO-SPID controller reliability is verified by varying the time constants of different blocks of AVR system in the range of -50% to +50%. In each case, the controller response is calculated using step signal as reference signal. From the step response of

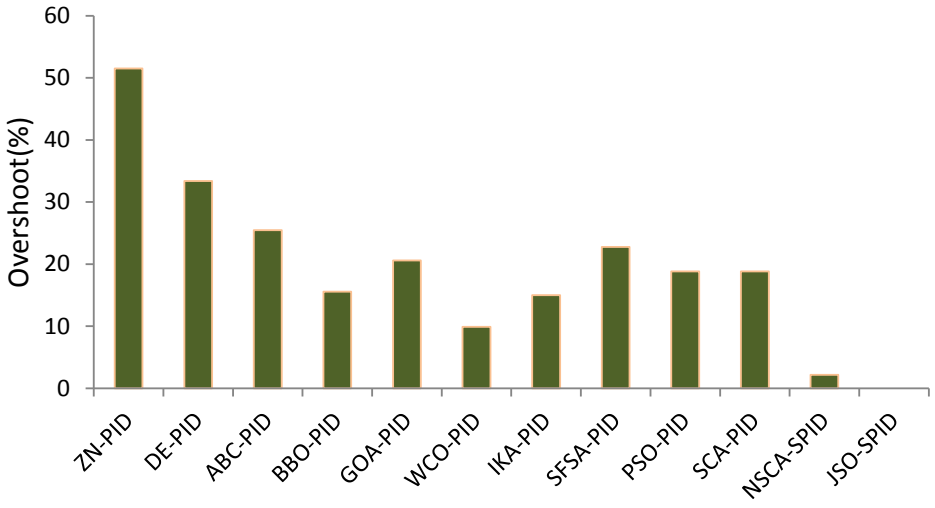

Figure 6.8 Comparison of overshoot

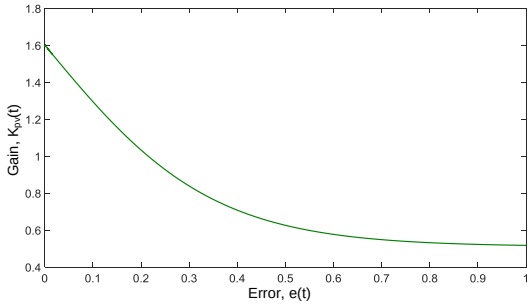

Figure 6.9 $K_{pv}(t)$ vs e(t)

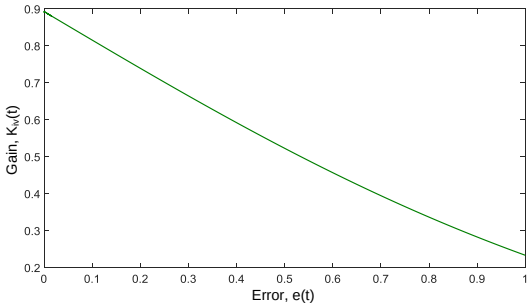

Figure 6.10 $K_{iv}(t)$ vs e(t)

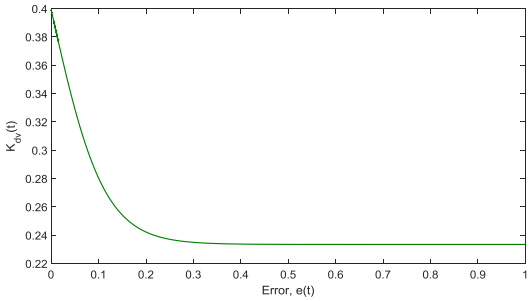

Figure 6.11 $K_{dv}(t)$ vs e(t)

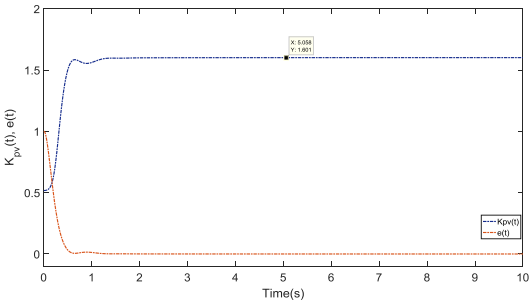

Figure 6.12 $K_{pv}(t)$ and e(t)

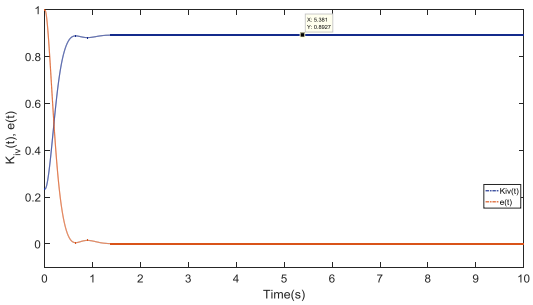

Figure 6.13 $K_{iv}(t)$ and e(t)

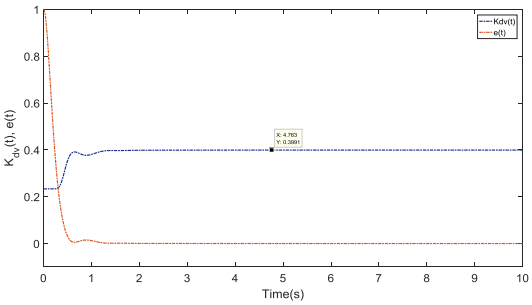

Figure 6.14 $K_{pv}(t)$ and e(t)

the system various performance metrics like rise time, settling time, overshoot and peak error values are calculated and tabulated as shown in table 6.3. Correspondingly, the step response for variations in the AVR system parameters τ_a, τ_e, τ_g, and τ_s are shown in the figures 6.15, 6.16, 6.17, and 6.18 respectively.

Table 6.3 Robust analysis of proposed JSO-SPID controller

Parameter	Deviation(%)	Value	Rise time(s)	Settling time(s)	Overshoot(%)	Peak
τ_a	50	0.15	0.326937	1.1759	3.6668	1.0367
	25	0.125	0.3205197	1.0612	1.6295	1.0163
	-25	0.075	0.3315133	0.6018283	0	1
	-50	0.05	0.3683437	0.6372062	0	1
τ_e	50	0.6	0.4115553	1.0167	3.8873	1.0389
	25	0.5	0.3675781	0.5522281	1.8721	1.0187
	-25	0.3	0.2662282	0.9037487	0	1
	-50	0.2	0.207573	0.815204	0	1
τ_g	50	1.5	0.4784809	1.1707	2.7522	1.0276
	25	1.25	0.4007993	0.6215516	1.021	1.0102
	-25	0.75	0.2406998	0.9344763	0.4083	1.0041
	-50	0.5	0.1688934	1.2033	4.2544	1.0425
τ_s	50	0.015	0.3099309	0.9318803	0	1
	25	0.0125	0.3146207	0.4970452	0	1
	-25	0.0075	0.3242642	0.5181904	0	1
	-50	0.005	0.3291654	0.5284382	0	1

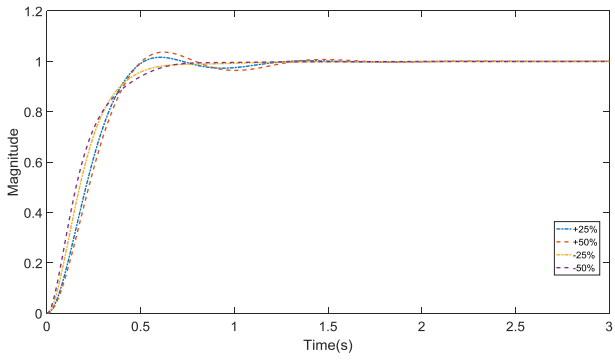

Figure 6.15 Response for variations in τ_a

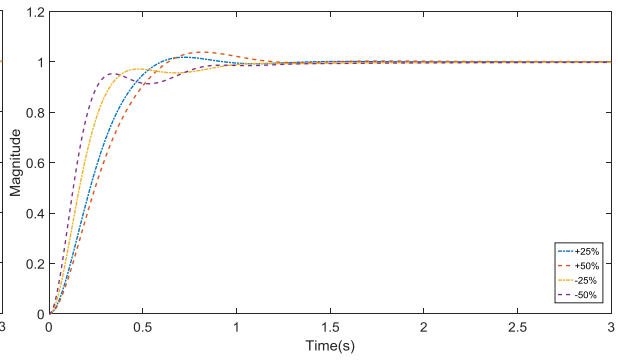

Figure 6.16 Response for variations in τ_e

From the robust analysis, shown in table 6.3, we can observe that the controller is able to maintain its optimum performance even though there is variation of parameters from -50% to +50%. These values indicates that the controller is robust for parameter variations in the plant. In spite of significant variations in the AVR system parameters

Table 6.4 Quantitative analysis for robustness of JSO-SPID controller

Measures	Rise time (s)	Settling time (s)	Overshoot (%)
Optimum value	0.3194	0.5077	0
Max. value	0.4784	1.1759	4.25
Min. value	0.31	0.497	0
Max. deviation from optimum value	0.159	0.6682	4.25
Min. deviation from optimum value	0.0094	0.0107	0
Max. deviation (%)	33.23	66.82	4.25
Min. deviation (%)	2.94	2.11	0

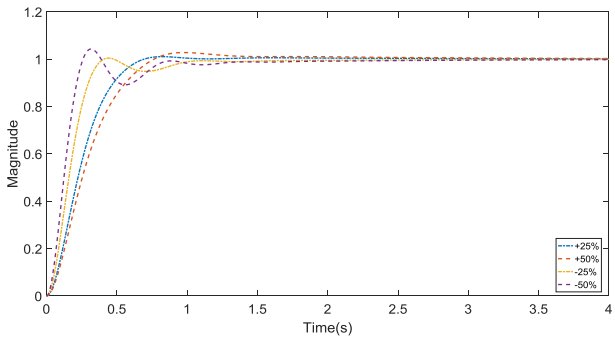

Figure 6.17 Response for variations in τ_g

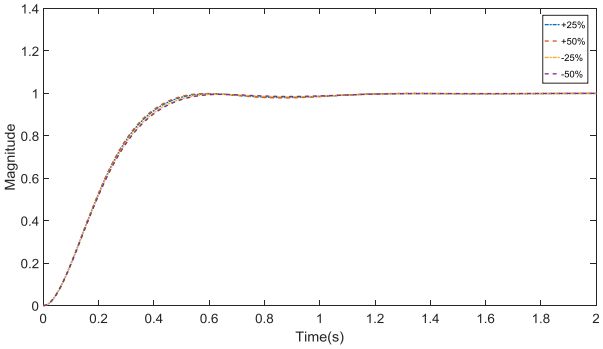

Figure 6.18 Response for variations inτ_s

the controller produced a maximum rise time of 0.4784 seconds, settling time of 1.1759 seconds and overshoot of 4.25%. The quantitative evaluation of the robust analysis results are given in table 6.4.

6.4.4 Load response

To investigate the effect of load on the JSO-SPID controller different set points are given in the single run. The load response of different controllers is shown in the figure 6.19. The load response helps to identify the controller behavior for variations in the target signal or set point changes. The response of NSCA-SPID and JSO-SPID controllers for load variations is given in the figure 6.20. From the figure, it is clearly observed that the response produced by the proposed JSO-SPID controller is much smoother than the NSCA-SPID controller.

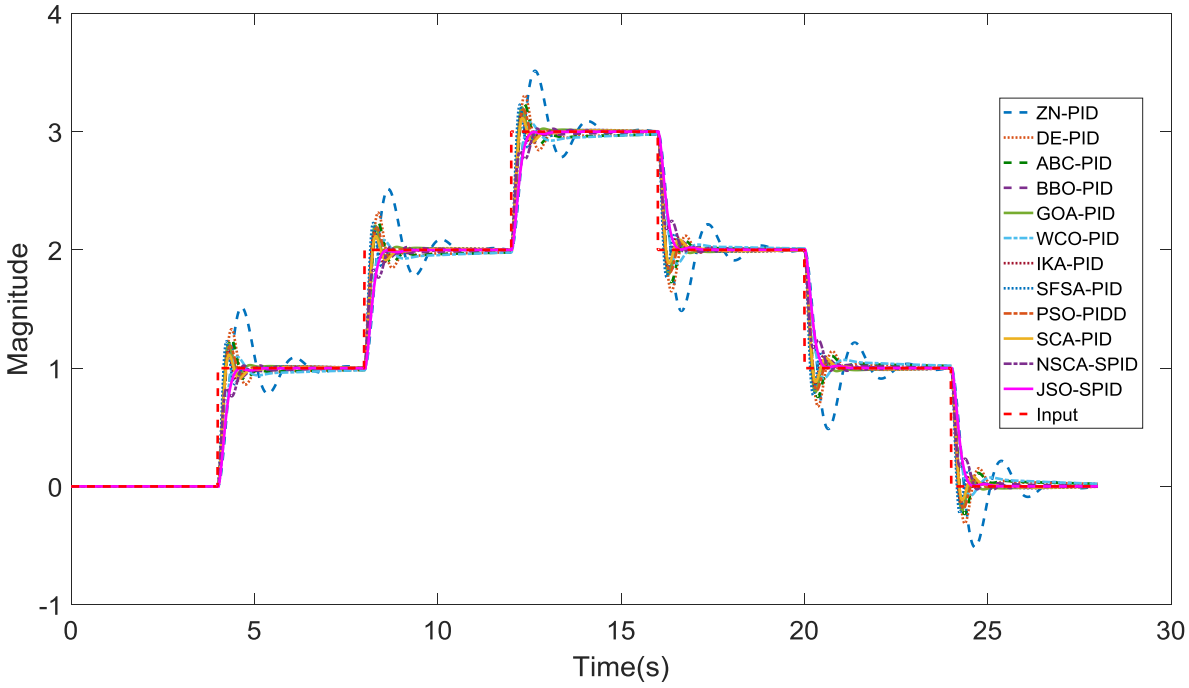

Figure 6.19 Comparison of load response of SPID/PID controllers

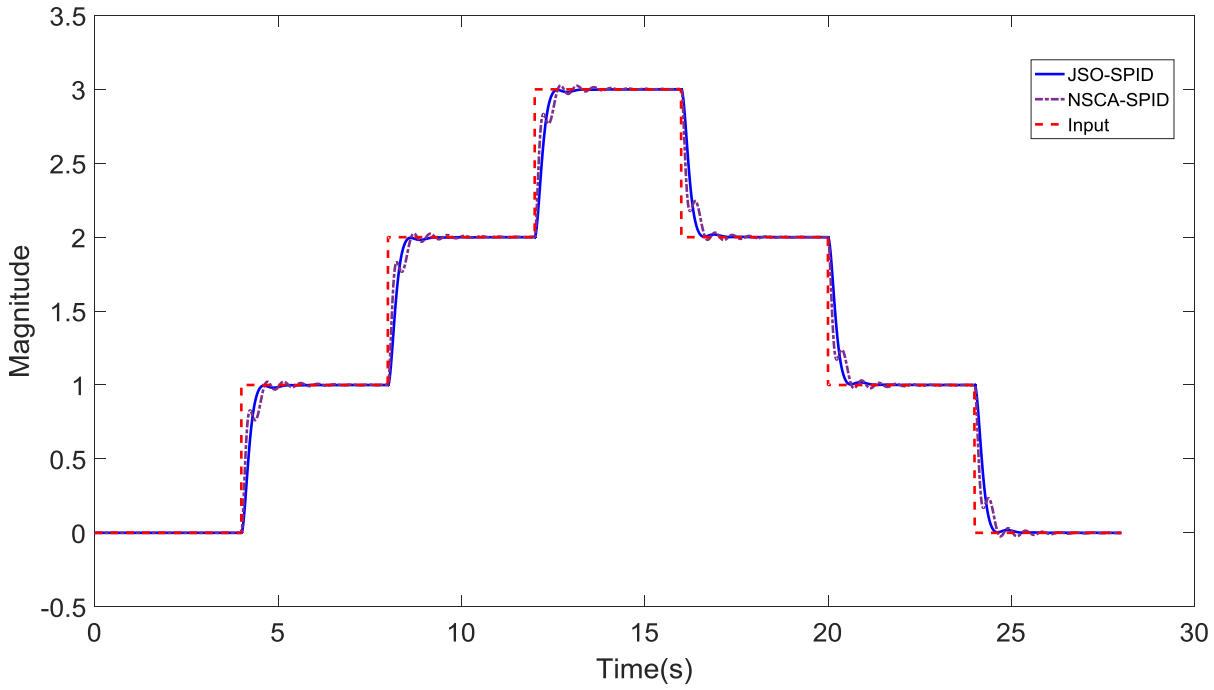

Figure 6.20 Disturbance response of JSO-SPID and NSCA-SPID controllers

6.4.5 Disturbance analysis

To further inquire the controller behavior under sudden input changes disturbance analysis is carried out. During the analysis the proposed JSO-SPID controller is compared

with the existing state of the art methods mentioned in the literature. The corresponding results are shown in the figure 6.21. From the figure, it is clearly observed that the

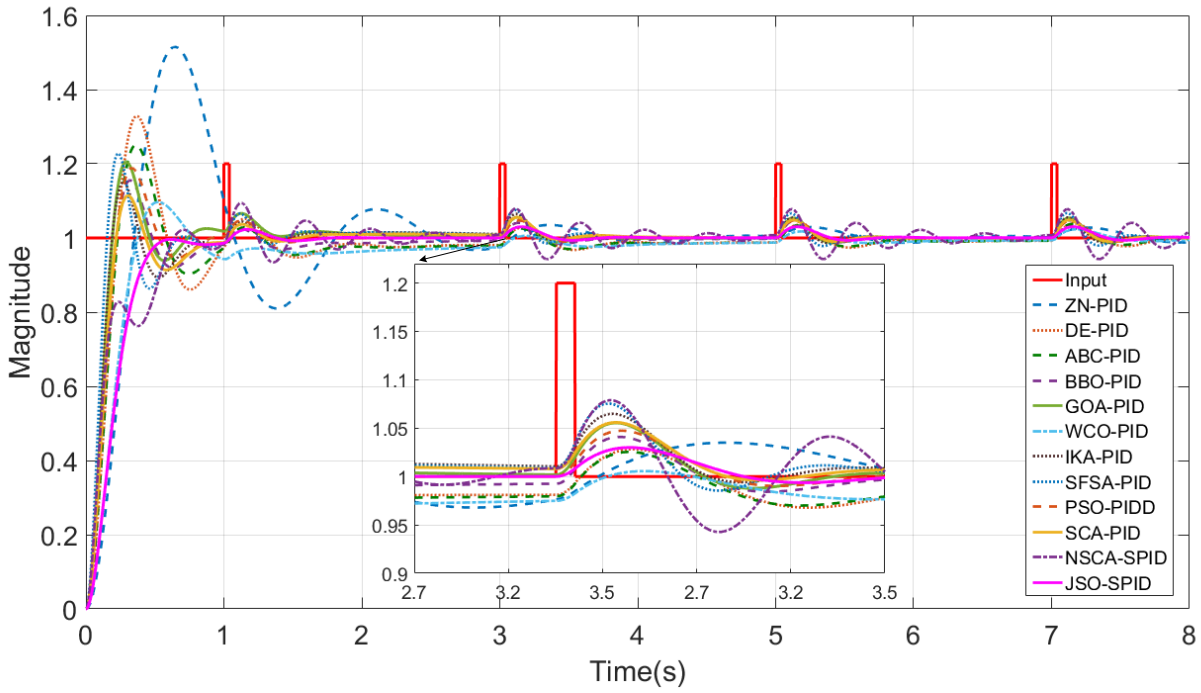

Figure 6.21 Comparison of disturbance response of SPID/PID controllers

proposed controller responded minimally for the sudden disturbances. When compared with the existing NSCA-SPID controller (shown in figure 6.22) the JSO-SPID contrller produced less oscillations. The corresponding controller responses are shown in the figure 6.23. The analysis of the response indicates that the proposed JSO-SPID controller responded optimally than the NSCA-SPID controller.

A comparative study between NSCA-SPID and the proposed JSCA-SPID controllers for IAE, ISE, ITAE, and ITSE objective functions was carried out. The corresponding results are given in table 6.5 and shown in figure 6.24.

Table 6.5 Comparative study of objective functions for disturbance response

	IAE	ISE	ITAE	ITSE
JSO-SPID	0.299	0.1515	0.4664	0.0572
NCSA-SPID	0.4187	0.1095	1.222	0.0822

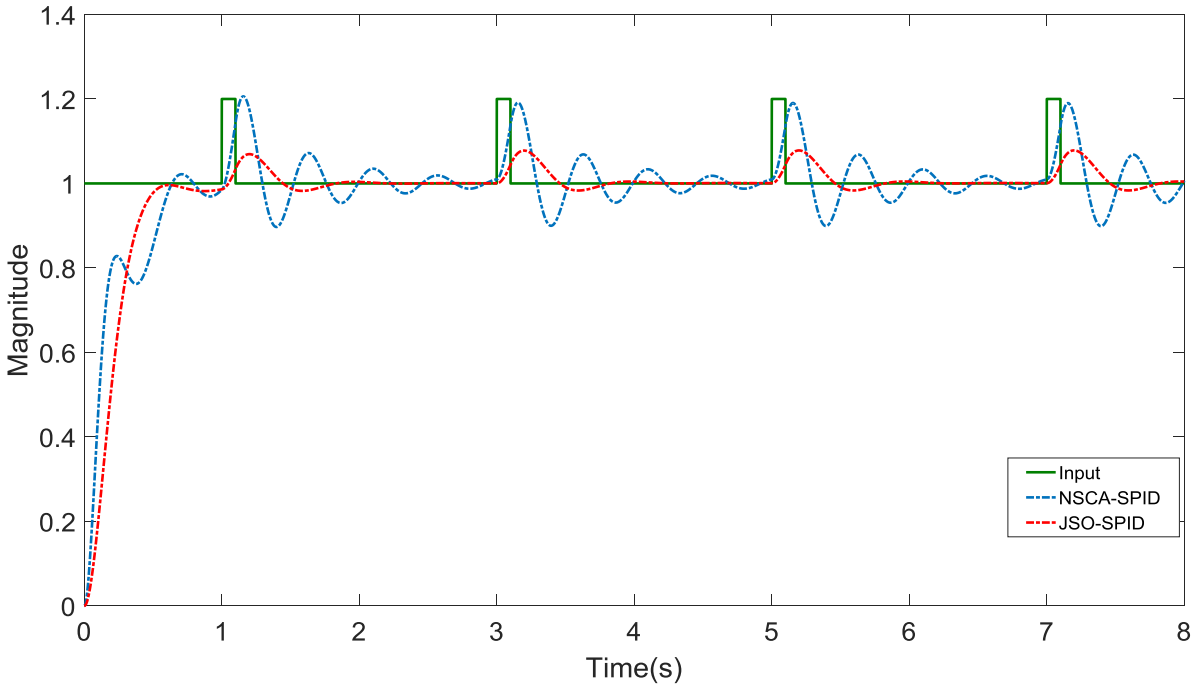

Figure 6.22 Comparison of disturbance response of SPID/PID controllers

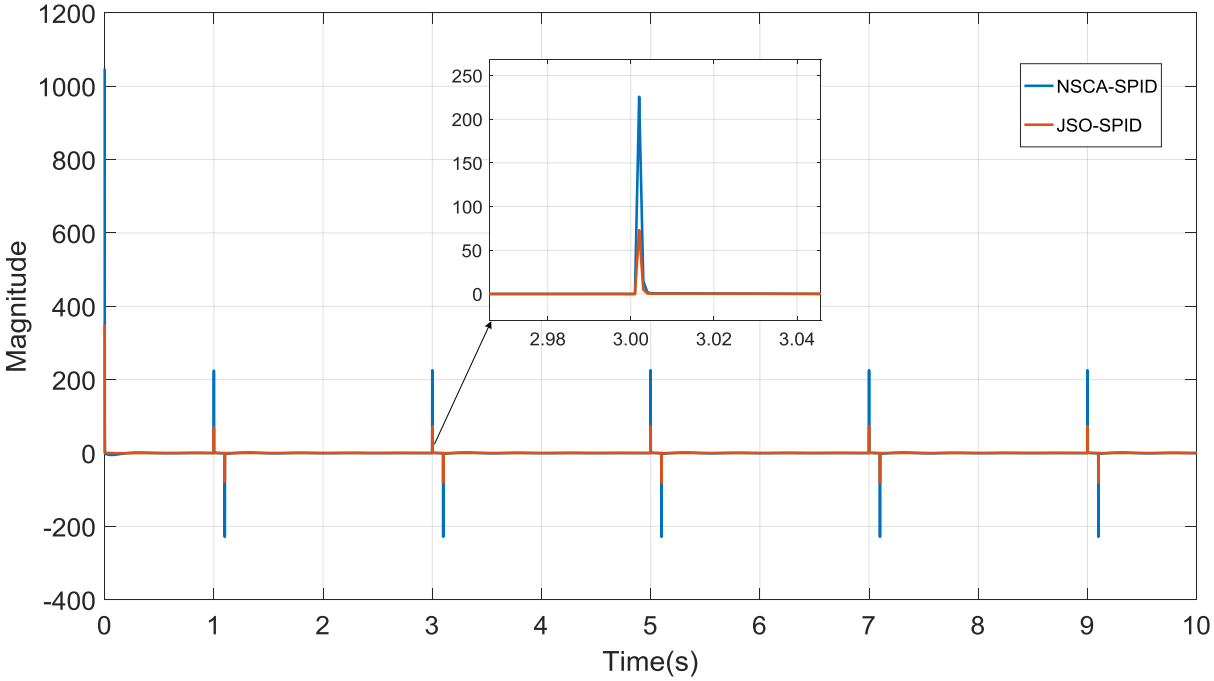

Figure 6.23 Control signal for disturbance response of JSO-SPID and NSCA-SPID controllers

6.4.6 Sawtooth response

The tracking behavior of the proposed controller is studied using a saw-tooth signal as the reference input. The sawtooth response of various SPID/PID controllers along with

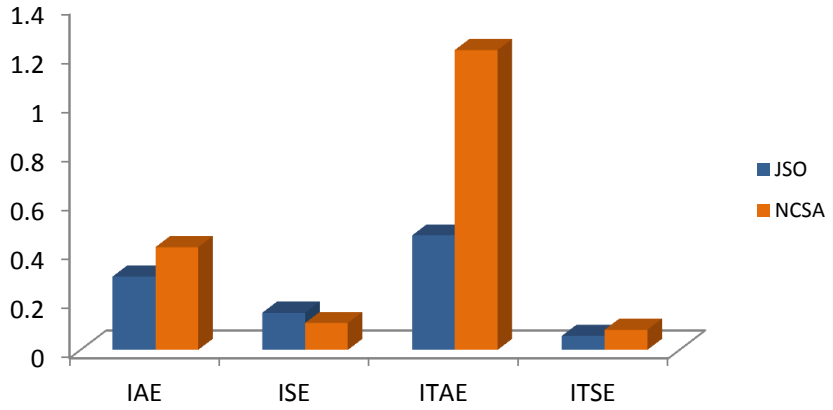

Figure 6.24 Comparison of objective functions for disturbance response

JSO-SPID controlelr is given in the figure 6.25. From the figure, we can clearly observe the effect of rise time and overshoot on the overall tracking error. Since the SPID controllers have moderately good rise time values the initial deviation is more. Since these controllers produce very low overshoot, they are able to settle to the reference tracking signal quickly than the PID controller. The figure also indicates that the JSO-SPID controller settled quickly than the other controllers.

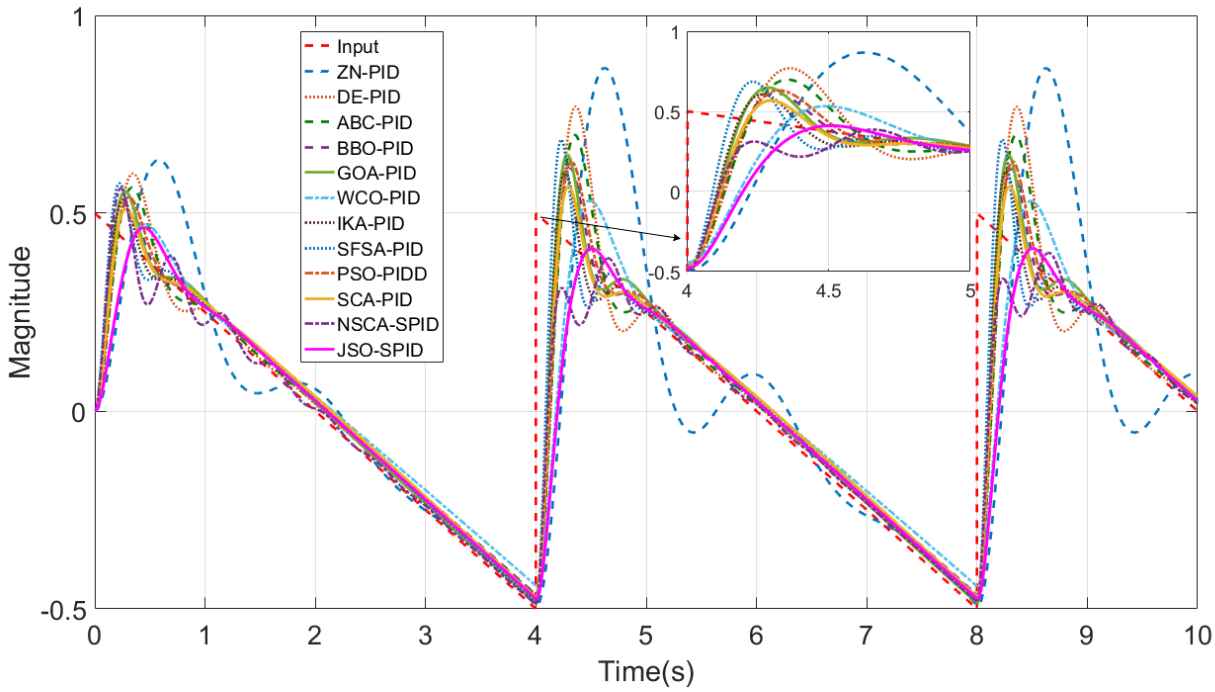

Figure 6.25 Comparison of sawtooth response of SPID/PID controllers

The comparative analysis of different objective function values for the sawtooth

response is given in table 6.6 and visualized in figure 6.26.

Table 6.6 Comparative study of objective functions for sawtooth response

	IAE	ISE	ITAE	ITSE
JSO-SPID	0.530613	0.19044	2.688483	1.03001
NCSA-SPID	0.686863	0.296236	3.399753	1.611317

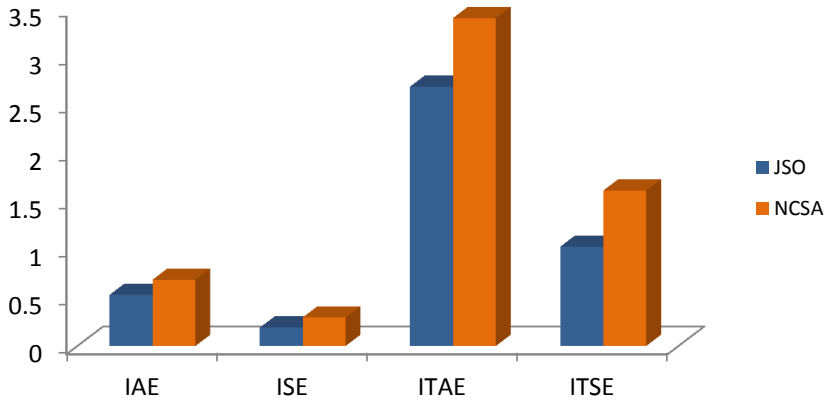

Figure 6.26 Comparison of objective functions for sawtooth response

6.4.7 Frequency response

The frequency response and stability of the proposed JSO-SPID controller is analyzed with the help of bode plot and pole-zero map. The proposed controller's magnitude plot and phase plot are given in figure 6.27. The gain margin of the overall system is 42.7dB where as the phase margin is 70.7 rad and delay margin is 0.109s. During the controller operation the system is stable and this can also be verified by looking at the pole locations. The corresponding pole zero map is shown in the figure 6.28.

The pole locations are given by $-4.079 + 8.869j$, $-4.079 - 8.869j$, $-101.2 + 0.0j$, $-3.534 + 0.0j$, and $-0.6551 + 0.0j$. From the pole locations it is identified that all poles are on left half of S-plane, which indicates that the overall system is stable.

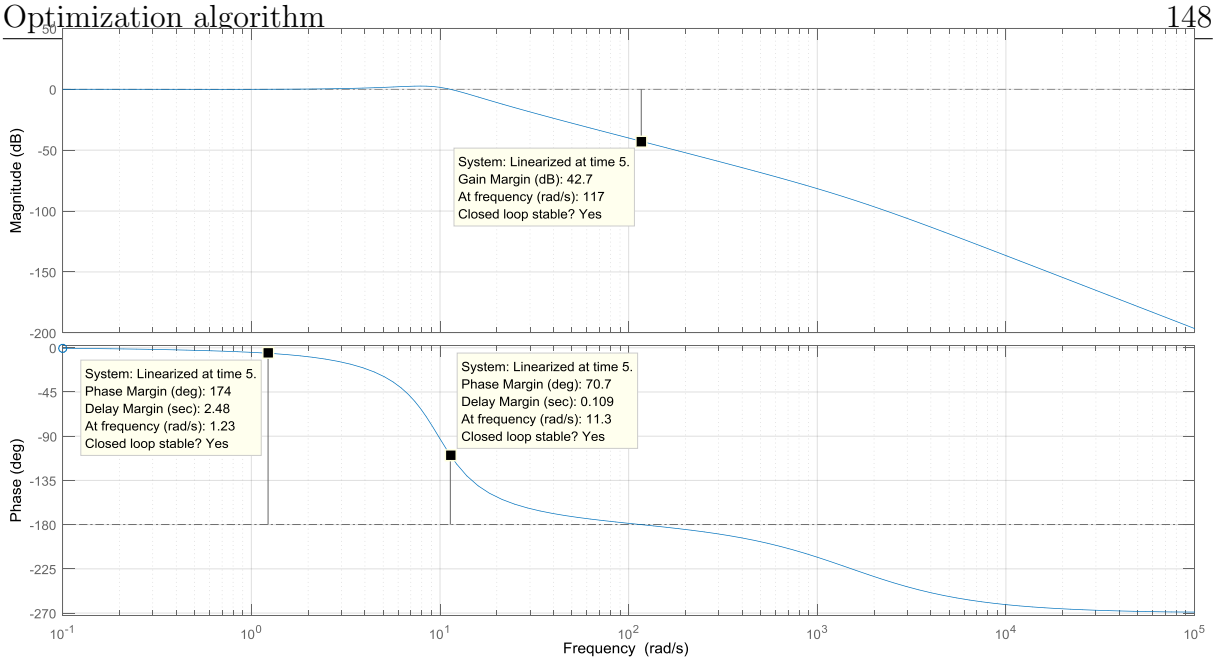

Figure 6.27 Bode plot of JSO-SPID controller

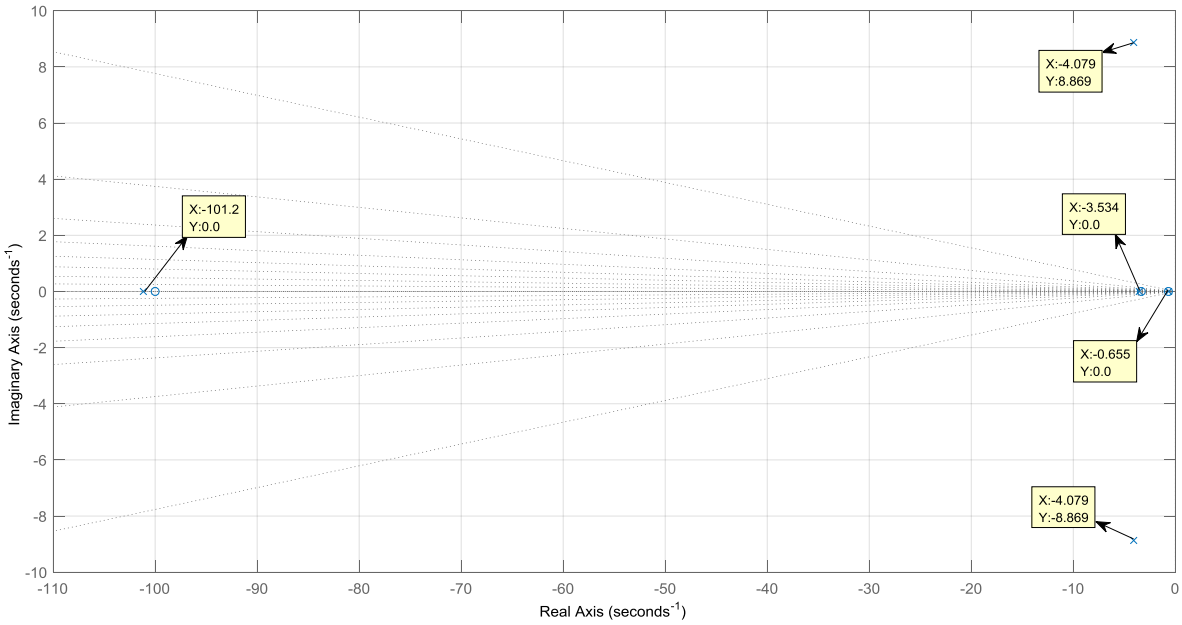

Figure 6.28 Pole zero map of JSO-SPID controller

6.4.8 Summary

A new type of controller called sigmoid PID controller was developed to enhance the AVR system response. The existing NSCA-SPID controller has shown unnecessary oscillations in the steady state response of AVR system. To minimize these oscillations JSO optimization algorithm was used in the present method. On comparison with the existing PID controllers the proposed JSO-SPID controller improved the performance

metrics such as settling time, overshoot, and peak error. When compared with the existing NSCA-SPID controller the proposed controller has shown better performance in terms of rise time, overshoot, and steady state error.

6.5 Conclusion

A novel tuning method to identify the optimum parameters of sigmoid PID controller for AVR system is presented. The SPID controller differs from traditional PID controller in such a way that it continuously varies its gains according to the error value of the system. This makes the controller more dynamic and supports wide range of operation. To optimize the controller gains jellyfish search optimization (JSO) algorithm is used. The simulation results showed that the proposed JSO-SPID controller is able to optimize the AVR system response than the existing state of the art methods. When compared with NSCA-SPID controller, the proposed controller produced much smoother response for step and sudden disturbances. Moreover, it is also found from the simulation study that the SPID controllers naturally produce much lower overshoot values than the existing PID controller. The robust analysis results indicates that the JSO-SPID controller is robust to uncertain variations in the plant parameters. The bode and pole-zero analysis showed that the JSO-SPID controller is stable and produced good gain margin and phase margin values.

Chapter 7

Conclusions and Future Scope

This chapter concludes the thesis by underlining the main contributions. It also presents the possible directions of future work.

7.1 Conclusions

In the thesis various controllers and tuning techniques are developed to control the response of DC motor and automatic voltage regulator (AVR) system. To optimize the system response meta-heuristic algorithms and neural networks are used.

NARXnet based system identification and controller design was presented using data-driven approach for controlling the DC motor speed. The training data for plant network was generated from the mathematical model of the plant and for the controller network, data was generated from HHO tuned FOPID controller using Simulink. The proposed design method for controller was compared with traditional optimization based approaches. The results indicate that the NARXnet controller shows superior performance over the existing controllers in tracking the set points. In addition, the NARXnets can capture the dynamics of FOPID controller; the developed method can also be used for approximation of FOPID/PID controllers. Since the neural networks perform better under noisy environment, the proposed idea can be extended for system identification under environmental noise and interference effects.

A novel architecture based on neural networks for realization of fractional order

PID controllers (FOPID) has been presented. The architecture is developed from the standard discretized fractional order controller equation. For the discretization purpose Al-aloui operator is used as it combines the advantages of Tustin and Euler methods. To optimize the controller parameters a novel optimization algorithm Chaotic Political Optimizer (CHPO) is developed by hybridizing chaotic maps with the Political optimizer algorithm. To analyze the performance of proposed single neuron FOPID (SNFOPID) controller, a DC motor has been considered as plant and the proposed controller is subjected to optimize the DC motor response. Moreover, a new objective function is defined with the combination ITAE and system performance measures to optimize the overall system response.

A novel optimization algorithm has been proposed by hybridizing chaotic maps with the black widow optimization (BWO) algorithm. The effects of 10 chaotic maps have been studied and found that logistic map generates best results for most of the uni-modal and multi-modal benchmark functions. The proposed Chaotic BWO (ChBWO) algorithm has been used in the optimization of FOPID controller parameters for AVR system. To identify the finest controller parameters, a new and efficient objective criterion with a combination of ITAE and ZLG functions are developed.

For the modeling and control of dynamic systems, a neural network-based solution employing BR-NARXnets is presented. To verify the proposed methodology AVR system is considered and its response is optimized using NARXnets.

A novel tuning method to identify the optimum parameters of sigmoid PID controller for AVR system is presented. The SPID controller differs from traditional PID controller in such a way that it continuously varies its gains according to the error value of the system. This makes the controller more dynamic and supports wide range of operation. To optimize the controller gains jellyfish search optimization (JSO) algorithm is used. The simulation results showed that the proposed JSO-SPID controller is able to optimize the AVR system response better than the existing state of the art methods.

7.2 Future Scope

During the thesis work, issues were identified related to implementation of meta-heuristic algorithms and FOPID controller on hardware platforms. These issues form the basis for future scope. The detailed summary is given as follows.

The methods developed in the thesis are limited to offline tuning of FOPID/PID controllers. Therefore for more flexibility and reliability of the controller the techniques can be extended to online tuning. As meta-heuristic algorithms require good amount of computing power new methods can be introduced to reduce the number computations in the proposed algorithms. In the thesis, hardware implementation of FOPID controller is performed using Dspace tools which are expensive. To lower the cost of development, there is scope to create FOPID controller designs that can be used with basic micro controller platforms. The proposed neural network based architectures can be implemented on hardware platforms such as Nvidia boards and raspberry pi etc. Moreover, in the thesis only NARXnets are used to design optimal controllers for various systems. Various other types of recurrent neural networks such as LSTM (Long Short Term Memory) and GRU (Gated Recurrent Units) can also be used for the controller development.

Appendix A

Chaotic maps and benchmark functions

A.1 Chaotic maps used for the study

Table A.1 Various types of chaotic maps

S.No	Chaotic map	Equation
1	Chebyshev map	$x_{i+1} = cos(icos^{-1}(x_i))$
2	Circle map	$x_{i+1} = mod(x_i + b - (\frac{a}{2\pi})sin(2\pi x_k), 1),$ $= 0.5\, and\, b = 0.2$
3	Gauss/mouse map	$x_{i+1} = \begin{cases} 1, & x_i = 0 \\ \frac{1}{mod(x_i,1)}, & otherwise \end{cases}$
4	Iterative map	$x_{i+1} = sin(\frac{a\pi}{x_i}), a = 0.7$
5	Logistic map	$x_{i+1} = ax_i(1 - x_i), a = 4$
6	Piecewise map	$x_{i+1} = \begin{cases} \frac{x}{P} & 0 \le x_i < P \\ \frac{x_i - P}{0.5 - P} & P \le x_i < 0 \\ \frac{1 - P - x_i}{0.5 - P} & 0.5 \le x_i < 1 - P \\ \frac{1 - x_i}{P} & 1 - P \le x_i < 1 \end{cases}, P = 0.4$
7	Sine map	$x_{i+1} = \frac{a}{4}sin(\pi x_i), a = 4$
8	Singer map	$x_{i+1} = \mu(7.86x_i - 23.31x_i^2 + 28.75x_i^3 - 13.302875x_i^4),$ $\mu = 0.7$
9	Sinusoidal map	$x_{i+1} = ax_i^2 sin(\pi x_i), a = 2.3$
10	Tent map	$x_{i+1} = \begin{cases} \frac{x_i}{0.7} & x_i < 0.7 \\ \frac{10}{3}(1 - x_i) & x_i \ge 0.7 \end{cases}$

A.2 Uni-modal and Multi-modal benchmark functions

Table A.2 Definitions of uni-modal and multi-modal benchmark functions

Function	Equation	Range
	uni-modal functions	
Quartic noise	$f_1(x) = \sum_{i=1}^{n} ix_i^4 + \text{random}[0,1)$	[-1.28,1,28]
Powell Sum	$f_2(x) = \sum_{i=1}^{n} \lvert x_i\rvert^{i+1}$	[-1,1]
Schwefel's 2.20	$f_3(x) = \sum_{i=1}^{n} \lvert x_i\rvert$	[-100,100]
Schwefel's 2.21	$f_4(x) = max_{i=1,\ldots,n}\lvert x_i\rvert$	[-100,100]
Zakharov	$f_6(x) = \sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} 0.5ix_i)^2 + (\sum_{i=1}^{n} 0.5ix_i)^4$	[-5,10]
perm 0,D beta	$f_7(x) = \sum_{i=1}^{d}[\sum_{j=1}^{d}(j + \beta(x_j^i - \frac{1}{j^i}))]$	[-50,50]
Three-Hump Camel	$f_8(x,y) = 2x^2 - 1.05x^4 + \frac{x^6}{6} + xy + y^2$	[-5,5]
Schwefel's 1.20	$f_9(x) = \sum_{i=1}^{n}(\sum_{j=1}^{n} x_i)$	[-100,100]
Schwefel's 2.22	$f_{10}(x) = \sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	[-100,100]
	multi-modal functions	
Rastrigin	$f_{11}(x,y) = 10n + \sum_{i=1}^{n}(x_i^2 - 10cos(2\pi x_i))$	[-5.12,5.12]
Periodic	$f_{12}(x) = 1 + \sum_{i=1}^{n} sin^2(x_i) - 0.1e^{(\sum_{i=1}^{n} x_i^2)}$	[-10,10]
Xin-She Yang	$f_{13}(x) = \sum_{i=1}^{n} \epsilon_i \lvert x_i\rvert^i$	[-10,10]
Griewank	$f_{14}(x) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}})$	[-100,100]
Genaralized Penalized	$f_{15}(x) = \frac{\pi}{n}(10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] +$ $(y_n - 1)^2) + \sum_{i=1}^{n} u(x_i, a, k, m)$	[-50,50]
Goldstein Price	$f_{16}(x,y) = [1 + (x + y + 1)^2(19 - 14x + 3x^2 - 14y + 6xy + 3y^2)]$ $[30 + (2x - 3y)^2(18 - 32x + 12x^2 + 4y - 36xy + 27y^2)]$	[-2,2]
Schwefel's 2.26	$f_{17}(x) = 418.9829n - \sum_{i=1}^{n} -x_i(sin\sqrt{\lvert x_i\rvert})$	[-500,500]

Publications

Publications

International Journals

1. Vijaya Kumar Munagala and Ravi Kumar Jatoth , Improved PI?D? controller for AVR system using Chaotic Black Widow Optimization (ChBWO)Algorithm, Computers and Electrical Engineering (Elsevier), 97 (2022) 107600, SCIE indexed, IF: 4.152.

2. Munagala, V.K., Jatoth, R.K. A novel approach for controlling DC motor speed using NARXnet based FOPID controller, Evolving Systems (April, 2022) Springer, vol.13, issue .2, SCIE indexed, IF: 2.347.

3. Vijaya Kumar Munagala and Ravi Kumar Jatoth, Optimization of a novel Single Neuron FOPID(SNFOPID) controller using Chaotic Political Optimizer algorithm with application to DC motor speed control, Engineering Applications of Artificial intelligence (Elsevier) (Under review), SCI indexed.

4. Vijaya Kumar Munagala and Ravi Kumar Jatoth, BR-NARXnets for Identification and Control of Automatic Voltage Regulator System, Soft Computing (Springer) (Under review), SCI indexed.

5. M. Vijaya Kumar, J. Ravi Kumar, Jailsingh Bhookya, A novel sigmoid PID (SPID) Controller for AVR System using Jellyfish Search Optimization Algorithm, Control Engineering Practice, (Elsevier) (Communicated), SCI indexed.

International Conferences

1. Vijaya Kumar Munagala, Ravi Kumar Jatoth, ?Design of Fractional Order PID/PID Controller for Speed Control of DC Motor using Harris Hawks Optimization?, International Conference on Advances in Systems, Control and Computing (AISCC-2020), MNIT Jaipur, 27-28 February, 2020., Scopus

2. Vijaya Kumar Munagala and Ravi Kumar Jatoth, Optimal Design of Fractional Order PID Controller for AVR System using Black Widow Optimization (BWO) Algorithm. In proceedings of Workshop on Machine Learning, Deep learning and Computational intelligence for Wireless Communication (MDCWC2020), National Institute of Technology Tiruchirappalli.22-24 October, 2020, Scopus

3. Vijaya Kumar Munagala and Ravi Kumar Jatoth, Efficient Tuning of FOPID Controller Using Jellyfish Search Optimization(JSO) Algorithm for DC Motor Speed Control, IEEE International Symposium on Smart Electronic Systems (iSES), National Institute of Technology Warangal, 19-21 December, 2022., IEEE Explore

Bibliography

[1] J. Agarwal, G. Parmar, R. Gupta, and A. Sikander, "Analysis of grey wolf optimizer based fractional order pid controller in speed control of dc motor," *Microsystem Technologies*, vol. 24, no. 12, pp. 4997–5006, 2018.

[2] B. Hekimoğlu, "Optimal tuning of fractional order pid controller for dc motor speed control via chaotic atom search optimization algorithm," *IEEE Access*, vol. 7, pp. 38 100–38 114, 2019.

[3] A. T. Azar and S. Vaidyanathan, *Handbook of Research on Advanced Intelligent Control Engineering and Automation*. Engineering Science Reference, 2015.

[4] R. D. RC and R. B. RH, "Modern control systems," 2005.

[5] J. J. d'Azzo and C. D. Houpis, *Linear control system analysis and design: conventional and modern*. McGraw-Hill Higher Education, 1995.

[6] C. A. Monje, Y. Chen, B. M. Vinagre, D. Xue, and V. Feliu-Batlle, *Fractional-order systems and controls: fundamentals and applications*. Springer Science & Business Media, 2010.

[7] K. J. Åström and T. J. McAvoy, "Intelligent control," *Journal of Process control*, vol. 2, no. 3, pp. 115–127, 1992.

[8] M. Ge, M.-S. Chiu, and Q.-G. Wang, "Robust pid controller design via lmi approach," *Journal of process control*, vol. 12, no. 1, pp. 3–13, 2002.

[9] D. E. Rivera, M. Morari, and S. Skogestad, "Internal model control: Pid controller design," *Industrial & engineering chemistry process design and development*, vol. 25, no. 1, pp. 252–265, 1986.

[10] M.-T. Ho and C.-Y. Lin, "Pid controller design for robust performance," *IEEE Transactions on Automatic Control*, vol. 48, no. 8, pp. 1404–1409, 2003.

[11] K. H. Ang, G. Chong, and Y. Li, "Pid control system analysis, design, and technology," *IEEE transactions on control systems technology*, vol. 13, no. 4, pp. 559–576, 2005.

[12] X.-S. Yang, "Metaheuristic optimization," *Scholarpedia*, vol. 6, no. 8, p. 11472, 2011.

[13] J. A. Parejo, A. Ruiz-Cortés, S. Lozano, and P. Fernandez, "Metaheuristic optimization frameworks: a survey and benchmarking," *Soft Computing*, vol. 16, no. 3, pp. 527–561, 2012.

[14] X.-S. Zhang, *Neural networks in optimization*. Springer Science & Business Media, 2013, vol. 46.

[15] I. Podlubny, "Fractional-order systems and pi/sup/spl lambda//d/sup/spl mu//-controllers," *IEEE Transactions on automatic control*, vol. 44, no. 1, pp. 208–214, 1999.

[16] P. Shah and S. Agashe, "Review of fractional pid controller," *Mechatronics*, vol. 38, pp. 29–41, 2016.

[17] M. Nakagawa and K. Sorimachi, "Basic characteristics of a fractance device," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 75, no. 12, pp. 1814–1819, 1992.

[18] S. Westerlund and L. Ekstam, "Capacitor theory," *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 1, no. 5, pp. 826–839, 1994.

[19] J. Wang, "Realizations of generalized warburg impedance with rc ladder networks and transmission lines," *Journal of the Electrochemical Society*, vol. 134, no. 8, p. 1915, 1987.

[20] K. S. Miller and B. Ross, *An introduction to the fractional calculus and fractional differential equations*. Wiley, 1993.

[21] M. D. Ortigueira, *Fractional calculus for scientists and engineers.* Springer Science & Business Media, 2011, vol. 84.

[22] P. Igor, "Fractional differential equations. mathematics in science and engineering," 1999.

[23] B. J. Lurie, "Three-parameter tunable tilt-integral-derivative (tid) controller," *US-PATENT-5,371,670*, 1994.

[24] A. Oustaloup, *La commande CRONE: commande robuste d'ordre non entier.* Hermes, 1991.

[25] A. Oustaloup, F. Levron, B. Mathieu, and F. M. Nanot, "Frequency-band complex noninteger differentiator: characterization and synthesis," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 1, pp. 25–39, 2000.

[26] A. Oustaloup, P. Melchior, P. Lanusse, O. Cois, and F. Dancla, "The crone toolbox for matlab," in *CACSD. Conference Proceedings. IEEE International Symposium on Computer-Aided Control System Design (Cat. No. 00TH8537).* IEEE, 2000, pp. 190–195.

[27] I. Podlubny, "Fractional-order systems and pi/sup/spl lambda//d/sup/spl mu//-controllers," *IEEE Transactions on automatic control*, vol. 44, no. 1, pp. 208–214, 1999.

[28] B. M. Vinagre, I. Podlubny, L. Dorcak, and V. Feliu, "On fractional pid controllers: a frequency domain approach," *IFAC Proceedings Volumes*, vol. 33, no. 4, pp. 51–56, 2000.

[29] H.-F. Raynaud and A. Zergainoh, "State-space representation for fractional order controllers," *Automatica*, vol. 36, no. 7, pp. 1017–1021, 2000.

[30] C. A. Monje, B. M. Vinagre, A. J. Calderon, V. Feliu, and Y. Chen, "Auto-tuning of fractional lead-lag compensators," *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 319–324, 2005.

[31] D. Valerio and J. S. da Costa, "A review of tuning methods for fractional pids," in *4th IFAC Workshop on Fractional Differentiation and its Applications, FDA*, vol. 10, no. 5, 2010.

[32] P. Wang and D. Kwok, "Optimal design of pid process controllers based on genetic algorithms," *Control Engineering Practice*, vol. 2, no. 4, pp. 641–648, 1994.

[33] Y. Mitsukura, T. Yamamoto, and M. Kaneda, "A genetic tuning algorithm of pid parameters," in *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, vol. 1. IEEE, 1997, pp. 923–928.

[34] C.-H. Lee and F.-K. Chang, "Fractional-order pid controller optimization via improved electromagnetism-like algorithm," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8871–8878, 2010.

[35] L. Dorcak, J. Terpak, M. Papajova, F. Dorcakova, and L. Pivka, "Design of the fractional-order piλdμ controllers based on the optimization with self-organizing migrating algorithm," *Acta Montanistica Slovaca*, vol. 12, no. 4, pp. 285–293, 2007.

[36] D. Maiti, S. Biswas, and A. Konar, "Design of a fractional order pid controller using particle swarm optimization technique," *arXiv preprint arXiv:0810.3776*, 2008.

[37] A. A. Aldair and W. J. Wang, "Design of fractional order controller based on evolutionary algorithm for a full vehicle nonlinear active suspension systems," *International Journal of Control and Automation*, vol. 3, no. 4, pp. 33–46, 2010.

[38] Y. Zhang and J. Li, "Fractional-order pid controller tuning based on genetic algorithm," in *2011 International Conference on Business Management and Electronic Information*, vol. 3. IEEE, 2011, pp. 764–767.

[39] A. Rajasekhar, V. Chaitanya, and S. Das, "Fractional-order pi λ d μ controller design using a modified artificial bee colony algorithm," in *International Conference on Swarm, Evolutionary, and Memetic Computing*. Springer, 2011, pp. 670–678.

[40] W. Sheng and Y. Bao, "Fruit fly optimization algorithm based fractional order fuzzy-pid controller for electronic throttle," *Nonlinear Dynamics*, vol. 73, no. 1, pp. 611–619, 2013.

[41] S. A. Taher, M. H. Fini, and S. F. Aliabadi, "Fractional order pid controller design for lfc in electric power systems using imperialist competitive algorithm," *Ain Shams Engineering Journal*, vol. 5, no. 1, pp. 121–135, 2014.

[42] Y. Mousavi and A. Alfi, "A memetic algorithm applied to trajectory control by tuning of fractional order proportional-integral-derivative controllers," *Applied Soft Computing*, vol. 36, pp. 599–617, 2015.

[43] X. Liu, "Optimization design on fractional order pid controller based on adaptive particle swarm optimization algorithm," *Nonlinear Dynamics*, vol. 84, no. 1, pp. 379–386, 2016.

[44] A. Zamani, S. M. Barakati, and S. Yousofi-Darmian, "Design of a fractional order pid controller using gbmo algorithm for load–frequency control with governor saturation consideration," *ISA transactions*, vol. 64, pp. 56–66, 2016.

[45] C. Li, N. Zhang, X. Lai, J. Zhou, and Y. Xu, "Design of a fractional-order pid controller for a pumped storage unit using a gravitational search algorithm based on the cauchy and gaussian mutation," *Information Sciences*, vol. 396, pp. 162–181, 2017.

[46] B. Bourouba, S. Ladaci, and A. Chaabi, "Moth-flame optimisation algorithm-based fractional order pi λ d μ controller with mrac tuning configuration," *International Journal of Systems, Control and Communications*, vol. 9, no. 2, pp. 148–171, 2018.

[47] R. Mohammadi Asl, E. Pourabdollah, and M. Salmani, "Optimal fractional order pid for a robotic manipulator using colliding bodies design," *Soft Computing*, vol. 22, no. 14, pp. 4647–4659, 2018.

[48] A. T. El-Deen, A. H. Mahmoud, and A. R. El-Sawi, "Optimal pid tuning for dc motor speed controller based on genetic algorithm," *Int. Rev. Autom. Control*, vol. 8, no. 1, pp. 80–85, 2015.

[49] A. Madadi and M. M. Motlagh, "Optimal control of dc motor using grey wolf optimizer algorithm," *Technical Journal of Engineering and Applied Science*, vol. 4, no. 4, pp. 373–379, 2014.

[50] U. Bhatnagar, "Application of grey wolf optimization in optimal control of dc motor and robustness analysis," *SKIT Research Journal*, 2018.

[51] M. Khalilpour, N. Razmjooy, H. Hosseini, and P. Moallem, "Optimal control of dc motor using invasive weed optimization (iwo) algorithm," in *Majlesi Conference on Electrical Engineering, Majlesi New Town, Isfahan, Iran*, 2011.

[52] I. Khanam and G. Parmar, "Application of sfs algorithm in control of dc motor and comparative analysis," in *2017 4th IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics (UPCON)*. IEEE, 2017, pp. 256–261.

[53] A. K. Mishra, V. K. Tiwari, R. Kumar, and T. Verma, "Speed control of dc motor using artificial bee colony optimization technique," in *2013 International conference on control, automation, robotics and embedded systems (CARE)*. IEEE, 2013, pp. 1–6.

[54] R. K. Achanta and V. K. Pamula, "Dc motor speed control using pid controller tuned by jaya optimization algorithm," in *2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*. IEEE, 2017, pp. 983–987.

[55] B. Hekimoğlu, S. Ekinci, V. Demiray, R. Doguruci, and A. Yıldırım, "Speed control of dc motor using pid controller tuned by salp swarm algorithm," *Proc. IENSC*, pp. 1878–1889, 2018.

[56] R. V. Jain, M. Aware, and A. Junghare, "Tuning of fractional order pid controller using particle swarm optimization technique for dc motor speed control," in *2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*. IEEE, 2016, pp. 1–4.

[57] S. Ekinci, D. Izci, and B. Hekimoğlu, "Optimal fopid speed control of dc motor via opposition-based hybrid manta ray foraging optimization and simulated annealing algorithm," *Arabian Journal for Science and Engineering*, vol. 46, no. 2, pp. 1395–1409, 2021.

[58] S. Ekinci, B. Hekimoğlu, and D. Izci, "Opposition based henry gas solubility optimization as a novel algorithm for pid control of dc motor," *Engineering Science and Technology, an International Journal*, vol. 24, no. 2, pp. 331–342, 2021.

[59] S. Ekinci, D. Izci, and B. Hekimoğlu, "Pid speed control of dc motor using harris hawks optimization algorithm," in *2020 International conference on electrical, communication, and computer engineering (ICECCE)*. IEEE, 2020, pp. 1–6.

[60] V. K. Munagala and R. K. Jatoth, "Design of fractional-order pid/pid controller for speed control of dc motor using harris hawks optimization," in *Intelligent algorithms for analysis and control of dynamical systems*. Springer, 2021, pp. 103–113.

[61] E. Çelik and N. Öztürk, "First application of symbiotic organisms search algorithm to off-line optimization of pi parameters for dsp-based dc motor drives," *Neural Computing and Applications*, vol. 30, no. 5, pp. 1689–1699, 2018.

[62] E. Çelik and H. Gör, "Enhanced speed control of a dc servo system using pi+ df controller tuned by stochastic fractal search technique," *Journal of the Franklin Institute*, vol. 356, no. 3, pp. 1333–1359, 2019.

[63] M. Zamani, M. Karimi-Ghartemani, N. Sadati, and M. Parniani, "Design of a fractional order pid controller for an avr using particle swarm optimization," *Control Engineering Practice*, vol. 17, no. 12, pp. 1380–1387, 2009.

[64] M. E. Ortiz-Quisbert, M. A. Duarte-Mermoud, F. Milla, R. Castro-Linares, and G. Lefranc, "Optimal fractional order adaptive controllers for avr applications," *Electrical Engineering*, vol. 100, no. 1, pp. 267–283, 2018.

[65] I. Pan and S. Das, "Chaotic multi-objective optimization based design of fractional order piλdμ controller in avr system," *International Journal of Electrical Power & Energy Systems*, vol. 43, no. 1, pp. 393–407, 2012.

[66] I. A. Khan, A. S. Alghamdi, T. A. Jumani, A. Alamgir, A. B. Awan, and A. Khidrani, "Salp swarm optimization algorithm-based fractional order pid controller for dynamic response and stability enhancement of an automatic voltage regulator system," *Electronics*, vol. 8, no. 12, p. 1472, 2019.

[67] G.-Q. Zeng, J. Chen, Y.-X. Dai, L.-M. Li, C.-W. Zheng, and M.-R. Chen, "Design of fractional order pid controller for automatic regulator voltage system based on multi-objective extremal optimization," *Neurocomputing*, vol. 160, pp. 173–184, 2015.

[68] Y. Tang, M. Cui, C. Hua, L. Li, and Y. Yang, "Optimum design of fractional order piλdμ controller for avr system using chaotic ant swarm," *Expert Systems with Applications*, vol. 39, no. 8, pp. 6887–6896, 2012.

[69] Z. Bingul and O. Karahan, "A novel performance criterion approach to optimum design of pid controller using cuckoo search algorithm for avr system," *Journal of the Franklin Institute*, vol. 355, no. 13, pp. 5534–5559, 2018.

[70] D.-L. Zhang, T. Ying-Gan, and G. Xin-Ping, "Optimum design of fractional order pid controller for an avr system using an improved artificial bee colony algorithm," *Acta Automatica Sinica*, vol. 40, no. 5, pp. 973–979, 2014.

[71] A. M. Mosaad, M. A. Attia, and A. Y. Abdelaziz, "Comparative performance analysis of avr controllers using modern optimization techniques," *Electric Power Components and Systems*, vol. 46, no. 19-20, pp. 2117–2130, 2018.

[72] J. Bhookya and R. K. Jatoth, "Optimal fopid/pid controller parameters tuning for the avr system based on sine–cosine-algorithm," *Evolutionary Intelligence*, vol. 12, no. 4, pp. 725–733, 2019.

[73] M. Micev, M. Ćalasan, and D. Oliva, "Fractional order pid controller design for an avr system using chaotic yellow saddle goatfish algorithm," *Mathematics*, vol. 8, no. 7, p. 1182, 2020.

[74] S. Gao, Y. Yu, Y. Wang, J. Wang, J. Cheng, and M. Zhou, "Chaotic local search-based differential evolution algorithms for optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 6, pp. 3954–3967, 2019.

[75] M. Micev, M. Ćalasan, Z. M. Ali, H. M. Hasanien, and S. H. A. Aleem, "Optimal design of automatic voltage regulation controller using hybrid simulated annealing–manta ray foraging optimization algorithm," *Ain Shams Engineering Journal*, vol. 12, no. 1, pp. 641–657, 2021.

[76] Z.-L. Gaing, "A particle swarm optimization approach for optimum design of pid controller in avr system," *IEEE transactions on energy conversion*, vol. 19, no. 2, pp. 384–391, 2004.

[77] L. dos Santos Coelho, "Tuning of pid controller for an automatic regulator voltage system using chaotic optimization approach," *Chaos, Solitons & Fractals*, vol. 39, no. 4, pp. 1504–1514, 2009.

[78] H. Zhu, L. Li, Y. Zhao, Y. Guo, and Y. Yang, "Cas algorithm-based optimum design of pid controller in avr system," *Chaos, Solitons & Fractals*, vol. 42, no. 2, pp. 792–800, 2009.

[79] A. Chatterjee, V. Mukherjee, and S. Ghoshal, "Velocity relaxed and craziness-based swarm optimized intelligent pid and pss controlled avr system," *International Journal of Electrical Power & Energy Systems*, vol. 31, no. 7-8, pp. 323–333, 2009.

[80] H. Gozde and M. C. Taplamacioglu, "Comparative performance analysis of artificial bee colony algorithm for automatic voltage regulator (avr) system," *Journal of the Franklin Institute*, vol. 348, no. 8, pp. 1927–1946, 2011.

[81] S. Panda, B. K. Sahu, and P. K. Mohanty, "Design and performance analysis of pid controller for an automatic voltage regulator system using simplified particle swarm optimization," *Journal of the Franklin Institute*, vol. 349, no. 8, pp. 2609–2625, 2012.

[82] P. K. Mohanty, B. K. Sahu, and S. Panda, "Tuning and assessment of proportional–integral–derivative controller for an automatic voltage regulator system employing local unimodal sampling algorithm," *Electric Power Components and Systems*, vol. 42, no. 9, pp. 959–969, 2014.

[83] E. Çelik and R. Durgut, "Performance enhancement of automatic voltage regulator by modified cost function and symbiotic organisms search algorithm," *Engineering science and technology, an international journal*, vol. 21, no. 5, pp. 1104–1111, 2018.

[84] R. Lahcene, S. Abdeldjalil, and K. Aissa, "Optimal tuning of fractional order pid controller for avr system using simulated annealing optimization algorithm," in

2017 5th International Conference on Electrical Engineering-Boumerdes (ICEE-B). IEEE, 2017, pp. 1–6.

[85] A. M. Mosaad, M. A. Attia, and A. Y. Abdelaziz, "Whale optimization algorithm to tune pid and pida controllers on avr system," *Ain Shams Engineering Journal*, vol. 10, no. 4, pp. 755–767, 2019.

[86] S. Ekinci and B. Hekimoğlu, "Improved kidney-inspired algorithm approach for tuning of pid controller in avr system," *IEEE Access*, vol. 7, pp. 39 935–39 947, 2019.

[87] A. I. Omar, S. H. A. Aleem, E. E. El-Zahab, M. Algablawy, and Z. M. Ali, "An improved approach for robust control of dynamic voltage restorer and power quality enhancement using grasshopper optimization algorithm," *ISA transactions*, vol. 95, pp. 110–129, 2019.

[88] D. H. Kim, "Hybrid ga?bf based intelligent pid controller tuning for avr system," *Applied Soft Computing*, vol. 11, no. 1, pp. 11–22, 2011.

[89] H. M. Hasanien, "Design optimization of pid controller in automatic voltage regulator system using taguchi combined genetic algorithm method," *IEEE systems journal*, vol. 7, no. 4, pp. 825–831, 2012.

[90] M.-J. Blondin, J. Sanchis, P. Sicard, and J. Herrero, "New optimal controller tuning method for an avr system using a simplified ant colony optimization with a new constrained nelder–mead algorithm," *Applied soft computing*, vol. 62, pp. 216–229, 2018.

[91] Z. Boussaada, O. Curea, A. Remaci, H. Camblong, and N. Mrabet Bellaaj, "A nonlinear autoregressive exogenous (narx) neural network model for the prediction of the daily direct solar radiation," *Energies*, vol. 11, no. 3, p. 620, 2018.

[92] H. T. Siegelmann, B. G. Horne, and C. L. Giles, "Computational capabilities of recurrent narx neural networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 27, no. 2, pp. 208–215, 1997.

[93] J. M. Menezes and G. A. Barreto, "On recurrent neural networks for auto-similar traffic prediction: A performance evaluation," in *2006 International telecommunications symposium*. IEEE, 2006, pp. 534–539.

[94] B. M. Wilamowski, "Neural network architectures and learning algorithms," *IEEE Industrial Electronics Magazine*, vol. 3, no. 4, pp. 56–63, 2009.

[95] Z. Liu, X. Zhuang, and S. Wang, "Speed control of a dc motor using bp neural networks," in *Proceedings of 2003 IEEE Conference on Control Applications, 2003. CCA 2003.*, vol. 2. IEEE, 2003, pp. 832–835.

[96] S. A. Hamoodi, I. I. Sheet, and R. A. Mohammed, "A comparison between pid controller and ann controller for speed control of dc motor," in *2019 2nd International conference on electrical, communication, computer, power and control engineering (ICECCPCE)*. IEEE, 2019, pp. 221–224.

[97] M. Milovanović, D. Antić, M. Spasić, S. S. Nikolić, S. Perić, and M. Milojković, "Improvement of dc motor velocity estimation using a feedforward neural network," *Acta Polytechnica Hungarica*, vol. 12, no. 6, pp. 107–126, 2015.

[98] B. M. Zaineb, A. Aicha, B. H. Mouna, and S. Lassaad, "Speed control of dc motor based on an adaptive feed forward neural imc controller," in *2017 International conference on green energy conversion systems (GECS)*. IEEE, 2017, pp. 1–7.

[99] E. Çelik, H. Gör, N. Öztürk, and E. Kurt, "Application of artificial neural network to estimate power generation and efficiency of a new axial flux permanent magnet synchronous generator," *International Journal of Hydrogen Energy*, vol. 42, no. 28, pp. 17 692–17 699, 2017.

[100] P. Q. Dzung and L. Phuong, "Ann-control system dc motor," *IEEE Transactions*, pp. 82–90, 2002.

[101] H. S. Dakheel, "Speed control of separately exited dc motor using artificial neural network," *Journal of Engineering and Sustainable Development*, vol. 16, no. 4, pp. 349–362, 2012.

[102] W. M. Elsrogy, M. Fkirin, and M. M. Hassan, "Speed control of dc motor using pid controller based on artificial intelligence techniques," in *2013 International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, 2013, pp. 196–201.

[103] A. Bernard, "Speed control of separately excited dc motor using artificial intelligent approach," Ph.D. dissertation, Universiti Tun Hussein Onn Malaysia, 2013.

[104] M. Aamir, "On replacing pid controller with ann controller for dc motor position control," *arXiv preprint arXiv:1312.0148*, 2013.

[105] Y. Naung, S. Anatolii, and Y. H. Lin, "Speed control of dc motor by using neural network parameter tuner for pi-controller," in *2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*. IEEE, 2019, pp. 2152–2156.

[106] A. K. Bhullar, R. Kaur, and S. Sondhi, "Design of fopid controller for optimizing avr system using neural network algorithm," in *2020 IEEE 17th India council international conference (INDICON)*. IEEE, 2020, pp. 1–7.

[107] A. A. Bhutto, F. A. Chachar, M. Hussain, D. K. Bhutto, and S. E. Bakhsh, "Implementation of probabilistic neural network (pnn) based automatic voltage regulator (avr) for excitation control system in matlab," in *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*. IEEE, 2019, pp. 1–5.

[108] M. Elsisi, "Design of neural network predictive controller based on imperialist competitive algorithm for automatic voltage regulator," *Neural Computing and Applications*, vol. 31, no. 9, pp. 5017–5027, 2019.

[109] A. P. Memon, A. S. Memon, A. A. Akhund, and R. H. Memon, "Multilayer perceptrons neural network automatic voltage regulator with applicability and improvement in power system transient stability," *International Journal of Emerging Trends in Electrical and Electronics (IJETEE ISSN: 2320-9569), IRET publication*, vol. 9, no. 1, pp. 30–38, 2013.

[110] A. P. Memon, M. A. Uqaili, and Z. Memon, "Design of ffnn avr for enhancement of power system stability using matlab/simulink," *Mehran University Research Journal of Engineering and Technology*, vol. 31, no. 03, 2012.

[111] M. Tahan, E. Tsoutsanis, M. Muhammad, and Z. A. Karim, "Performance-based health monitoring, diagnostics and prognostics for condition-based maintenance of gas turbines: A review," *Applied energy*, vol. 198, pp. 122–144, 2017.

[112] M. G. De Giorgi, A. Ficarella, and M. Quarta, "Dynamic performance simulation and control of an aeroengine by using narx models," in *MATEC web of conferences*, vol. 304. EDP Sciences, 2019, p. 03005.

[113] A. Ranganathan, "The levenberg-marquardt algorithm," *Tutoral on LM algorithm*, vol. 11, no. 1, pp. 101–110, 2004.

[114] Y. Q. Chen and K. L. Moore, "Discretization schemes for fractional-order differentiators and integrators," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 49, no. 3, pp. 363–367, 2002.

[115] Q. Zhang, B. Song, H. Zhao, J. Zhang *et al.*, "Discretization of fractional order differentiator and integrator with different fractional orders," *Intelligent Control and Automation*, vol. 8, no. 02, p. 75, 2017.

[116] Q. Askari, I. Younas, and M. Saeed, "Political optimizer: A novel socio-inspired meta-heuristic for global optimization," *Knowledge-based systems*, vol. 195, p. 105709, 2020.

[117] B. Alatas, "Chaotic harmony search algorithms," *Applied mathematics and computation*, vol. 216, no. 9, pp. 2687–2699, 2010.

[118] M. Ahmadi and H. Mojallali, "Chaotic invasive weed optimization algorithm with application to parameter estimation of chaotic systems," *Chaos, Solitons & Fractals*, vol. 45, no. 9-10, pp. 1108–1120, 2012.

[119] V. Hayyolalam and A. A. P. Kazem, "Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems," *Engineering Applications of Artificial Intelligence*, vol. 87, p. 103249, 2020.

[120] A. Sikander, P. Thakur, R. C. Bansal, and S. Rajasekar, "A novel technique to design cuckoo search based fopid controller for avr in power systems," *Computers & Electrical Engineering*, vol. 70, pp. 261–274, 2018.

[121] M. Ausloos, *The logistic map and the route to chaos: From the beginnings to modern applications*. Springer Science & Business Media, 2006.

[122] D. J. MacKay, "Bayesian interpolation," *Neural computation*, vol. 4, no. 3, pp. 415–447, 1992.

[123] F. Dan Foresee and M. T. Hagan, "Gauss-newton approximation to bayesian learning," in *International Conference on Neural Networks*, vol. 3, 1997, pp. 1930–1935.

[124] M. T. Hagan, H. B. Demuth, and M. Beale, *Neural network design*. PWS Publishing Co., 1997.

[125] M. A. Sahib and B. S. Ahmed, "A new multiobjective performance criterion used in pid tuning optimization algorithms," *Journal of advanced research*, vol. 7, no. 1, pp. 125–134, 2016.

[126] A. Ateş, B. B. Alagöz, C. Yeroğlu, and H. Alisoy, "Sigmoid based pid controller implementation for rotor control," in *2015 European Control Conference (ECC)*. IEEE, 2015, pp. 458–463.

[127] M. Ahmad and R. R. Ismail, "A data-driven sigmoid-based pi controller for buck-converter powered dc motor," in *2017 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE)*. IEEE, 2017, pp. 81–86.

[128] J.-S. Chou and D.-N. Truong, "A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean," *Applied Mathematics and Computation*, vol. 389, p. 125535, 2021.

[129] J. G. Ziegler, N. B. Nichols *et al.*, "Optimum settings for automatic controllers," *trans. ASME*, vol. 64, no. 11, 1942.

[130] U. GÜVENÇ, A. H. IŞIK, T. YİĞİT, and I. Akkaya, "Performance analysis of biogeography-based optimization for automatic voltage regulator system," *Turkish*

Journal of Electrical Engineering and Computer Sciences, vol. 24, no. 3, pp. 1150–1162, 2016.

[131] N. Razmjooy, M. Khalilpour, and M. Ramezani, "A new meta-heuristic optimization algorithm inspired by fifa world cup competitions: theory and its application in pid designing for avr system," *Journal of Control, Automation and Electrical Systems*, vol. 27, no. 4, pp. 419–440, 2016.

[132] M. A. Sahib, "A novel optimal pid plus second order derivative controller for avr system," *Engineering Science and Technology, an International Journal*, vol. 18, no. 2, pp. 194–206, 2015.

[133] M. Suid and M. Ahmad, "Optimal tuning of sigmoid pid controller using nonlinear sine cosine algorithm for the automatic voltage regulator system," *ISA transactions*, vol. 128, pp. 265–286, 2022.