People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University M'Hamed BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering
Department of Power and Control

Final Year Project Report Presented in Partial Fulfilment of the Requirements for the Degree of

MASTER

In **Power Engineering**

Option: **Power Engineering**

Title:

Optimal Design of PID parameters for Automatic Voltage Regulator using Metaheuristics Algorithms

Presented by:

- **Yasser Aklouchi (Power Engineering)**
- **Amar Hadj Ali (Control Engineering)**

Supervisor:

Prof. A. KHELDOUN

Registration Number ..... /2021

Abstract

Application of classical PID and fractional order PID (FOPID) controller to an automatic voltage regulator (AVR) is presented and studied in this project. Fractional-order PIλDμ controller is a generalization of standard PID controller using fractional calculus. Compared to PID controller, the tuning of FOPID is more complex and remains a challenging problem. This project proposes some population based optimization metaheuristics algorithms called Ant Lion Optimizer algorithm (ALO), Particle Swarm Optimizer algorithm (PSO) and Grey Wolf Optimizer algorithm (GWO) for solving the PID and FOPID controller tuning problem. ALO has proved to have very high efficiency and FOPID provide good control performance with respect to reference input and improve the system robustness with respect to model uncertainties after comparison are made.

Keywords:
Automatic voltage regulator (AVR); Metaheuristic algorithm; Fractional Order PID controller; Classical PID controller

Dedication

To my father, who taught me how to be a man.

To my mother, who taught me how to be kind, humble, and caring.

To my sisters, whom I am immensely proud of.

To Azzeddine, Fathi, Awris, Abdelwaheb, and Ala Eddine; we do not have the same blood pumping through our veins, but you are my brothers.

To those who work hard and are ambitious enough to reach for great heights.

A. YASSER

This is dedicated to my beloved family, especially my parents and my two brothers, Hamid and Djamel. Their unconditional love and constant support drive me forward every single day.

To all my friends who have encouraged me and offered me support during the different stages of my life.

H.AMAR

Acknowledgments

First of all, praise and thanks be to Allah, the most beneficent, the most merciful, for providing us with countless opportunities and guiding us through our educational careers.

We would like to thank our supervisor ***Prof****.* ***KHELDOUN. A*** *for his support, guidance, and patience throughout our research journey. We thank him for all the invaluable suggestions he offered and for providing us with the necessary tools to succeed. May Allah reward him for his kindness and hard work.*

Special thanks go to all those who have supported and motivated us throughout our research journey. You have truly helped us carry this humble work to the finish line.

Table of Contents

Abstract ii

Dedication iii

Acknowledgments iv

Table of Contents v

List Of Tables ix

List of Figures x

General Introduction xi

CHAPTER 1 THE AUTOMATIC VOLTAGE REGULATOR MODELING

1.1 Introduction 1

1.2 The Need for Automatic Voltage Regulation 1

1.3 Power System Control 2

1.3.1 Voltage and Reactive Power Control 3

1.3.2 Excitation System 3

1.4 Voltage Regulation Concept 4

1.4.1 Automatic Voltage Regulator (AVR) 5

1.4.2 The simplified model of an AVR system with FOPID Controller 6

1.4.3 AVR block diagram 7

1.4.3.1 FOPID controller 8

1.4.3.2 Amplifier model 9

1.4.3.3 Exciter model 9

1.4.3.4 Generator model 9

1.4.3.5 Sensor model 9

1.5 Conclusion 10

Chapter 2 The proposed Metaheuristic Algorithms

2.1 Introduction. 11
2.2 Ant Lion Optimizer (ALO) Algorithm 11
2.2.1 Inspiration 11
2.2.2 Concept of ALO algorithm 12
2.2.3 ALO parameters 13
2.2.3.1 The position of ants 13
2.2.3.2 The fitness function 14
2.2.3.3 The position of each antlion 14
2.2.4 Random walks of ants 16
2.2.5 Trapping in antlion's pits 16
2.2.6 Building trap 17
2.2.7 Sliding ants towards antlion 17
2.2.8 Catching prey and re-building the pit 18
2.2.9 Elitism 19
2.3 ALO algorithm 20
2.3.1 Flow chart of the Ant Lion Optimization (ALO) 21
2.4 Particle swarm optimization (PSO) algorithm 22
2.4.1 Concept of PSO algorithm 23
2.4.2.1 Population size 24
2.4.2.2 Personal & Global Best 24
2.4.2.3 Particle's Velocity 25
2.4.2.4 Position update 27
2.4.2.5 Stopping criteria 28

2.5 Grey Wolf Optimization (GWO) algorithm....28
2.5.1 Concept of GWO algorithm....29
2.5.2 GWO Parameters....29
2.6 Conclusion....31

Chapter 03 PID Design

3.1 Introduction....32
3.2 The Proportional-Integral-Derivative (PID) Control Theory....32
3.2.1 The Proportional Term (P)....33
3.2.2 The Integral Term (I)....34
3.2.3 The Derivative Term (D)....34
3.3 Tuning of a PID Controller....35
3.4 PID Parameter Design Using ZN Rule....36
3.4.1 Simulation Results....37
3.5 Uncontrolled AVR Performance....38
3.6 Performance Specification....39
3.6.1 Fitness Function....40
3.6.2 Algorithms Settings....41
3.7 Obtained results....42
3.8 Conclusion....47

CHAPTER 04 FOPID Design

4.1 Introduction....48
4.2 Why Fractional Order....48
4.3 Fractional Calculus....49

4.3.1 Definitions 49
4.3.2 Integer Order Approximations 50
4.4 Fractional Order PID Controller: 51
4.4.1 Integral Action 52
4.4.2 Derivative Action: 52
4.5 FOPID Tuning: 53
4.5.1 FOPID controller specifications 53
4.5.2 The Proposed Tuning Algorithm 54
4.5.3 The Proposed Objective Functions for FOPID 54
4.6 Simulation results 55
4.7 Obtained results 55
4.7.1 Comparison with ALO-PID Controller: 56
4.7.2 Stability check 57
4.7.3 Comparison with other FOPID Controllers 58
4.7.4 Robustness test 59
4.8 Conclusion 59

List of Tables

Table 1.1 the three main components of AVR system 5

Table 1.2 Models of the components in an AVR system 10

Table 3.1 Controller parameters of Closed-loop ZN Tuning Rule 37

Table 3.2 Performances of ZN based controller of the AVR system 38

Table 3.3 Time performance indices of the uncontrolled AVR 38

Table 3.4 some used fitness functions 40

Table 3.5 Ant Lion Optimization (ALO) Settings 41

Table 3.6 Particle Swarm Optimization (PSO) Settings 42

Table 3.7 Grey Wolf Optimization (GWO) Settings 42

Table 3.8 Results of ALO 43

Table 3.9 Results of PSO 43

Table 3.10 Results of GWO 44

Table 3.11 Results of the transient response and the optimum parameters of the PID 46

Table 4.1 Positive and Negative effects of Derivative and Integral actions 49

Table 4.2 cost functions 52

Table 4.3 Performances of different objective function 55

Table 4.4 Improvement of the performances 57

Table 4.5 Best performance indices of different FOPID controllers 58

List of Figures

Fig 1.1 Schematic diagram of different timescales of power system controls 2

Fig.1.2 Schematic diagram of a synchronous machine with excitation system control.....4

Fig 1.3 the model of AVR system.....6

Fig 1.4 Schematic diagram of AVR control loop with fractional order PID controller.....7

Fig 1.5 Block diagram of the AVR system 8

Fig 2.1 Cone-shaped traps and hunting behavior of antlions 12

Fig 2.2 Three random walks 13

Fig 2.3 Random walk of an ant inside an antlion's trap.....17

Fig 2.4 Adaptive lower (ct) and upper (dt) bounds.....19

Fig 2.5 Flow chart of the Ant Lion Optimization (ALO).....21

Fig 2.6 how swarm of birds move into better regions.....22

Fig 2.7 Deception of position and velocity updates.....27

Fig 3.1 Controller Inputs and Outputs.....23

Fig 3.2 Block Diagram of the PID controller, ...33

Fig 3.3 Proportional control terme.....34

Fig 3.4 Integral control term.....34

Fig 3.5 Derivative control term.....35

Fig 3.6 The ultimate period measurement of ZN closed loop tuning method. 36

Fig 3.7 Terminal voltage step response of the AVR system with ZN tuning method 37

Fig 3.8 Uncontrolled AVR step response.....38

Fig 3.9 Unit step response of AVR system with different algorithm 45

Fig 3.10 Comparison of terminal voltage step response of AVR system with optimized PID 46

Fig 4.1 Fractional-order PID vs classical PID from points to plane 52

Fig 4.2 Terminal voltage step response of the AVR controlled with FOPID.....56

Fig 4.3 Terminal voltage step response of AVR system controlled by ALO-FOPID, and ALO PID 56

Fig 4.4 Bode diagram 57

Fig 4.5 Terminal voltage step response of AVR system controlled by ALO-FOPID, , ALO[37]-FOPID and GA-FOPID 58

Fig 4.6 Voltage response profiles of the AVR system using.....59

Fig 4.7 Terminal voltage step response of AVR system controlled by ALO-FOPID, and ALO-PID 59

General Introduction

An automatic voltage regulator (AVR) system is used to maintain terminal voltage magnitude of a generator in an electrical power system. The magnitude of this voltage is maintained at a specified level by controlling the generator excitation. Generally, it is hard to find optimal controller parameters with classical tuning methods such as Ziegler-Nichols, Cohen Coon, and gain-phase margin. Therefore, several heuristic optimization methods have been proposed for tuning controller parameters, during the past two decades.

In literature, the controller types used for improving the dynamic response of AVR system are proportional-integral-derivative (PID) , fractional order PID (FOPID) , PID-acceleration (PIDA) , gray PID (GPID) , and fuzzy logic PID (FLPID) the present project aims to investigate and design a Fractional Order PID (FOPID) controller for automatic voltage regulation. The heuristic optimization based tuning methods that have been applied to improve the performances is Ant Lion Optimization (ALO) algorithm. The report is structured into four main chapters. Chapter 1 deals with power system control. Main aim of AVR system is to control and maintain the voltage between limits by adjusting the excitation of the alternator. Then, Chapter 2 discusses different investigated algorithms, namely the Ant Lion Optimization (ALO), Grey Wolf Optimization (GWO) and Practical Swarm Optimization (PSO) algorithms. In Chapter 3, discusses the tuning of proportional integral derivative (PID) controllers**.** Chapter 4 includes the basic concepts of FOPID controllers. The tuning process of FOPID controllers has also been discussed, the simulation results are given and discussed Finally, the project is end up by a general conclusion.

Chapter1

THE AUTOMATIC VOLTAGE REGULATOR MODELING

1.1 Introduction

In the power system it is necessary that system should be operate in steady state condition so most of the equipment's are designed to work in predetermined values of voltage and frequency of operation. So it is desirable that overall voltage profile must be within the permissible limit at all the times during the operation. The load on the system is not constant all the time, it's always varying and these load change requirements we can get through automatic excitation system, automatic excitation system in the sense that Automatic Voltage Regulator so it is used to stabilize the voltage in the power system. In case of any deviation from these values results in decrease in performance and lifetime of these equipment's. Hence for effective operation of the power system the Automatic Voltage Regulator (AVR) is installed at each generating plants. The main objective of AVR system is to maintain the terminal voltage of the alternator in the generating station. The Power system consist of generator it having Automatic Voltage Regulator to stabilize their voltage [1].

This chapter will give a brief explanation of the automatic voltage regulator.

1.2 The Need for Automatic Voltage Regulation

The deviation of voltage in power plant or in any generation station can damage many equipment, these damages will add extra costs and downtime. then the automatic voltage regulator is needed to satisfy many objectives, such as:

- Keeping the voltage levels within a safety range.
- Avoiding malfunction operations of electrical equipment.
- To maintain the system's voltage within the operating range and keep the operation of the machines near to the steady state stability limit.
- To adjust and regulate sharing of the reactive load between machines operating in parallel.
- To keep voltage under system fault conditions that ensures rapid operation of protective devices (as: relays, switches…)
- Provide close control of field circuits, in order to keep machine in synchronization with the system when operating at unity or leading power factor.

1.3 Power System Control

The power system comprises the subsystems: Electricity Generation, Transmission, Distribution and Consumption (Loads). The associated control system has a hierarchical structure. This means that the control system consists of a number of nested control loops that control different quantities in the system. In general, the control loops on lower system levels (example, locally in the generation station) are characterized by smaller time constants than the control loops on a higher system level. In our case, the automatic voltage regulation system, which regulates the voltage of the generator terminals to the reference (set) value, responds typically in a timescale of a second or less, while the secondary voltage control (transformer stations), which determines the reference values of the voltage controlling devices, among which are the generators, operates in a timescale of tens of seconds or minutes. That means that these two control loops are virtually decoupled from each other. This is also generally true for other controls in the system, resulting in a number of decoupled control loops operating in different time-scales [2].

A schematic diagram showing the different timescales is presented in Figure 1.1. The overall control system is very complex, but due to the decoupling, it is in most cases possible to study the different control loops individually. This facilitates the task, and with appropriate simplifications, one can quite often use classical standard control theory methods to analyze these controllers.

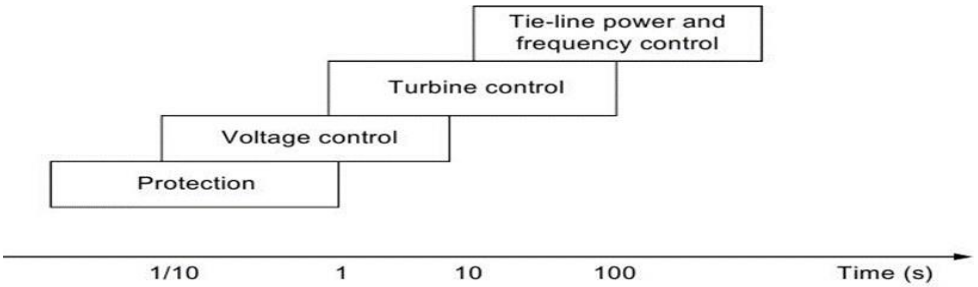


Fig 1.1 Schematic diagram of different timescales of power system controls

1.3.1 Voltage and Reactive Power Control

Dynamically maintaining constant (or controlled) voltage in a power system is a fundamental requirement of power quality. Passive (inductive, capacitive) loads and active loads (motors) require both active and reactive power flows in the power system [2]. Since the reactive power cannot be transmitted over a long distance, voltage has to be maintained by special devices, the proper selection and coordination of equipment for controlling reactive power and voltage are among the major challenges of power systems engineering.

The control of voltage levels is accomplished by controlling the production, absorption, and flow of reactive power at all levels in the system. the main methods used to control the voltage levels:

(a) At generating units, automatic voltage regulator (AVR) controls field excitation to maintain the terminal voltage of the generator constant.

(b) Power-electronics-controlled capacitors and inductors by static voltage controllers placed at various locations in a power system.

1.3.2 Excitation System

The basic function of an excitation system is to provide direct current to the synchronous machine field winding. The excitation system must supply and adjust automatically the field current of the synchronous generator to maintain the terminal voltage as the output varies within the continuous capability of the generator. From power system viewpoint, the excitation system should contribute to effective control of voltage and enhancement of system stability. It should be capable of responding rapidly to a disturbance to enhance transient stability. Based on the excitation power source, three main excitation system are distinguished:

(a) DC excitation system: the excitation system of this kind utilize DC generator as a source of excitation power and provide current to the rotor of synchronous machine through slip rings.

(b) AC excitation system: the excitation system of this category utilize alternators as a source of the main generator power. Controlled or non-controlled rectifiers rectify the ac output of this alternator.

(c) Static excitation system: all the component in the system are static, static rectifiers, controlled or uncontrolled, the supply power of the rectifiers is from the main generator through a transformer to step down the voltage to appropriate level.

A schematic diagram of a synchronous machine with excitation system is shown in Figure 1-2

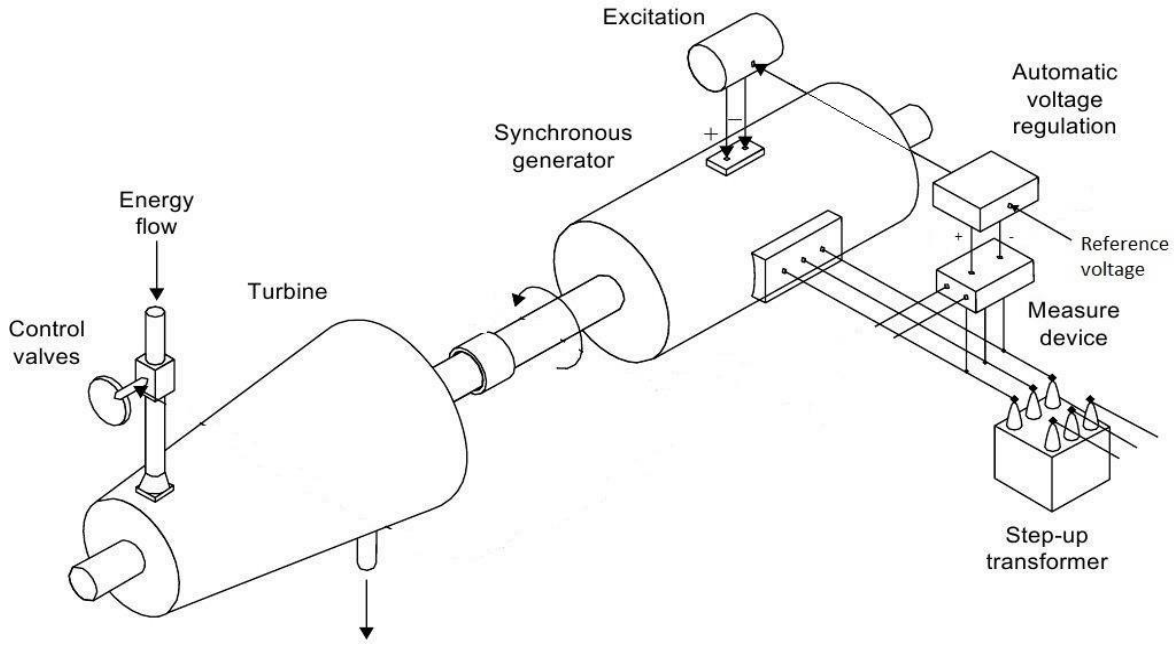


Fig 1.2 Schematic diagram of a synchronous machine with excitation system control.

The reactive power output of synchronous machine can be adjusted within the limits of the capability curve by the excitation system. This offers a very efficient and fast way to control the terminal voltage of the machine, which is the most important way of voltage control in most power systems.

1.4 Voltage Regulation Concept

The basic function of an AVR system is to maintain the terminals voltage of asynchronous generator within a specific range. When there is a change in the load power system demand the AVR system stabilizes the voltages. As we knew previously this can be performed using excitation

control method, then the generated field current allows controlling the reactive power and therefore the induced primary voltage. In Automatic Voltage Regulation, the synchronous generator (SG), AVR, and exciter control circuits form a closed loop control system. The Automatic Voltage Regulator (AVR) senses the generator terminal voltage and adjusts the excitation in order to keep the voltage within the adequate range and required limits. A limiting circuitry is built into the AVR. Table1 summarizes the three main components of AVR system [4].

Table 1.1 the three main components of AVR system.

Different blocks	Main operation
Exciter	Supplies the field winding with direct current and thus comprises the Power part of the excitation system.
Controller	Treats and amplifies the input signals to a level and form that is suitable for the control of the exciter. Input signals are pure control signals as well as functions for stabilizing the exciter system.
Measuring device	Measures the terminal voltage of the generator, rectifies and filters it. Load compensation can be implemented if the voltage at a point away from the generator terminals, such as at a fictional point inside the generator's transformer, should be kept constant.

1.4.1 Automatic Voltage Regulator (AVR)

AVR (Automatic voltage regulator) is a system which mainly designed to automatically maintain a constant voltage level. It is used the power system to stabilize voltage which occurs because of variation the load also it is an instrument that adjusts voltage by means of automatic control device. The automatic voltage regulator is used to control the variation in the voltage. It senses the fluctuation in voltage and changes them into a stable voltage. The variation in the voltage occurs because of the variation in load. The variation in voltage is harmful for the equipment of the power system. Generally, the AC automatic voltage regulator is a device designed to regulate

voltage automatically that is, it senses variation in the voltage level and turn it into a stable voltage level. The automatic voltage regulators for generators were electromechanical systems, but a modern AVR uses solid-state devices. An AVR is work as feedback control system that measures the output voltage of the generator and compares that output to a set point, and generates an error signal that error signal controls the excitation of the generator. As the terminal voltage of the alternator increases as the excitation current in the field winding of the generator increases [5]. The model of the *AVR* system is depicted in Figure 1.3**.**

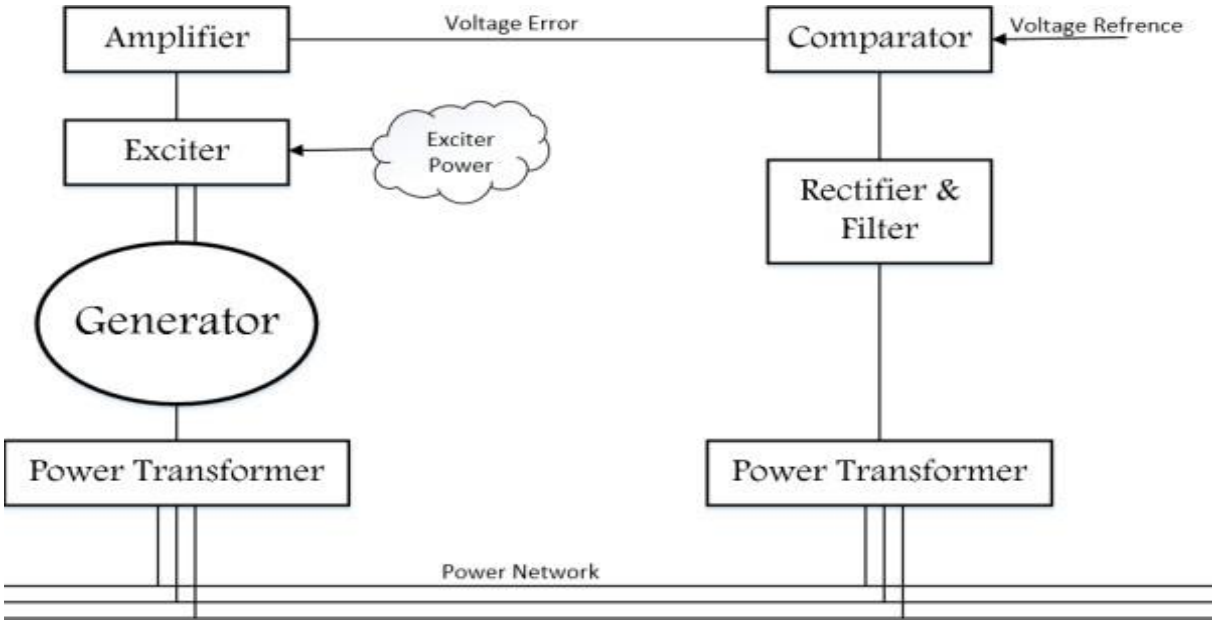


Fig 1.3 the model of AVR system [5]

1.4.2 The simplified model of an AVR system with FOPID Controller

The components of the *AVR* system are generator, exciter, amplifier, sensor and *PID* controller .In our project, the classical PID will be replaced by the fractional order *PID*

where Kp**,** Ki, and Kd represent the gains of the proportional, integral, and derivative components, respectively; λ and μ are the real number orders with $0 < \lambda < 2$ and $0 < \mu < 2$. Figure 1.4 shows the schematic diagram of the AVR system with FOPID controller.

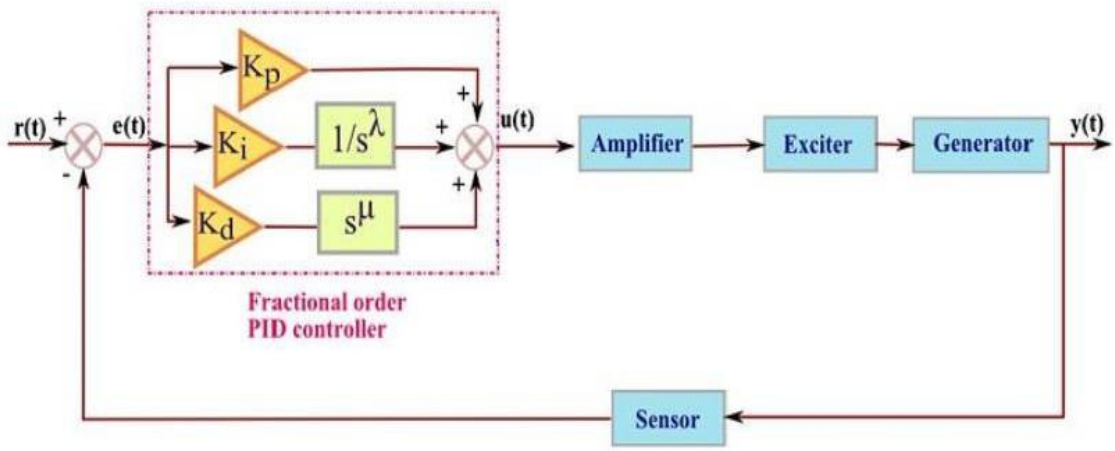


Fig 1.4 Schematic diagram of AVR control loop with fractional order PID controller [6].

The voltage sensor is repeatedly sense the output voltage y(t) of the generator. and the signal rectified ,smoothed and analyzed to see the deviation from the reference signal in the comparator .The error voltage resulting from this is used by the fractional order PID controller to generate a control signal which is then amplified to control the field windings of the generator by means of exciter [6].

1.4.3 AVR block diagram

The disturbances such as a load changes and transmission line parameter changes on a synchronous generator, cause an oscillatory behavior .This electromechanical oscillations are harmful to the stability of the power system .To enhance the power system stability ,the synchronous generators are equipped with excitation system which is controlled by an AVR .[7] , Figure 1.5 shows the transfer functions description of each components.

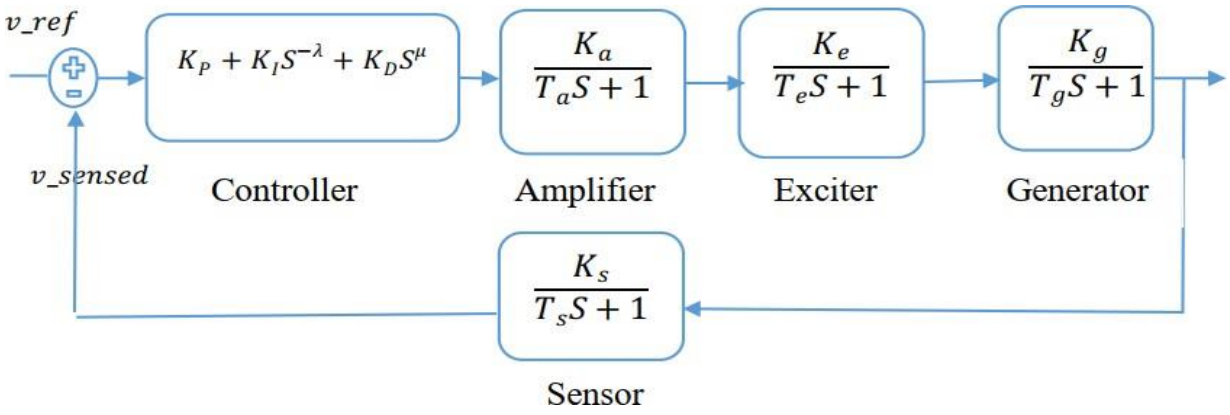


Fig 1.5 Block diagram of the AVR system

1.4.3.1 FOPID controller

The fractional order PID controllers (FOPID) is a generalization of the PID controller. FOPID controllers are characterized by five parameter, which are the proportional gain KP, the integrating gain KI, the derivative gain KD, the fractional integrating order λ, and the fractional derivative order μ. The $FOPID$ has two extra parameters compared to classical PID controller. It enables to have more freedom to design a $FOPID$ controller. The transfer function is given by

$$T_{FOPID}(s) = K_P + K_I S^{-\lambda} + K_D S^{\mu} \qquad (1.1)$$

where

$\boldsymbol{KP}$: Proportional gain.
$\boldsymbol{KI}$: Integrating gain.
$\boldsymbol{KD}$: Derivative gain.
$\boldsymbol{\lambda}$: integrating order.
$\boldsymbol{\mu}$:derivative order.

1.4.3.2 Amplifier model

Transfer function of an exciter is modeled by a gain and a time constant given by:

$$\frac{V_R(s)}{V_E(s)} = \frac{K_A}{1+\tau_A S} \tag{1.2}$$

where KA and τA are respectively the gain and time constant of the amplifier system. Usual values of KA are in the range of 10–40 and the amplifier time constant τA is very small ranging from 0.02 to 0.1 s.

1.4.3.3 Exciter model

Transfer function of an exciter modeled by a gain of KE and a time constant of τE is given by:

$$\frac{V_F(s)}{V_R(s)} = \frac{K_E}{1+\tau_E S} \tag{1.3}$$

Typical values of KE are in the range of 10,400 and the time constant τE is in the range of 0.5–1

1.4.3.4 Generator model

The generator is represented by a transfer function given by:

$$\frac{V_t(s)}{V_F(s)} = \frac{K_G}{1+\tau_G S} \tag{1.4}$$

The generator gain KG and time constant τG are load dependent. KG varies between 0.7 and 1.0, and τG between 1.0 and 2.0 s from full load to no load.

1.4.3.5 Sensor model

A sensor may be represented by a simple first-order transfer function with a gain KR and time constant τR and is given by:

$$\frac{V_s(s)}{V_t(s)} = \frac{K_R}{1+\tau_R S} \tag{1.5}$$

Normally $\boldsymbol{\tau}$R is very small, ranging from 0.001 to 0.06 s and KR is around 1.0.

Table 1.2 summarize all the transfer functions, with ranges of each parameter included in the transfer functions:

Table 1.2 Models of the components in an AVR system.

	Transfer function	Parameter limits Used parameter values	limits Used parameter values
amplifier	$\boldsymbol{Ta(s)} = \frac{\mathbf{KA}}{\mathbf{1+\tau_A S}}$	$10 \leq Ka \leq 40$ $0.02 \leq ra \leq 0.1$	Ka= 10 ra = 0.1
Exciter	$\boldsymbol{Te(s)} = \frac{\mathbf{KE}}{\mathbf{1+\tau_E S}}$	$1 \leq Ke \leq 2$ $0.4 \leq re \leq 1.0$	Ke = 1.0 re = 0.4
Generator	$\boldsymbol{TG(s)} = \frac{\mathbf{KG}}{\mathbf{1+\tau_G S}}$	$0.7 \leq KG \leq 1$ $1.0 \leq rG \leq 2.0$	KG = 1.0 rG = 1.0
Sensor	$\boldsymbol{TS(s)} = \frac{\mathbf{ks}}{\mathbf{1+\tau_s S}}$	KS = 1 $0.001 \leq rs \leq 0.06$	KS = 1.0 rS = 0.01

1.5 Conclusion

In this chapter, we have seen that an AVR is a control system which can automatically maintain a constant AC voltage level. It basically consists of simple negative feedback control loop. The AVR may use either an electromechanical mechanism, or electronic components. The continuous improvement in power electronic controls and processor power is bringing further advances in voltage control, with more flexible protection of the generator and its connected circuits. The importance of maintaining the terminal voltage level within safety range was discussed. The automatic voltage regulator has been introduced, along with the most basic concepts governing it and its closed control loop using Fractional orders PIDs.

Chapter 2

The proposed Metaheuristic Algorithms

2.1 Introduction

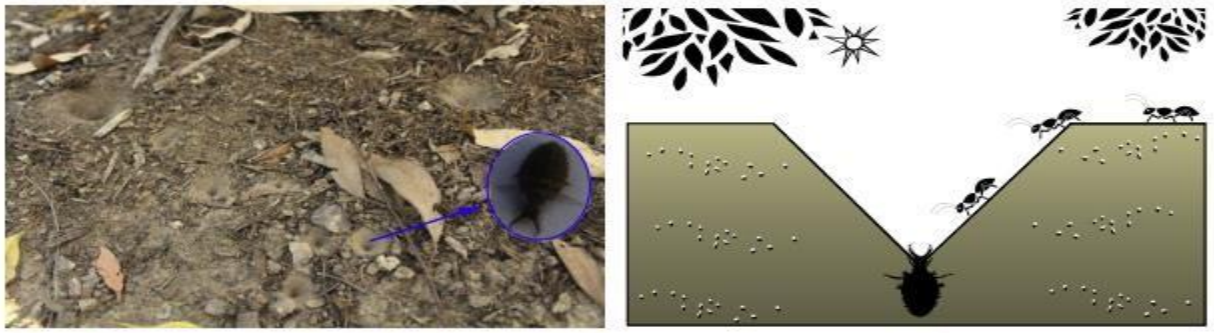
Meta-heuristic Algorithms are now becoming powerful methods for solving many tough optimization problems. The vast majority of these algorithms have been derived from the behavior of biological systems and/or physical systems in nature [8]. To carry out the aforementioned design procedure, this chapter will presented the three meta-heuristics algorithms that will be used for the sake of tuning the parameters of FOPID controller. Sections 2.2, 2.4, and 2.5 will introduce the Ant Lion Optimization **(ALO),** Particle Swarm Optimization **(PSO**), and Grey Wolf Optimization **(GWO),** respectively.

2.2 Ant Lion Optimizer (ALO) Algorithm

Optimization problems exist in different fields of studies. To solve an optimization problem, different steps need to be taken. Firstly, the parameters of the problem should be identified. Based on the nature of the parameters, problems may be classified as continuous or discrete. Secondly, the constraints that are applied to the parameters have to be recognized (**Seedily, 2015**) [9].

2.2.3.1 Inspiration

Antlions (doodlebugs) belong to the Myrmeleontidae family and Neuroptera order (net-winged insects). The lifecycle of antlions includes two main phases: larvae and adult. A natural total lifespan can take up to 3 years, which mostly occurs in larvae (only3–5 weeks for adulthood) [10]. Antlions undergo metamorphosis in a cocoon to become adult. They mostly hunt in larvae and the adulthood period is for reproduction. Their names originate from their unique hunting behavior and their favorite prey. An antlion larva digs a cone-shaped pit in sand by moving along a circular path and throwing out sands with its massive jaw Fig 2.1(a) shows several cone-shaped pits with different sizes. After digging the trap, the larvae hide underneath the bottom of the cone (as a sit-and-wait predator and waits for insects (preferably ant) to be trapped in the pit as illustrated in Fig 2.1(b)



(a) (b)

Fig 2.1 Cone-shaped traps and hunting behavior of antlions

The edge of the cone is sharp enough for insects to fall to the bottom of the trap easily. Once the antlion realizes that a prey is in the trap, it tries to catch it. However, insects usually are not caught immediately and try to escape from the trap. In this case, antlions intelligently throw sands towards to edge of the pit to slide the prey into the bottom of the pit. When a prey is caught into the jaw, it is pulled under the soil and consumed. After consuming the prey, antlions throw the leftovers outside the pit and amend the pit for the next hunt [11].

Another interesting behavior that has been observed in life style of antlions is the relevancy of the size of the trap and two things: level of hunger and shape of the moon. Antlions tend to dig out larger traps as they become hungrier and/or when the moon is full. They have been evolved and adapted this way to improve their chance of survival. It also has been discovered that an antlion does not directly observe the shape of the moon to decide about the size of the trap, but it has an internal lunar clock to make such decisions [9].

2.2.3.2 Concept of ALO algorithm

The ALO algorithm mimics interaction between antlions andantes in the trap. To model such interactions, ants are required to move over the search space, and antlions are allowed to hunt them and become fitter using traps. Since ants move stochastically in nature when searching for food, a random walk is chosen for modelling ants' movement as follows:

$$X(t) = [0, \text{cumsum}\,(2r(t_1) - 1), \text{cumsum}\,(2r(t_2) - 1), \dots, \text{cumsum}\,(2r(t_n) - 1)] \quad (2.1)$$

where cumsum calculates the cumulative sum, **n** is the maximum number of iteration, **t** shows the step of random walk (iteration in this study), and **r**(**t**) is a stochastic function defined as follows:

$$r(t) = \begin{cases} 1 & \text{if rand} > 0.5 \\ 0 & \text{if rand} \leqslant 0.5 \end{cases} \quad (2.2)$$

where t shows the step of random walk (iteration in this study) and rand is a random number generated with uniform distribution in the interval of [0,1]. To have an image of this random walk, **Fig 2.2** is provided that illustrates three random walks over 500 iterations. This figure shows that the random walk utilized may fluctuate dramatically around the origin (red curve), have increasing trend (black curve), or have descending behavior (blue curve) [9].

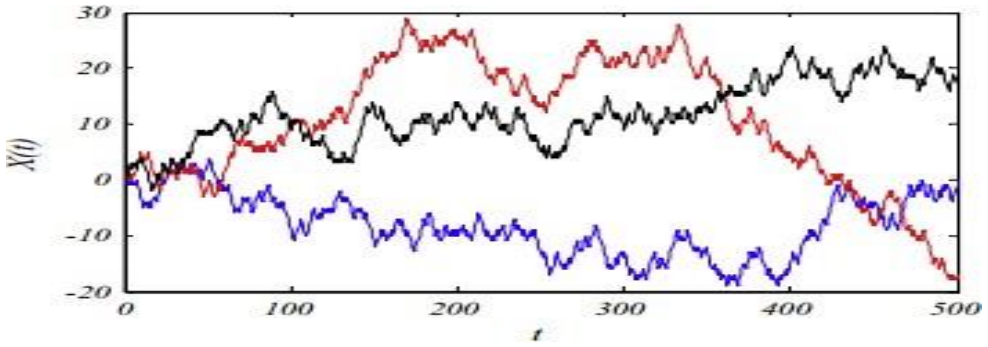


Fig 2.2 Three random walks

2.2.3.3 ALO parameters

2.2.3.1 The position of ants

The position of ants are saved and utilized during optimization in the following matrix:

$$M_{\text{Ant}} = \begin{bmatrix} A_{1,1} & A_{1,2} & \dots & \dots & A_{1,d} \\ A_{2,1} & A_{2,2} & \dots & \dots & A_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ [A_{n,1} & A_{n,2} & \dots & \dots & A_{n,d}] \end{bmatrix} \quad (2.3)$$

where M_{Ant} is the matrix for saving the position of each ant, $A_{i,j}$ shows the value of the j-th variable (dimension) of i-th ant, n is the number of ants, and d is the number of variables. It should be noted that ants are similar to particles in PSO or individuals in GA. The position of an ant refers the parameters for a particular solution. Matrix M_{Ant} has been considered to save the position of all ants (variables of all solutions) during optimization [9].

2.2.3.2 The fitness function

For evaluating each ant, a fitness (objective) function is utilized during optimization and the following matrix stores the fitness value of all ants:

$$M_{OA} = \begin{bmatrix} f([A_{1,1}, A_{1,2}, \dots, A_{1,d}]) \\ f([A_{2,1}, A_{2,2}, \dots, A_{2,d}]) \\ \vdots \\ \vdots \\ [f([A_{n,1}, A_{n,2}, \dots, A_{n,d}])] \end{bmatrix} \quad (2.4)$$

where M_{OA} is the matrix for saving the fitness of each ant, Ai, j shows the value of j-th dimension of i-th ant, n is the number of ants, and f is the objective function

2.2.3.3 The position of each antlion

In addition to ants, we assume the antlions are also hiding somewhere in the search space. In order save their positions and fitness values, the following matrices are utilized:

$$M_{\text{Antlion}} = \begin{bmatrix} AL_{1,1} & AL_{1,2} & \dots & \dots & AL_{1,d} \\ AL_{2,1} & AL_{2,2} & \dots & \dots & AL_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ [AL_{n,1} & AL_{n,2} & \dots & \dots & AL_{n,d}] \end{bmatrix} \quad (2.5)$$

where M_{Antlion} is the matrix for saving the position of each antlion, AL_i, j shows the j-th dimension's value of i-th antlion, n is the number of antlions, and d is the number of variables (dimension).

$$M_{OAL} = \begin{bmatrix} f([AL_{1,1}, AL_{1,2}, \dots, AL_{1,d}]) \\ f([AL_{2,1}, AL_{2,2}, \dots, AL_{2,d}]) \\ \vdots \\ \vdots \\ [f([AL_{n,1}, AL_{n,2}, \dots, AL_{n,d}])] \end{bmatrix} \quad (2.6)$$

where M_{OAL} is the matrix for saving the fitness of each antlion, AL_i, j shows the j-th dimension's value of i-th antlion, n is the number of antlions, and f is the objective function

During optimization, the following conditions are applied:

- Ants move around the search space using different random walks.
- Random walks are applied to all the dimension of ants.
- Random walks are affected by the traps of antlions.
- Antlions can build pits proportional to their fitness (the higher fitness, the larger pit).
- Antlions with larger pits have the higher probability to catch ants.
- Each ant can be caught by an antlion in each iteration and the elite (fittest antlion).
- The range of random walk is decreased adaptively to simulate sliding ants towards antlions.
- If an ant becomes fitter than an antlion, this means that it is caught and pulled under the sand by the antlion.
- An antlion repositions itself to the latest caught prey and builds a pit to improve its change of catching another prey after each hunt [12]

2.2.3.4 Random walks of ants

Random walks are all based on the Eq (2.1). Ants update their positions with random walk at every step of optimization. Since every search space has a boundary (range of variable), however, Eq (2.1) cannot be directly used for updating position of ants. In order to keep the random walks inside the search space, they are normalized using the following equation (min–max normalization):

$$X_i^t = \frac{(X^t - a_i) \times (d_i - c^t)}{(d_i^t - a_i)} + c_i \qquad (2.7)$$

where a_i is the minimum of random walk of i-th variable, bi is the maximum of random walk in i-th variable, $c^t{}_i$ is the minimum of i-th variable at t-th iteration, and $d^t{}_i$ indicates the maximum of i-th variable at t-th iteration Eq.(2.7) should be applied in each iteration to guarantee the occurrence of random walks inside the search space [9].

2.2.3.5 Trapping in antlion's pits

As discussed above, random walks of ants are affected by antlions' traps. In order to mathematically model this assumption, the following equations are proposed:

$$c_i^t = \text{Antlion}_j^t + c^t \qquad (2.8)$$

$$d_i^t = \text{Antlion}_j^t + d^t \qquad (2.9)$$

where ct is the minimum of all variables at t-th iteration, dt indicates the vector including the maximum of all variables at t-th iteration, ct j is the minimum of all variables for i-th ant, dt j is the maximum of all variables for i-th ant, and Antlion j shows the position of the selected j-th antlion at t-th iteration

Eqs (2.8) and (2.9) show that ants randomly walk in a hyper sphere defined by the vectors **c** and d around a selected antlion. A conceptual model of this behavior is illustrated in Fig 2.3.

Fig 2.3 shows a two-dimensional search space. It may be observed that ants are required to move within a hypersphere around a selected antlion.

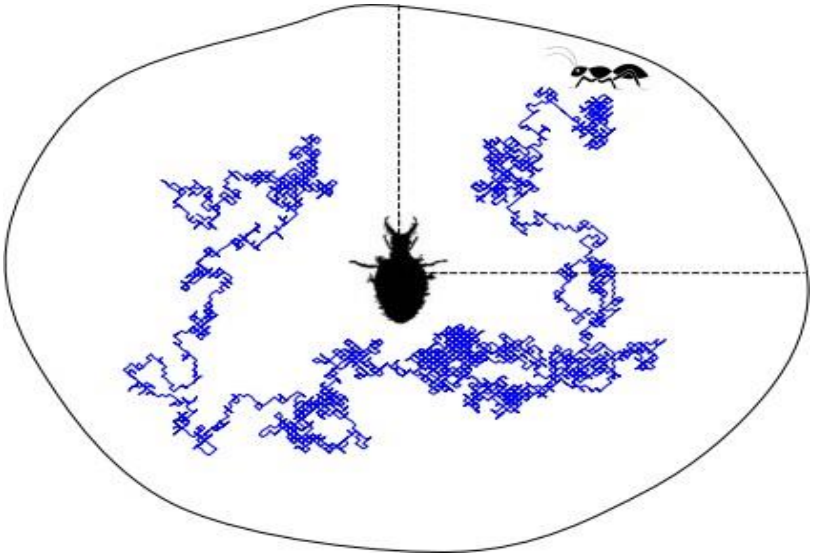


Fig 2.3 Random walk of an ant inside an antlion's trap

2.2.3.6 Building trap

In order to model the antlions' hunting capability, a roulette wheel is employed. As Fig.2. 3 show ants are assumed to be trapped in only one selected antlion. The ALO algorithm is required to utilize a roulette wheel operator for selecting antlions based of their fitness during optimization. This mechanism gives high chances to the fitter antlions for catching ants. [9]

2.2.3.7 Sliding ants towards antlion

With the mechanisms proposed so far, antlions are able to build traps proportional to their fitness and ants are required to move randomly. However, antlions shoot sands outwards the center of the pit once they realize that an ant is in the trap. This behavior slides down the trapped ant that is trying to escape. For mathematically modelling this behavior, the radius of ants's random walks

hyper-sphere is decreased adaptively. The following equations are proposed in this regard:

$$c^t = \frac{c^t}{I} \tag{2.10}$$

$$d^t = \frac{d^t}{I} \tag{2.11}$$

where I is a ratio, c^t is the minimum of all variables at t-th iteration, and d_t indicates the vector including the maximum of all variables at t-th iteration. In Eqs. (2.15) and (2.16), I ¼ 10**w** Tt where t is the current iteration, T is the maximum number of iterations, and w is a constant defined based on the current iteration (w = 2 when **t** > 0.1T, w = 3 when t > 0.5T, w = 4 when t > 0.75T, w = 5 when t > 0.9T, and w = 6 when t > 0.95T). Basically, the constant **w** can adjust the accuracy level of exploitation. Fig.2. 4 also shows the decreasing behavior using Eqs. (2.10) and (2.11). These equations shrink the radius of updating ant's positions and mimics sliding process of ant inside the pits. This guarantees exploitation of search space.

2.2.3.8 Catching prey and re-building the pit

The final stage of hunt is when an ant reaches the bottom of the pit and is caught in the antlion's jaw. After this stage, the antlion pulls the ant inside the sand and consumes its body. For mimicking this process, it is assumed that catching prey occur when ants becomes fitter (goes inside sand) than its corresponding antlion.

An antlion is then required to update its position to the latest position of the hunted ant to enhance its chance of catching new prey. The following equation is proposed in this regard

$$\text{Antlion}_j^t = Ant_i^t \text{ if } f(Ant_i^t) > f(\text{Antlion}_j^t) \tag{2.12}$$

where **t** shows the current iteration, $\text{Antlion}^t{}_j$ shows the position of selected j-th antlion at t-th iteration, and $\text{Ant}^t{}_i$ indicates the position of i-th ant at t-th iteration.

2.2.3.9 Elitism

Elitism is an important characteristic of evolutionary algorithms that allows them to maintain the best solution(s) obtained at any stage of optimization process. In this study the best antlion obtained so far in each iteration is saved and considered as an elite. Since the elite is the fittest antlion, it should be able to affect the movements of all the ants during iterations. Therefore, it is assumed that every ant randomly walks around a selected antlion by the roulette wheel and the elite simultaneously as follows:

$$Ant_i^t = \frac{R_A^t + R_E^t}{2} \tag{2.13}$$

where R^t_A is the random walk around the antlion selected by the roulette wheel at t-th iteration, R^t_E is the random walk around the elite at t-th iteration, and Ant^t_i indicates the position of i-th ant at t-th iteration.

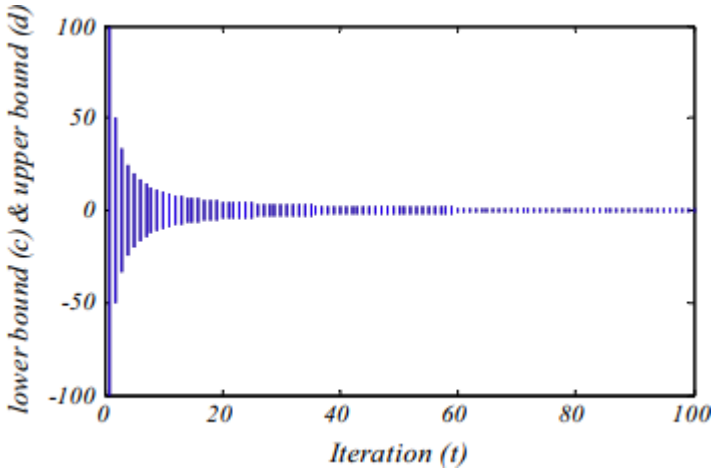


Fig 2.4 Adaptive lower (ct) and upper (dt) bounds

With the proposed operators in the preceding subsections, the ALO optimization algorithm can now be defined. The ALO algorithm is defined as a three-tuple function that approximates the global optimum for optimization problems as follows:

$$\mathrm{ALO}(\mathrm{A}, \mathrm{B}, \mathrm{C}) \tag{2.14}$$

where A is a function that generates the random initial solutions, B manipulates the initial population provided by the function A, and C returns true when the end criterion is satisfied. The functions A, B, and C are defined as follows

2.3 ALO algorithm

$$\emptyset \xrightarrow{A} \{M_{\text{Ant}}, M_{OA}, M_{\text{Antlion}}, M_{OAL}\} \quad (2.15)$$

$$\{M_{\text{Ant}}, M_{\text{Antlion}}\} \xrightarrow{B} \{M_{\text{Ant}}, M_{\text{Antlion}}\} \quad (2.16)$$

$$\{M_{\text{Ant}}, M_{\text{Antlion}}\} \xrightarrow{C} \{\text{ true, false }\} \quad (2.17)$$

where M_{Ant} is the matrix of the position of ants, MAntlion includes the position of antlions**,** MOA contains the corresponding fitness of ants, and MOAL has the fitness of antlions.

The simulated annealing can be summarized as pseudo code.

Initialize the first population of ants and antlions randomly
Calculate the fitness of ants and antlions
Find the best antlions and assume it as the elite (determined optimum)
***while** the end criterion is not satisfied*
***for** every ant*
Select an antlion using Roulette wheel
*Update c and d using equations Eqs. **(2.9) and (2.10)***
*Create a random walk and normalize it using Eqs. **(2.6) and (2.17)***
*Update the position of ant using **(2.17)***
end for
Calculate the fitness of all ants
*Replace an antlion with its corresponding ant it if becomes fitter **(Eq. (2.11))***
Update elite if an antlion becomes fitter than the elite
end while
***Return** elite*

The pseudo codes the ALO algorithm [9]

2.3.1 Flow chart of the Ant Lion Optimization (ALO)

The ALO process is summarized below and is illustrated in Figure 2.5

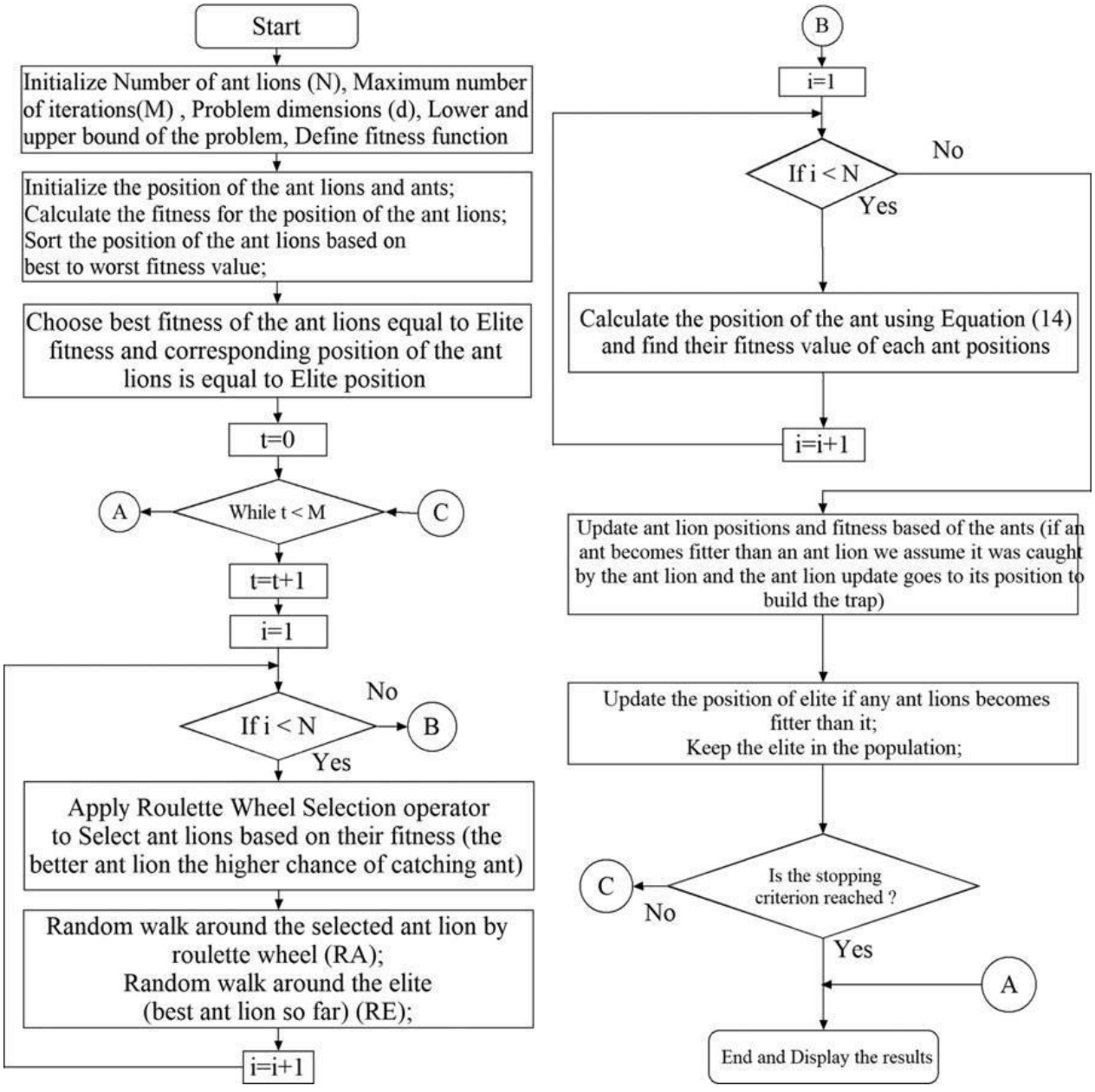


Fig 2.5 Flow chart of the Ant Lion Optimization (ALO) [13]

The of Advantages of ALO :

• Simple implementation

• Very few algorithm parameter to tune

• Very efficient global search algorithm

• Fast convergence and mostly provides better solution

• There is no need of information about the system except the cost function

2.4 Particle swarm optimization (PSO) algorithm

A ***PSO*** algorithm is a new meta-heuristic search algorithm ,inspired from the nature social behavior and dynamic movement of insects, birds and fishes ,which was proposed by *James kennedy* and *Russel Eberhar* in 1995 [15],it combines self-experiences with social experiences ,and it is applied in many fields such as optimization of parameters ,solving nonlinear equation ,fuzzy control system tuning, etc.

As described ***PSO*** algorithm imitates bird or insects behavior, Individuals insect with one another while learning from their own experience, and gradually the population members move into better regions of the problem space

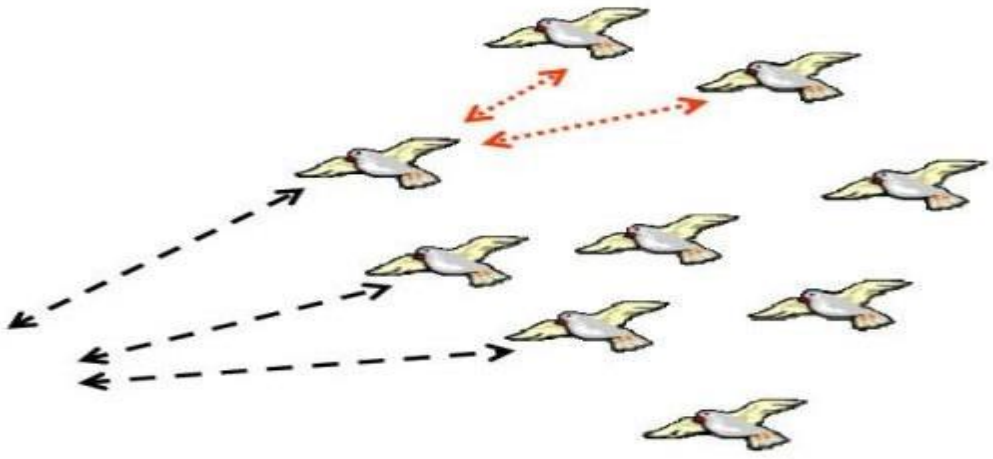


Fig 2.6 how swarm of birds move into better regions

2.4.1 Concept of PSO algorithm

PSO algorithm uses a number of agents (particles) that constitute a swarm moving around in the search space looking for the best solution .Each particle in search space adjust its flying according to its own flying experience as well as the flying experience of other particles. The *PSO* algorithm each time maintains the solutions founded by its particles in the search space. Each particle solution evaluated using an objective function to determine its fitness value. Initially, the PSO algorithm choose the solutions randomly, then evaluated according to the selected fitness function. Evaluation process allows defining best position for each particle and the best position for the whole swarm. The next position of the swarm will be determined based on the previous best position, and evaluated again. During each move (iteration), the best position of each agent is updated and the best position of the whole swarm as well. Below are the three steps that should be repeated in PSO until some stopping conditions is met:

• Evaluate the fitness of each particle;

• Update individual and global best fitness and positions;

• Update velocity and position of each particle;

The following pseudo code summarizes the major steps of the particle swarm optimization algorithm

```
For each particle
   Initialize particle
END
Do
  For each particle
  Calculate fitness value
If the fitness value is better than its peronal best
   set current value as the new pBest
End
Choose the particle with the best fitness value of all as gBest
For each particle
   Calculate particle velocity

Update particle position


   End
```

2.4.2 PSO Parameters

2.4.2.1 Population size

The population size is the total number of particles being used in the swarm, there is no regular way to choose population size in a specific application, the best value for a population size may depend on problem-specific features like the number of dimensions, constraint and other characteristics of the objective function. [16]

2.4.2.2 Personal & Global Best

The knowledge of personal and global best recorded positions is essential for the algorithm operation. In fact, updating the position of each particle in the swarm requires the knowledge of these two parameters. Even if the determination of such parameters in real animal populations is not that easy to understand, this is easy to explain for our virtual swarms by these equations.

$$P_{i,best}(t) = \begin{cases} P_{i,best}(t-1); \text{if } f[x_i(t)] \geq f[P_{i,best}(t-1)] \\ x_i(t); \text{otherwise} \end{cases} \quad (2.18)$$

$$G_{best}(t) = \text{argmin}\{ f[P_{1,best}(t)], f[P_{2,best}(t)], f[P_{3,best}(t)], \dots, f[P_{N,best}(t)] \} \quad (2.19)$$

Where:

$P_{i,best}(t)$ is the personal best position of particle i at iteration t

$f[x_i(t)]$ is fitness function evaluation at position $x_i(t)$

$x_i(t)$ is position of particle i at iteration t

$G_{best}(t)$ is the global best position discovered by the swarm from beginning till iteration t

$argmin$ is a function that returns the argument that causes the minimum image of function f .

2.4.2.3 Particle's Velocity

Let position of particle "i" among a population of "N " particles be $x_i(t)$ at time step "t", and let fitness function evaluation at this particular solution be $f_i(t)$. The next position to be visited by this particle is $x_i(t+1)$ expressed as:

$$\boldsymbol{x_i(t+1) = x_i(t) + v_i(t+1)} \tag{2.20}$$

where, $v_i(t+1)$ is the particle's new velocity. This new velocity is generally the sum of three components: inertia part, cognitive part, and social part. In the original PSO, the inertia part is just the previous particle's velocity. Besides, cognitive and social parts are attenuated differences between the previous particle's location and best positions so far visited by the particle itself and by the swarm respectively. This is shown in the following expression [17].

$$\boldsymbol{v_i(t+1) = v_i(t) + c_1 r_1[P_{best,i}(t) - x_i(t)] + c_2 r_2[G_{best}(t) - x_i(t)]} \tag{2.21}$$

Where:

$v_i(t)$ is the particle's velocity at time step "t,

$P_{best,i}(t)$ is the particle's best position visited so far since the start of the algorithm operation.

$G_{best}(t)$ is the best position visited so far by the whole population of particles.

c_2 and c_1 are acceleration coefficients.

r_1 and r_2 are random real coefficients confined to the range [0;1].

However, there are two more common ways of updating particle's velocity: inertia weight, and constriction factor. They have been both developed to improve the performance of the original PSO algorithm. They almost have the same form of velocity updating expression. Actually, both methods maintain the original three component sum expression along with providing an extra coefficient to control the effect of inertia part. They differ only in the effect's depth of this extra factor.

In inertia weight updating method, this coefficient is called "inertia weight" and affects only the previous particle's velocity as it appears in this equation.

$$v_i(t+1) = w(t) \times v_i(t) + c_1 r_1[P_{best,i}(t) - x_i(t)] + c_2 r_2[G_{best}(t) - x_i(t)] \quad (2.22)$$

The inertia weight above, $w(t)$, decreases linearly from its maximum to its minimum values during a single run as follows:

$$w(t) = w_{max} - \frac{w_{max} - w_{min}}{Iter_{max}} \times t \quad (2.23)$$

Where:

$w(t) \in [w_{min}\ ;\ w_{max}]$ is the inertia weight, commonly specified in [0.4;0.9].

$Iter_{max}$ is the maximum number of iteration in a single run of this PSO algorithm. t is the present iteration number.

This time variation of the inertia weight allows the particles to be highly affected by previous velocity in the first iterations. This effect decreases as the maximum number of iterations is being approached. Therefore, the inertia component effect is weakened to emphasize cognitive and social components' importance in grouping swarm particles at an optimal solution. If the effect of previous velocities was not weakened by this weight, then such a grouping would be impossible due to the fluctuations in particles updated positions in case of large inertia components. In other words, the algorithm favors an exploitation of the global optimal solution neighborhood instead of further exploration [18].

Constriction factor updating method also achieves these performances using a factor coefficient called "constriction factor" as specified in (2.24).

$$\boldsymbol{v_i(t+1) = \chi \times \{\ v_i(t) + c_1 r_1[P_{best,i}(t) - x_i(t)] + c_2 r_2[G_{best}(t) - x_i(t)]\ \}} \quad (2.24)$$

χ is the constriction factor, and it satisfies:

$$\chi = \frac{2}{|2 - \sqrt{\varphi^2 - 4\varphi} - \varphi|} \quad (2.25)$$

Where:

$\varphi = c_1 + c_2$ and $\varphi > 4$.

It is worth notice that this factor not only affects the inertia but also the cognitive and social components. Hence, we can expect to have some sort of extra control ability on the algorithm performance using this method instead of the previous one. Actually, this is what has been concluded in while limiting the maximum velocity range to X_{max}, the maximum diameter of the hyperspace, on each dimension. Furthermore, Clerc has used $\varphi = 4.1$ corresponding to $\chi = 0.729$ to provide good performance [19].

Constriction factor is actually a special case of inertia weight method. Indeed, inertia weight elements can be forced to satisfy some conditions to make it equivalent to constriction factor. This is achieved if we set $w = \chi$ along with $c_1 + c_2 > 4$, and $c_1 + c_2 = \varphi$.

2.4.2.4 Position update

Firstly, we calculate the particle's velocity using equation (2.29), then the new position of the ith particle in the search space at iteration t+1 is determined by the following equation:

$$\boldsymbol{x_i}^{\boldsymbol{t+1}} = \boldsymbol{x_i}^{\boldsymbol{t}} + \boldsymbol{v_i}^{\boldsymbol{t}}+1. \qquad (2.26)$$

Figure 2.7 shows how the velocity and position updated in PSO algorithm**.**

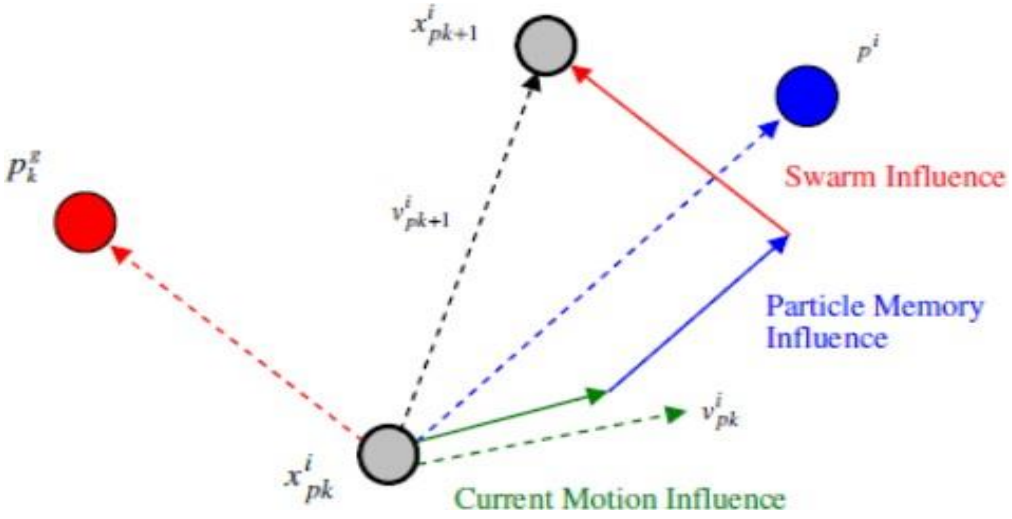


Fig 2.7 Deception of position and velocity updates

2.4.2.5 Stopping criteria

Stopping criteria are needed to finish the execution of optimization algorithm, it is provided with conditions that decides the end of operation. Below are listed the most common condition used in PSO [20] :

• Using a Maximum number of function evaluation.

• Swarm normalized radius is approximately zero.

The normal swarm radius is defined by:

$$\text{Rnorm(t)} = \frac{\mathbf{R\ max(t)}}{\boldsymbol{initial\ swarm\ diameter}}$$

Where:

$R\ norm(t)$ is the normal swarm radius at iteration t.

$R\ max(t)$ is the maximum radius of the swarm at iteration t, and is evaluated as shown in this equation.

$$\boldsymbol{R\ max(t)} = \| \boldsymbol{xm(t)} - \boldsymbol{Gbest(t)} \|$$

2.5 Grey Wolf Optimization (GWO) algorithm

This algorithm is inspired by the social behavior of grey wolves and it works on leadership hierarchy hunting strategy. Grey wolves are considered as the top-level predators; they live in a group size of 5–12 wolves. Based on the hunting strategy the grey wolves are classified into four categories such as alpha, beta, delta, and omega. Alpha wolves are leader of the bundle. This wolf has the authority to make decision for sleeping place, hunting, and so on. These wolves are otherwise called dominant wolves and they strictly instruct other wolves to follow his/her orders. An alpha wolf plays a major role in producing new solutions. Secondly, Beta wolves are second level of wolves next to the alpha wolves. These wolves are assistant wolves that guide the alpha wolves in decision-making. It also has certain rights to make decision whenever alpha wolves are passed away. These wolves listen to the alpha decision and provide response to the alpha. Thirdly, delta wolves are next level wolves which are also called subordinate wolves. These wolves are belonging to the categories of elders, sentinels, hunters, scouts, and caretakers. Deltas follow the instruction of alphas and betas and they manage next level wolves named 4 Journal of Computer

Networks and Communications omega. Finally, omega is the lowest ranking wolves and play the role of scapegoat. These wolves are must follow the instructions of all other dominant wolves. Omegas are not important wolves but, in some cases, they help others from facing internal problems. [21]

2.5.1 Concept of GWO algorithm

GWO algorithm is one of the interesting algorithms due to the group hunting strategy. Grey wolf hunting is classified into three categories (i) tracking, chasing, and approaching the prey, (ii) pursuing, encircling, and harassing the prey until it stops moving, and (iii) attacking towards the prey. In GWO, symbolic representation of alpha, beta, and deltas is represented as α, β, and δ. Grey wolf optimization contributes in both the exploration and exploitation phase. Exploitation is to search optimal solution in a local search space. In grey wolf, encircling prey and attacking for prey are two exploitation phases used to explore the optimal solution in a local search space. Search for prey works as the exploration phase in which the grey wolves search for the prey in a global search space [22].

2.5.2 GWO Parameters

In encircling prey, grey wolves recognize the location of prey and encircle them. In this phase, the position vector of the prey is defined and other search agents adjust its position based on the best solution obtained. The equation of encircling prey is given below:

$$\begin{aligned} \vec{D} &= |\vec{C} \cdot \vec{X_p}(k) - \vec{X}(k)| \\ \vec{X}(k+1) &= \vec{X_p}(k) - \vec{A} \cdot \vec{D} \end{aligned} \tag{2.27}$$

where k represents the current iteration, $\vec{A}$ and $\vec{C}$ are coefficient vectors, position vector of the prey is represented as $\vec{X}p$, $\vec{X}$ is the position vector, $||$ is the absolute value, and $\cdot$ is an element-by-element multiplication. The vectors $\vec{A}$ and $\vec{C}$ are computed as follows:

$$\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a}$$
$$\vec{C} = 2 \cdot \vec{r} \quad (2.28)$$

In hunting phase, grey wolves are directed by alpha (α) and some contributions are also provided to beta (β) and delta (δ). Initially, the best optimum is not identifiable due to vast search space whereas in the hunting strategy the alpha is considered as the first best candidate solution, beta is the second-best candidate solution and finally delta is the third best candidate solution. In all iterations, these three solutions are saved and updated to adjust the position of the lowest ranking solution omega.[22]

The equation of hunting strategy is formulated as follows:

$$\vec{D}_\alpha = |\vec{C}_1 * \vec{X}_\alpha - \vec{X}|$$
$$\vec{D}_\beta = |\vec{C}_2 * \vec{X}_\beta - \vec{X}|, \quad (2.29)$$
$$\vec{D}_\delta = |\vec{C}_3 * \vec{X}_\delta - \vec{X}|$$

where $\vec{D}_\alpha$, $\vec{D}_\beta$, and $\vec{D}_\delta$ are the modified distance vector between the alpha, beta, and delta position to the other wolves and $\vec{C}_1$,$\vec{C}_2$, and $\vec{C}_3$ are three coefficient vector aids in adjust distance vector $\vec{X}$ position of vector of other grey wolf (omega).

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 * (\vec{D}_\alpha)$$
$$\vec{X}_2 = \vec{X}_\beta - \vec{A}_2 * (\vec{D}_\beta) \quad (2.30)$$
$$\vec{X}_3 = \vec{X}_\delta - \vec{A}_3 * (\vec{D}_\delta)$$

where $\vec{X}_1$ is an obtained new position vector using alpha position $\vec{X}\alpha$ and distance vector $\vec{D}_\alpha$, $\vec{X}_2$ denotes that new position vector obtained using beta position $\vec{X}_\beta$ and distance vector $\vec{D}_\beta$, $\vec{X}_3$ represents the new position vector computed using delta position $\vec{X}_\delta$ and distance vector $\vec{D}\delta$, and $\vec{A}_1$, $\vec{A}_2$, and $\vec{A}_3$ are three coefficient vectors.

$$\vec{X}(k+1) = \frac{\sum_{i=1}^{n} \vec{X}_i}{n} \quad (2.31)$$

where $X(k+1) \rightarrow$ is new finalized new position vector computed by average sum of all positions obtained using alpha beta and delta wolf, and n represent the three wolves' alpha, beta, and delta ($n = 3$).

2.6 Conclusion

In this chapter, three types of global optimization methods, referred to as Ant Lion Optimization (ALO),Practical Swarm Optimization (PSO), and Grey Wolf Optimization (GWO) have been presented and investigated. The performance of these stochastic search methods will be discussed and compared to one another to determine their suitability and accuracy for tuning the parameters of PID controller later on chapter 3.

Chapter 3

PID Design

3.1 Introduction

During the past decades, several control structures have been developed significantly. Nevertheless, among these controllers, the PID controller is the most widely used controller in industries since it is easy to design and carry out. Stability, design, applications and performance of the PID controllers have been widely studied in literature. Therefore, the PID controller is still a proper choice for the AVR system [23]. In this chapter, the optimal design of the PID controller is achieved by minimizing the proposed time performance criterion using ALO, PSO and GWO algorithms for the AVR system. For obtaining the better control performance and of the step response can be achieved as the minimum values. Therefore, the proposed criterion includes the time domain specifications that are the overshoot, the settling time and the steady state error as well as mean of time weighted absolute error. By using this performance criterion, the optimal PID controller parameters in the AVR system are determined through ALO, PSO and GWO algorithms. This proposed method effectively investigates a high-quality solution and improves the transient response of the AVR system.

3.2 The Proportional-Integral-Derivative (PID) Control Theory

The controller is the unit designed to create a stable closed-loop system and also achieve some pre-specified dynamic and static process performance requirements. The input to the controller unit is usually an error signal based on the difference between a desired set-point or reference signal and the actual measured output [24]. Uc (s) and E(s) are the control and tracking error signals in frequency domain, respectively.

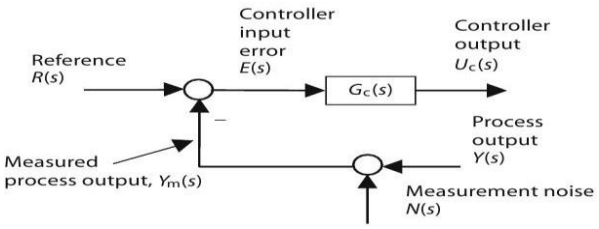


Fig 3.1 Controller Inputs and Outputs.

The actuator command developed by a PID controller is a weighted sum of the error signal and its integral and derivative. PID stands for the Proportional, Integral, and Derivative terms that sum to create the controller output signal.

$$Time\ domain\ formula: Uc(t) = K_P e(t) + K_I \int_0^t e(\tau)d\tau + K_D \frac{d}{dt} e(t) \quad (3.1)$$

$$Frequency\ domain\ formula: Uc(s) = [K_P + \frac{K_I}{s} + K_D s]\, E(s) \quad (3.2)$$

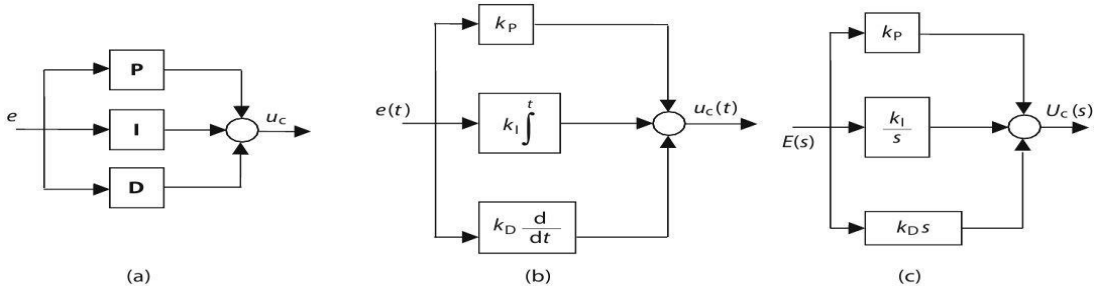


Fig 3.2 Block Diagram of the PID controller, (a) symbolic form, (b) time domain form, (c) frequency domain form

3.2.1 The Proportional Term (P)

Proportional control is denoted by the P-term in the PID controller. It used when the controller action is to be proportional to the size of the process error signal $e(t)=r(t)-y_m(t)$ and reduces error responses to disturbances. The time and Laplace domain representations for proportional control are given as:

$$\text{Time domain: } Uc(t) = K_P e(t) \quad (3.3)$$

$$\text{Frequency domain: } Uc(s) = K_P E(s) \quad (3.4)$$

where the proportional gain is denoted K_P. Figure 3.3 shows the block diagrams for proportional control.

Fig 3.3 Proportional control terme

3.2.2 The Integral Term (I)

Integral control is denoted by the I-term in the PID controller and is used when it is required that the controller correct for any steady offset from a constant reference signal value. Integral control overcomes the shortcoming of proportional control by eliminating offset without the use of excessively large controller gain. The time and Laplace domain representations for integral control are given as:

$$Time\ domain:\ Uc(t) = K_I \int_0^t e(\tau)d\tau \qquad (3.5)$$

$$Frequency\ domain:\ Uc(s) = \frac{K_I}{s}E(s) \qquad (3.6)$$

where the integral controller gain is denoted K_I.

Fig.3 4 Integral control term

3.2.3 The Derivative Term (D)

If a controller can use the rate of change of an error signal as an input, then this introduces an element of prediction into the control action. Derivative control uses the rate of change of an error signal to dampen the dynamic response and thereby improves stability of the system. It is represented by the D-term in the PID controller. The time and Laplace domain representations for derivative control are given as:

$$\text{Time domain}:\ \mathrm{Uc(t)} = \mathrm{K_D}\frac{\mathrm{d}}{\mathrm{dt}}\mathrm{e(t)} \qquad (3.7)$$

$$\text{Frequency domain}: \ Uc(s) = K_D sE(s) \qquad (3.8)$$

where the derivative control gain is denoted k_D. This particular form is termed pure derivative control, for which the block diagram representations.

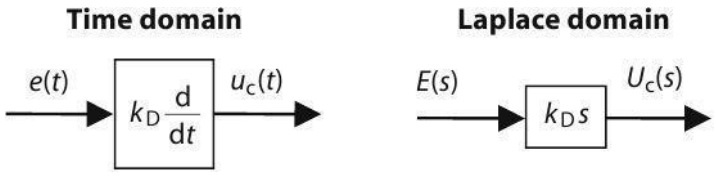


Fig 3.5 Derivative control term

3.3 Tuning of a PID Controller

This part of setting up a PID controller is to tune or choose numerical values for the PID coefficients. Many industrial process companies have in-house manuals that provide guidelines for the tuning of PID controllers for particular process plant units. Thus for simple processes it is often possible to provide rules and empirical formulae for the PID controller tuning procedure. The PID controller tuning methods are classified into two main categories

- Closed loop methods

- Open loop methods

Closed loop tuning techniques refer to methods that tune the controller during automatic state in which the plant is operating in closed loop. The open loop techniques refer to methods that tune the controller when it is in manual state and the plant operates in open loop.

Some of these closed loop tuning techniques base on the pro forma routines of the classical Ziegler–Nichols (Ziegler & Nichols 1942) methods and their numerous extensions of the associated rules. The Ziegler–Nichols methods use an on-line process experiment followed by the use of rules to calculate the numerical values of the PID coefficients. New versions of the Ziegler–Nichols (ZN) procedures were introduced, notably the Åström and Hägglund relay experiment (Åström & Hägglund, 1985).

Metaheuristic optimization methods reveled to be an extensive development for PID controller tuning and optimization of performances. In this project ALO, PSO and GWO algorithms has been proposed to tune and optimize the PID parameters.

3.4 PID Parameter Design Using ZN Rule

The Ziegler-Nichols' closed loop method is based on experiments executed on an established control loop (In our case, the AVR system) [25].

The tuning procedure is as follows:

1. PID controller is turned into a P controller by setting set Ki = 0 and Kd = 0. Initially set gain Kp= 0.

2. Kp is increased until there are sustained oscillations in the signals in the control system, e.g. in the process measurement, after an excitation of the system. (The sustained oscillations correspond to the system being on the stability limit.) This Kp value is denoted the ultimate (or critical) gain, Ku.

3. The ultimate (or critical) period Pu of the sustained oscillations is measured as shown in Figure 3.6

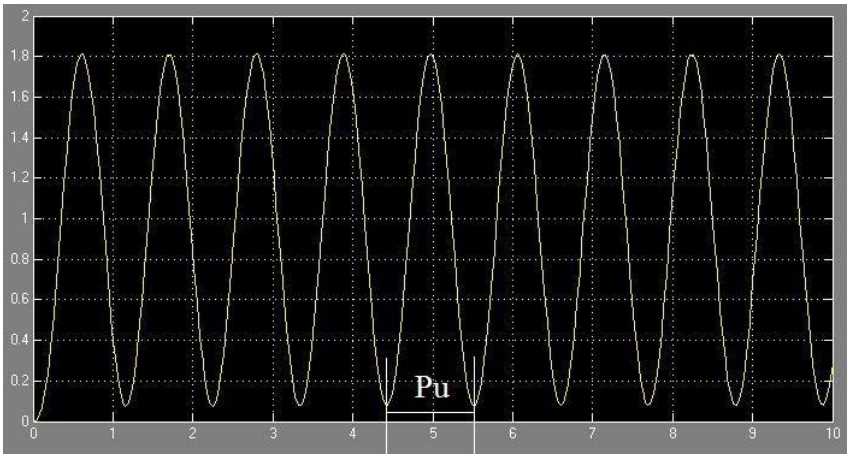


Fig.3 6 The ultimate period measurement of ZN closed loop tuning method.

5. Calculate the controller parameter values according to Table 3.1, and use these parameter values in the controller.

Table 3.1 Controller parameters of Closed-loop ZN Tuning Rule

Controller	Tuning parameters[a]		
	k_c	τ_i	τ_d
P	$k_u/2.0$	—	—
PI	$k_u/2.2$	$p_u/1.2$	—
PID	$k_u/1.7$	$p_u/2.0$	$p_u/8.0$

[a] k_u and p_u denote the ultimate gain and the ultimate period of the process respectively.

3.4.1 Simulation Results

The following Figure 3.7 shows the terminal voltage step response of the AVR system with a PID controller tuned using ZN method. By following the steps defined in the previous section, we have measured the ultimate gain and period we find that **Ku=1.70125** & **Pu=1.1**, by using the formulas given in Table 3.2 we calculated the PID controller parameters:

K_P=1.0212, **$K_i=K_P/\tau_i$**=1.0212/0.55=1.8567 and **$K_d=K_P*\tau_d$**=1.0212*0.1375=0.1404

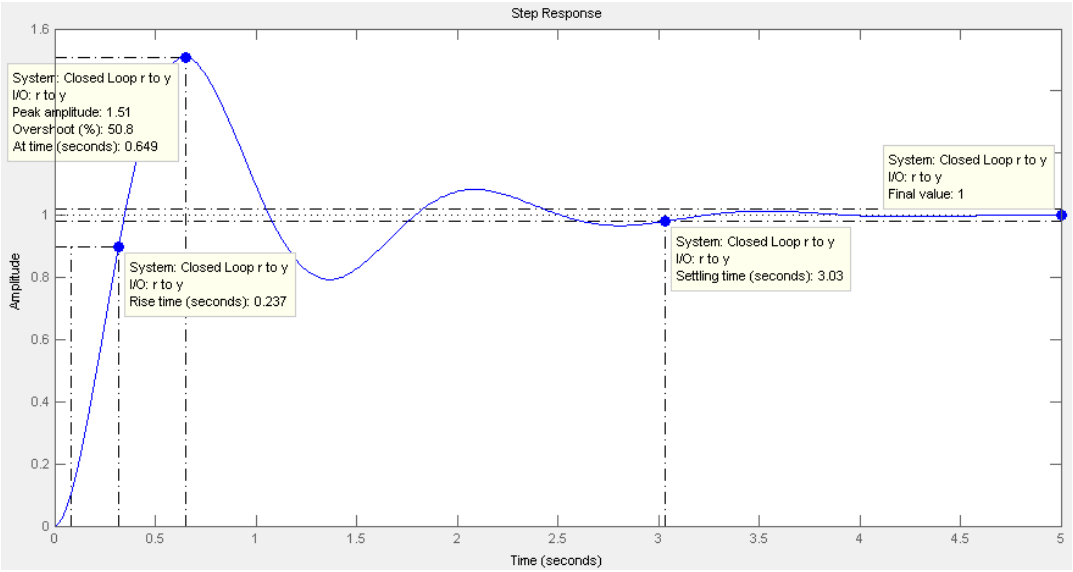


Fig 3.7 Terminal voltage step response of the AVR system with ZN tuning method

After performing the simulation of this case, we find that the performances criteria values are:

Table 3.2 Performances of ZN based controller of the AVR system

Rise time (sec)	Settling time (sec)	Overshoot %	Steady state error
0.237	3.03	50.8	0.0

3.5 Uncontrolled AVR Performance

The uncontrolled closed loop AVR system has a step response that is reported in figure 3.8.

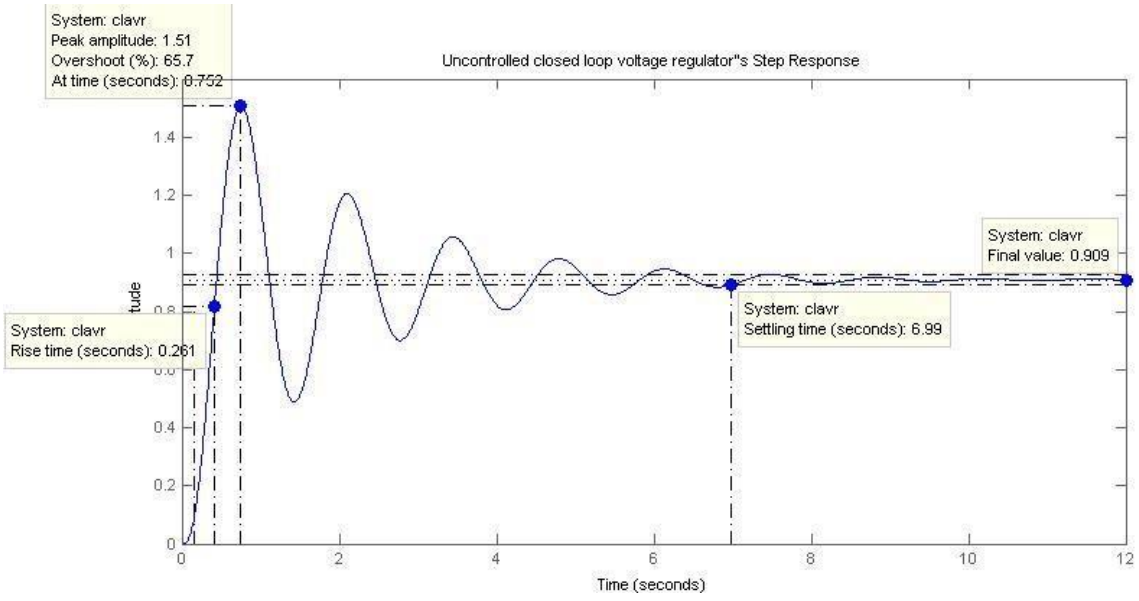


Fig 3.8 Uncontrolled AVR step response

Time performance of this uncontrolled system is shown in table 3.3.

Table 3.3 Time performance indices of the uncontrolled AVR

Rise time (s)	2% Settling time (s)	Overshoot %	Steady state error
0.261	6.99	65.7	0.091

It is evident that this performance is hardly acceptable for any of the voltage regulation systems in use now days. Not only stressing the steady state error which emphasizes a failure to ultimately catch up with the input reference; the percent overshoot will result in providing a 220V

operating apparatus, for example, with 364.54 V which can cause irrevocable damages. Finally, the transient performance is too slow to handle the quick changes in demand and supply. Hence, a three term controller is just well acquainted and required to control this system and improve its performance.

3.6 Performance Specification

Some classical and "natural" specifications are recalled for the Heaviside step response in time and frequency domain. The goal is to stress the mathematical structure of these costs and constraints that are often non-differentiable or non-analytic.

• **Rise time Tr:** This the time it takes the response to go from 10% to 90% of yss.

– Tr Measure the swiftness of the response (hence of the control system)

• **Settling time Ts:** This is the time at which the response enters and stays within ±δ of the steady-state value yss. It represents the ending of the transient phase. The used δ criteria is the 2%.

– Ts indicates the beginning of the practical steady-state response, and measures how fast the control system reaches practical steady state.

• **Peak time Tp:** The time such that: y(Tp) = Mp

• **Percentage Overshoot P.O:** The smaller the better

$$\boldsymbol{P.O} = \frac{\boldsymbol{Mp-yss}}{\boldsymbol{yss}} \times \mathbf{100}\% \tag{3.9}$$

Where: Mp is the maximum overshoot, and yss is the steady-state output (limt→∞ y(t))

• **Error e:** Which is the difference between the set-point r(t) and the output y(t), e(t)= r(t)-y(t).

• **MSE:** Mean Square Error:

$$\boldsymbol{MSE} = \frac{\mathbf{1}}{\boldsymbol{T}} \int_{\mathbf{0}}^{\boldsymbol{T}} \boldsymbol{e}^2(\boldsymbol{t})\boldsymbol{dt} \tag{3.10}$$

• **Bandwidth BW:** The frequency where the DC gain drops with 3dB.

• **Performance Criterion:** The four integral performance criteria formulas IAE, ISE, ITAE and ITSE are as follows:

$$IAE = \int_0^T |r(t) - y(t)|dt = \int_0^T |e(t)|dt \quad (3.11)$$

$$ITAE = \int_0^T t|e(t)|dt \quad (3.12)$$

$$ISE = \int_0^T |e^2(t)|dt \quad (3.13)$$

$$ITSE = \int_0^T t|e^2(t)|dt \quad (3.14)$$

3.6.1 Fitness Function

This Fitness Function (**objective function**) ***J(K)***, can be formed by different performance specifications, it includes the overshoot **Mp**, rise time **Tr,** settling time **Ts** and steady-state error **Ess** in the time domain, and bandwidth (**BW**), ISE, IAE, ITSE and ITAE in the frequency domain.

Here K is a row vector containing the control parameters, i.e. K= [Kp Ki Kd] that must satisfy the Routh-Hurwitz criterion.

To accomplish this goal and obtain best results, several Cost Functions have been selected and tested; Table 3.4 provides some of them

Table 3.4 some used fitness functions

Names	**Objective functions**
J1	**(1-exp(-0.7298))*(PO+ess) +exp(-0.7298)*(TS-TR);**
J2	**itae+itse+ise+iae+500*exp(TS-TR) +200*exp(PO-bw)+5000*exp(TS)**
J3	**mse+itae+itse+iae+ise;**
J4	**mse*itae*itse*iae*ise;**
J5	**(bw-29.3381) ^2+(TR-0.0733) ^2+(TS-0.1235) ^2 +(TP-0.2061) ^2+(PO) ^2+ess**
J6	**2*log((TS-TR)/0.1) +5*log(TS/0.2) +2*log(TR/0.1) +PO +13*((ise+iae)/(sqrt(2*pi))) *exp((itae+itse))**
J7	**100*PO+1000*(TS-TR) +(bw-29.3381) ^2+(TR-0.0733) ^2+(TS0.1235) ^2 +(TP*0.2061) ^2+PO^2+100*ess+100*(PO+50*ess)/bw+ 10*log(TP/.1) +1000*log(TS/0.01) +100*log((TS*TR)/0.1)+mse+itae+itse+iae+ise+1000*log((TS+TR+TP)/.3);**

Where:

k: vector [kp ki kd]

Po: percent overshoot

Ts: settling time

Tr: rise time

Tp: peak time

ess: steady state error

bw: bandwidth

mse: mean square error

ise: integral square error

itse: integral time square error

iae: integral absolute error

itae: integral time absolute error

3.6.2 Algorithms Settings

We have seen in the last part that selecting an appropriate cost function and tuning its parameters is vital for obtaining better control performance. Now, choosing the right algorithm is also crucial. Essentially, it is not possible to select the best search method for a given problem until the nature of that problem is well understood. Thus, the effort to identify which search method to use can be more time consuming than performing the eventual search. For this reasons, we have tested several optimization algorithms. The following tables provide some used algorithms with their corresponding setting

Table.3.5 Ant Lion Optimization (ALO) Settings

Maximum number of iterations	**200**
Number of search agents	**150**
Number of variables	**3**
random walk	**[0,1]**

Table 3.6 Particle Swarm Optimization (PSO) Settings

Iteration	**100**
Population size	**50**
Number of iteration	**100**
C1 (Personal Acc Coeff)	**2**
C2 (Social Acc Coeff)	**2**
Wdamp	**0.99**

Table 3.7 Grey Wolf Optimization (GWO) Settings

Maximum number of iterations	**500**
Number of search agents	**30**
Number of variables	**3**
Number of boundaries	**2**

3.7 Obtained results

After having performed several tests, we recorded the results in the three tables bellow table 3.8 for Antlion optimization table 3.9 for Particle swarm and 3.10 for Grey wolf optimization.

Table 3.8 Results of ALO

Cost function	Ess	Tr	Ts	Po	Kp	Ki	Kd
J1	0.0038	0.3226	0.5014	**0**	0.60048	0.41374	0.20134
J2	0.0035	0.1916	0.5583	8.9283	0.90249	0.88192	0.34726
J3	0.0032	0.0802	0.8646	33.6587	1.6502	1.7524	1.1108
J4	**0.0014**	**0.0799**	0.8506	33.8102	1.6364	1.9721	1.117
J5	0.0025	0.1384	4.1158	4.1163	0.37934	0.23785	0.62217
J6	1.5869	0.2684	0.4024	1.5869	0.67012	0.54025	0.2479
J7	0.0043	0.2495	**0.3719**	2	0.68393	0.63442	0.27278

Table 3.9 Results of PSO

Cost function	Ess	Tr	Ts	Po	Kp	Ki	Kd
J1	0.0043	0.2497	0.3822	1.9922	0.60048	0.41374	0.20134
J2	0.0048	0.2046	0.5553	6.7463	0.90249	0.88192	0.34726
J3	0.0053	**0.0797**	0.8388	33.9008	1.6502	1.7524	1.1108
J4	0.0053	0.0797	0.8388	33.9008	1.6364	1.9721	1.117
J5	0.0046	0.1881	0.5579	9.5183	0.37934	0.23785	0.62217
J6	0.0065	0.2698	0.4051	**1.4829**	0.67012	0.54025	0.2479
J7	**0.0030**	0.2523	**0.3766**	1.9728	0.6834	0.6152	0.2683

Table 3.10 Results of GWO

Cost function	Ess	Tr	Ts	Po	Kp	Ki	Kd
J1	**2.1833e-04**	0.3776	0.5910	**0**	0.54525	0.34896	0.16445
J2	0.0013	0.2014	0.6017	9.1155	0.90907	0.77473	0.31831
J3	0.0015	0.0800	0.8510	33.8100	1.6391	1.9671	1.1162
J4	0.0029	**0.0800**	0.8505	33.9028	1.6544	1.9773	1.115
J5	0.0024	0.1386	4.2445	3.9891	0.37138	0.22889	0.62232
J6	0.0093	0.2720	0.4086	1.4478	0.66384	0.53341	0.24408
J7	0.0033	0.2574	**0.3850**	1.6974	0.6746	0.59819	0.2628

From the above table, the three important parameters, namely, the Tr, Ts (which is a sign of the beginning of the steady-state phase), the Overshoot (which indicates whether or not the terminal voltage exceeding its target.), and the steady-state error have been reduced .and in the same time each cost function has own parameters however its clearly the cost function J7 has the batter results

Terminal voltage step response of the AVR controlled with the different PID controllers based on the PSO, ALO and GWO algorithms is shown in Figure 3.9. Moreover, Table 3.11 presents the best solutions found by the algorithms and the corresponding time indices based on, Ess, Tr, and Ts and Po.

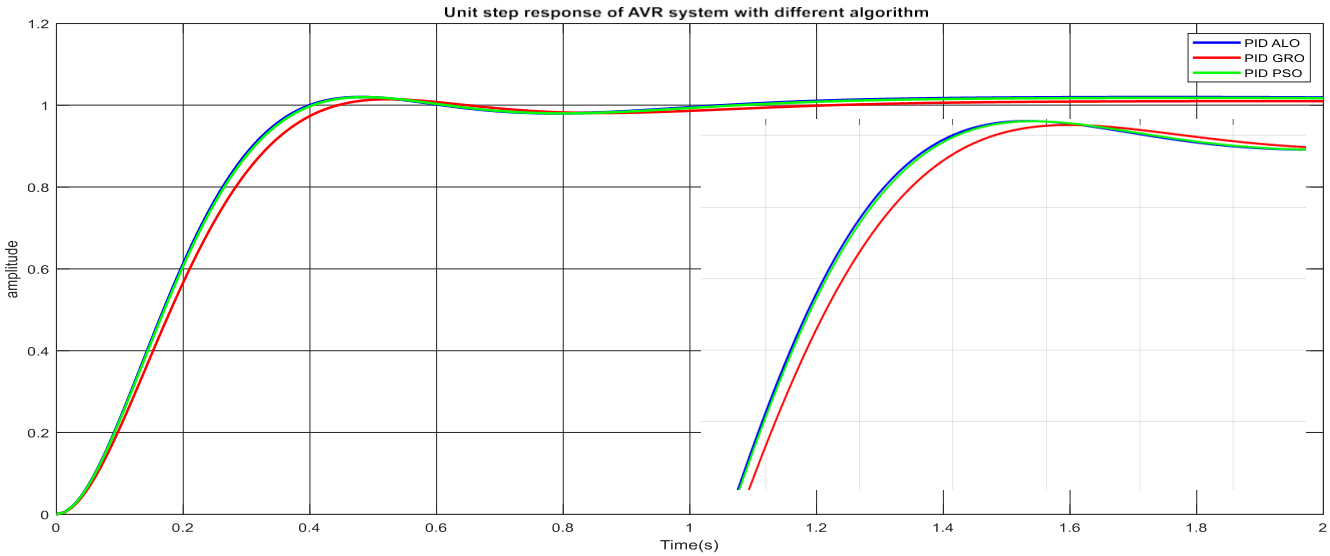


Fig 3.9 Unit step response of AVR system with different algorithm

Table 3.11 Results of the transient response and the optimum parameters of the PID

Algorithms	***PO%***	***TR***	***TS***	***Ess***	***KP***	***KI***	***KD***
ALO	2	**0.2495**	**0.3719**	0.0043	0.68393	0.63442	0.27278
PSO	1.9728	0.2523	0.3766	**0.0030**	0.6834	0.6152	0.2683
GWO	1.6974	0.2574	0.3850	**0.0033**	0.6746	0.59819	0.2628

In figure 3.9, it is clear that the unit step response is almost closer to each other for different algorithms based PID controllers. However, the PSO based PID controller has better performance for percent overshoots compared to others. On the other hand, the best performance in terms of settling time and rise time has been given by the ALO based PID controller.

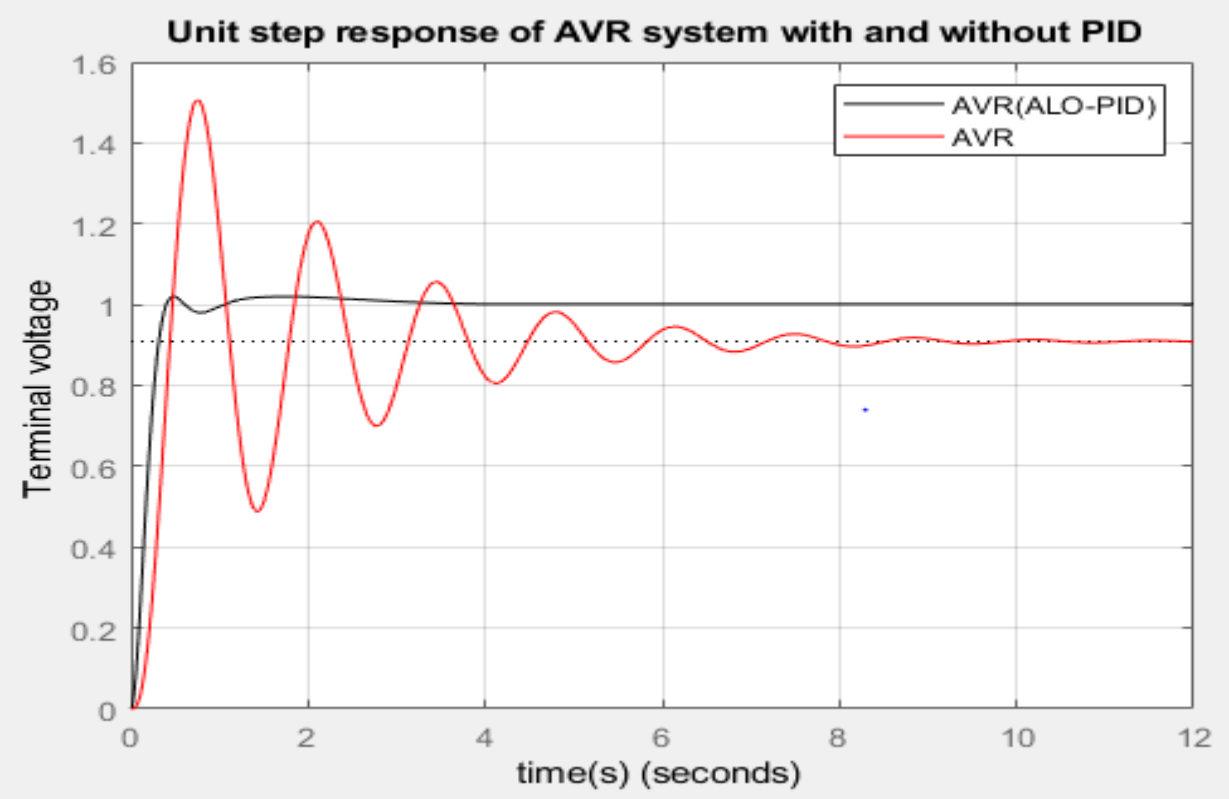


Fig 3.10 Comparison of terminal voltage step response of AVR system with optimized PID controller and without PID

we can see that our controller results are encouraging to a certain level. This implies that we have attained our objective consisting in performing the AVR performance by optimally tuning its PID controller

3.8 Conclusion

In order to achieve a better dynamic performance, improved robustness and stability, a new time domain performance criterion was proposed in tuning process of the PID controller for the AVR system. The proposed objective function was used in the ALO, PSO and GWO algorithms for optimum design of the PID controller parameters in the same AVR system. As can be seen from the simulation results obtained here, the proposed gain tuning method improved the system's dynamic response parameters such as maximum overshoot, rise time, settling time and steady state error. Moreover, the performance of the AVR with the proposed ALO based PID controller was compared to the PSO based PID and GWO based PID controllers. The comparative simulation results show that the proposed ALO based PID approach was capable of obtaining a superior response performance as compared to the other optimized PID controllers with various objective functions. Furthermore, the performance of the proposed PID controller as well as the other optimized PID controllers was tested under dynamic parameter uncertainties of the AVR system. It is found that more robust stability and better dynamic performance characteristics were obtained by the proposed ALO based PID controller compared with other PID controllers under the condition of the parameter uncertainty.

Chapter 4

FOPID Design

4.1 Introduction

Fractional calculus extends the ordinary calculus by extending the ordinary differential equations to fractional order differential equations, i.e. those having non-integer powers of derivatives and integrals. Such equations can provide linear models and transfer functions for some infinite dimensional physical systems [26]. On the other hand, fractional order controllers may be employed to achieve feedback control objectives for such systems. Modeling and control topics using the concept of fractional order systems have been recently attracting more attentions because the advancements in computation power allow simulation and implementation of such systems with adequate precision. Fractional Order PID (FOPID) controller is a convenient fractional order structure that has been employed for control purposes [27]- [28]. An FOPID is characterized by five parameters: the proportional gain, the integrating gain, the derivative gain, the integrating order and the derivative order.

4.2 Why Fractional Order

We have seen, in undergraduate courses of feedback control, the basic control actions and their effects in the controlled system behaviour in both time and frequency domains. We know that these actions are proportional, derivative and integral. Their main effects over the controlled system behaviour are [29]:

• To increase the speed of the response, and to decrease the steady-state error and relative stability for proportional action.

• To increase the relative stability and the sensitivity to noise, for derivative action.

• To eliminate the steady-state error and to decrease the relative stability for integral action.

However, some of these actions have undesired behaviors. The following table summarizes the pros and cons.

[2] It is important to remark that “fractional,” (F) or “fractional-order,” (FO) are improperly used words. A more accurate term should be “non-integer-order,” since the order itself can be irrational as well. However, a tremendousamount of work in the literature use “fractional” more generally to refer to the same concept. For this reason, we are using the term “FPID or FOPID

Table 4.1 Positive and Negative effects of Derivative and Integral actions

	positive effects	negative effects
derivative action	increased relative stability (the $\pi/2$ phase lead introduced)	increased sensitivity to high-frequency noise (by the increasing gain with slope of 20 dB/dec)
integral action	elimination of the steady-state errors (the infinite gain at zero frequency)	decreased relative stability (the $\pi/2$ phase lag introduced)

Considering this, we can conclude that we could achieve more satisfactory compromises between negative and positive effects by introducing more general control actions of the form s^n, $1/s^n$ where $n \in \mathbb{R}^+$. and combining the actions could develop more powerful and flexible design methods to satisfy the controlled system specifications.

4.3 Fractional Calculus:

4.3.1 Definitions:

Fractional calculus is a generalization of the ordinary calculus. The main idea is to develop a functional operator *D*, associated to an order *v* not restricted to integer numbers, that generalizes the usual notions of derivative (for a positive *v*) and integral (for a negative *v*). Just as there are several alternative definitions for the usual integer integrals (according to Riemann, Lebesgue, Steltjes, etc.), there are also several different definitions for fractional derivatives. The most common definition is due to Riemann and Liouville [30] that generalizes the two following definitions corresponding to integer orders:

$$ {}_{c}D_{x}^{-n} f(x) = \int_{c}^{x} \frac{(x-t)^{n-1}}{(n-1)!} f(t)dt,\ n \in \mathbb{N} \tag{4.1} $$

$$ D^n D^m f(x) = D^{n+m} f(x), m \in \mathbb{Z}^-_0 \text{ and } n, m \in \mathbb{N}_0 \tag{4.2} $$

The Laplace transform of D follows the well-known rules:

$$ L[{}_{0}D_{x}^{v} f(x)] = \begin{cases} s^v F(s) & \text{if } v \le 0 \\ s^v F(s) - \sum_{k=0}^{n-1} s^k \, {}_{0}D_{x}^{v-k-1} f(0) & \text{if } n-1 < v < n \in \mathbb{N} \end{cases} \tag{4.3} $$

This means that, if zero initial conditions are assumed, the systems with dynamic behaviour described by differential equations involving fractional derivatives give rise to transfer functions with fractional powers of *s*.

4.3.2 Integer Order Approximations:

The most usual way of making use, both in simulations and hardware implementations, of transfer functions involving fractional powers of *s* is to approximate them with usual (integer order) transfer functions. One of the best-known approximations is due to Manabe and Oustaloup who use the recursive distribution of poles and zeroes. The approximating transfer function is given by [31]:

$$ s^v \approx k \prod_{n=1}^{N} \frac{1+(s/\omega_{z,n})}{1+(s/\omega_{p,n})}, v > 0 \tag{4.4} $$

The approximation is valid in the frequency range [ω_l, ω_h]. Gain *k* is also adjusted so that both sides of (4.4) shall have unit gain at 1 rad/s. The number of poles and zeros (*N*) is chosen beforehand, and the desired performance of the approximation strongly depends thereon: low values result in simpler approximations, but may cause ripples in both gain and phase behaviours. Such ripples can be practically eliminated by increasing *N*, but the approximation will become computationally heavier. Frequencies of poles and zeroes in (4.4) are given by:

$$ \begin{aligned} &\omega_{z,1} = \omega_l \sqrt{\eta}, \\ &\omega_{p,n} = \omega_{z,n} \varepsilon; n = 1 \dots N, \\ &\omega_{z,n+1} = \omega_{p,n} \eta; n = 1 \dots N-1, \\ &\varepsilon = (\omega_h/\omega_l)^{v/N} \\ &\eta = (\omega_h/\omega_l)^{(1-v)/N} \end{aligned} \tag{4.5} $$

The case $v<0$ can be dealt with by inverting (4.4). For $|v|>1$, the approximation becomes unsatisfactory. So it is usual to split the fractional powers of s as follows

$$s^v = s^n s^\delta, v = n + \delta, n \in \mathbb{Z}, \delta \in [0,1] \quad (4.6)$$

and only the latter term, i.e. s^δ, needs to be approximated.

4.4 Fractional Order PID Controller:

The most common form of fractional PID controller is $PI\text{-}\lambda D_\mu$, the generalized transfer function of this controller is given by:

$$G_C(s) = K_P + \frac{K_1}{S^\lambda} + K_D S^\mu \text{ with } \lambda, \mu \geq 0 \quad (4.7)$$

The selection of λ and μ in equation (4-7) gives the following classical controllers:

• for λ=1 and μ=1,then it's the classical PID controller.

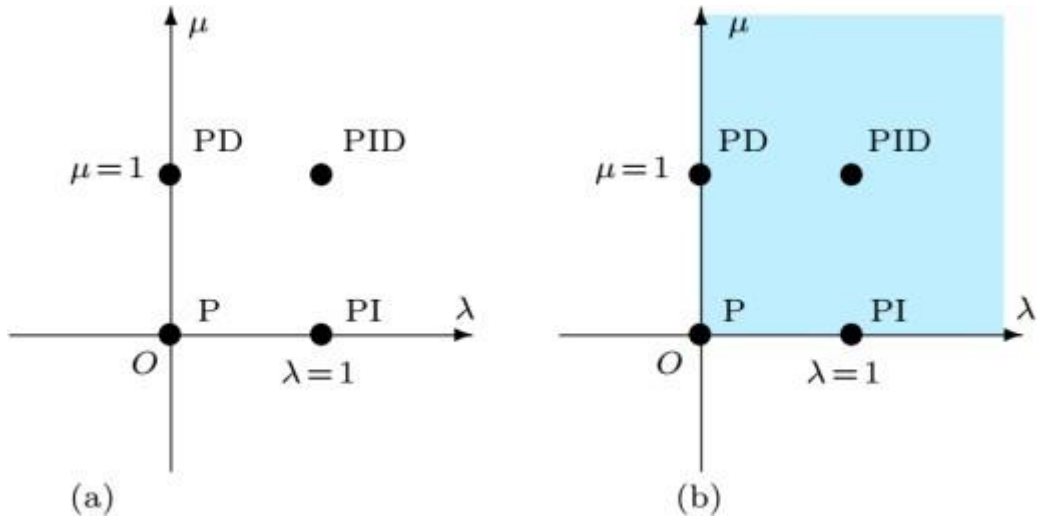
• for λ=0 and μ=1,then it's the classical PD controller.

• for λ=1 and μ=0,then it's the classical PI controller.

• for λ=0 and μ=0,then it's the classical P controller.

This point out that, the $FOPID$ controller is an extension to the conventional PID controller; Moreover, all the classical controllers are a special case of fractional PID controller. In a graphical way, the control possibilities using a fractional-order PID controller are shown in Figure 4.1, extending the four control points of the classical PID to the range of control points of the quarter-plane defined by selecting the values of λ and μ.

The figure below shows clearly how to obtain the different pid controller



(a) integer-order (b)fractional order

Fig 4.1 Fractional-order PID vs classical PID from points to plane

In general, the values of λ and μ are between 0 and 2.

Let us conduct a discussion about the fractional integral and derivative control actions to see how powerful the FPID is.

4.4.1 Integral Action

In contrast to the classical PID Controller, the FPID controller gives more freedom to the integral action of type $K/s\lambda$ for $\lambda \in [0, 2]$ in all different domains. For example, the displacement of the root locus of the system towards the right half-plane, the amount of the increment in the slope of the magnitude, and the decrement amount in the phase plot are all controlled by fractional-order integral value λ. This means that the selection of the value of λ needs consideration of the effects mentioned above [32].

4.4.2 Derivative Action

Following a procedure similar to that for the integral action, it is easy to prove that all the effects of derivative actions mentioned in the conventional PID can be weighted by the selection of the order of the derivative action, that is, μ ∈ [0, 2]. In the frequency domain, for example, by varying the value of μ between zero and one, it is possible:

• To introduce a constant increment in the slopes of the magnitude curve that varies between

0dB/dec and 20 dB/dec.

• To introduce a constant delay in the phase plot that varies between 0 rad and π/2 rad.

As we have seen, introducing fractional orders for the integrator and differentiator components of the PID controller has potential benefits due to providing more degrees of freedom. It is expected that the fractional PID controllers will replace their classical, integer-order variants in industrial control applications [33]. Tuning and auto-tuning techniques play a significant role in accelerating this process.

4.5 FOPID Tuning:

4.5.1 FOPID controller specifications:

The fractional-order PID controller is defined as a generalisation of the standard PID controller in equation 4.7.

The parameter set to optimize consists of fractional PID parameters:

$$\theta = [K_p \quad K_i \quad K_d \quad \lambda \quad \mu] \tag{4.8}$$

We have chosen the upper and lower bounds of the decision variables, according to the part (4.4), to be 5 and 0 for [Kp Ki Kd], 2 and 0 for [$\boldsymbol{\lambda}$ $\boldsymbol{\mu}$] respectively. This means that, our search space is a five-dimension.

We have used the GL definition-based numerical method designed in Matlab by [34] to model the fractional order system. Since this method is a fixed step method, we needed to choose a small enough step size to ensure proper simulation results.

After writing the code of the optimization problem in Matlab Software, we have noticed the following: by using a small enough step, say h = 0.0001, the execution time was so big, in a way that it takes more than five hours to complete just one run of the program. For the sake of reasonable processing speed, we have done some research, and finally, we come up with a solution of using the Oustalop filter approximation to approximate the integral and derivative terms in the FOPID transfer function. approximation is valid. However, the problem with this approximation is that currently there is no method available for determining suitable values of w_l, w_h and N [35]. Considering this, we have used a designed Matlab function in [34] that uses a default lower frequency bound of w_l = 0.001

rad/sec, a high frequency bound of w_h =1000 rad/sec, and a default order of approximation of N = 5.

4.5.2 The Proposed Tuning Algorithm:

Metaheuristic optimization methods revealed to be an extensive development for FOPID controller tuning and optimization of performances. In this section, since we have found the ALO algorithm has better results, have been proposed to tune and optimize the FOPID parameters. And we will compare it with results of PID and other papers.

4.5.3 The Proposed Objective Functions for FOPID:

Because we are aiming to obtain a good transient performance of the AVR system, many fitness functionshave been explored in our simulations. (as we have seen in chapter 3)

In Optimization and control theory, one first should choose which control objective is to be achieved. Our objective is to have an optimum response that is characterized by:

- Fastest rising and settling time
- Smallest steady state error and percentage overshoot.

To accomplish this goal and obtain best results, several Cost Functions have been selected and tested; Table 4.2 provides some of them.

Table 4.2 cost functions

Names	Functions
J1	IAE
J2	ISE
J3	$ITAE$
J4	$ITSE$
J5	(100*PO+1000*Log(Ts/0.01)+ 100*Log((Ts-Tr)/0.1)
J6	$ISE + IAE + ITSE + ITAE;$
J7	PO + Ess^2 + Ts + Tr + Tp + PU

4.6 Simulation results

In the previous chapters, the necessary theory on the plant, the FOPID controller, and the optimization algorithms were introduced. This chapter is devoted to present simulation results related to optimization of FOPID being applied in the AVR system by ALO. We will provide some discussions about the importance of the selection of algorithms and fitness functions to solve a given optimization problem. Next, we will address the best-obtained results. A brief comparison with some published work.

It should be noted that most of the experiments have been implemented by using MATLAB software based on FOMCON toolbox [34] on a 3.10 GHz PC withprocessor i5-6200U and 4GB RAM.

4.7 Obtained results

After having performed several tests, we recorded the results for each cost function in table below:

Table 4.3 Performances of different objective function

Cost function	Ess	Tr	Ts	Po	Kp	Ki	Kd	λ	μ
J1	1.0024e-04	0.0398	0.5831	9.1017	1.1287	3.4165	0.97374	0.14566	1.5024
J2	0.0181	0.0078	1.6277	32.9078	3.4569	2.9852	2 .9852	0.97123	1.831
J3	0.0053	0.0471	0.6587	18.7685	3.1269	2.178	1.3936	0.71218	1.2931
J4	0.0035	0.0301	0.4679	18.0586	3.0417	3.4214	0.1693	1.5315	1.4677
J5	**0.002**	**0.0322**	**0.2760**	**6.4870**	**3.9616**	**2.4417**	**1.1663**	**1.1018**	**1.0814**
J6	0.0053	0.0376	0.2558	14.2625	3.4989	2 .8027	1.1302	0.05028	1.4656
J7	0.0023	0.0508	0.8354	0.5622	3.0706	0.52942	0.64813	1.8597	1.5683

From the above table, the important parameters, Tr, PO, Ts and Ess have been reduced and at the same time each cost function has own parameters however its clearly the cost function **J5** has the better results. Terminal voltage step response of the AVR controlled with FOPID using cost function **J5** is shown in Figure 4.2.

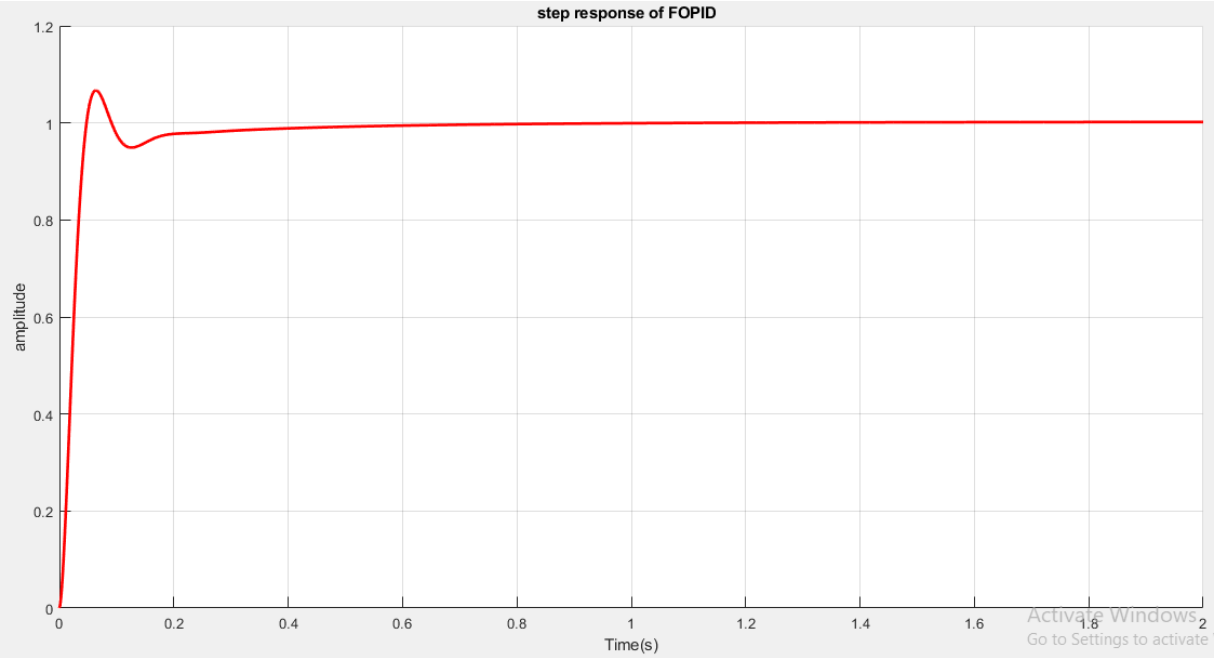


Fig 4.2 Terminal voltage step response of the AVR controlled with FOPID

4.7.1 Comparison with ALO-PID Controller

The voltage unit step response of the AVR system with FOPID and PID controllers is shown in Fig. 4.3 It is obvious from this simulation that AVR system with FOPID has a very high efficiency.

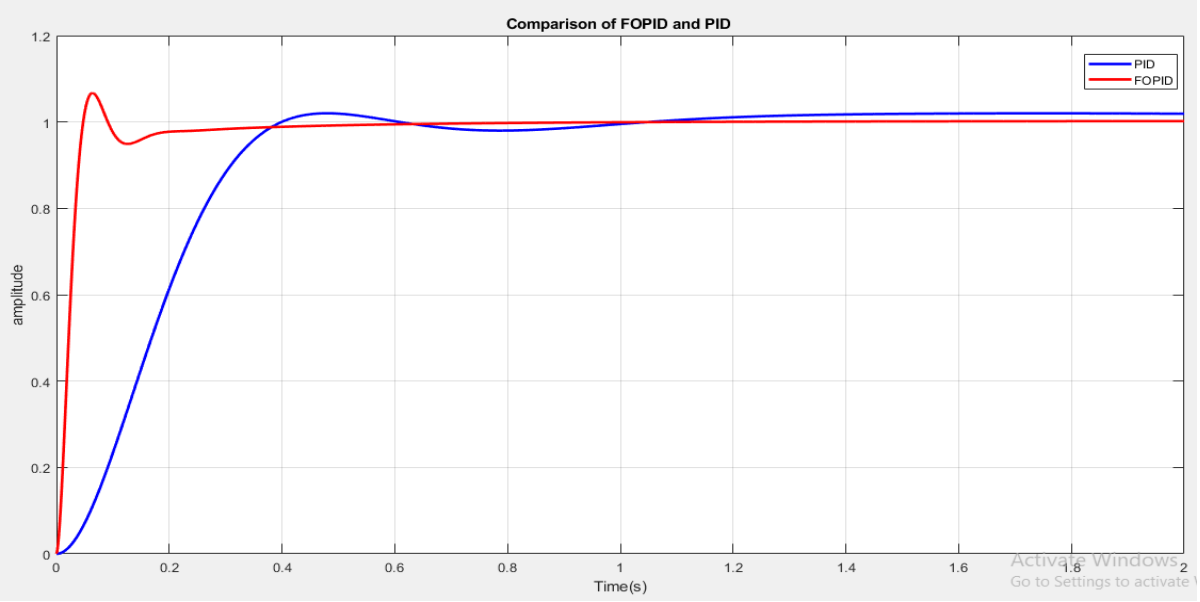


Fig 4.3 Terminal voltage step response of AVR system controlled by ALO-FOPID, and ALO-PID

Table 4.4 Improvement of the performances

	AVR system controlled by FOPID	*AVR system controlled by PID*	*Improvement %*
Tr	**0.0322**	**0.2495**	***87.09***
Ts	***0.276***	**0.3719**	***25.79***
Po	***6.487***	**2**	
Ess	**0.002**	**0.0043**	**53.49**

4.7.2 Stability check

Using the FOMCON toolbox [34] that is based on overload programming technique to enable the related methods of the Matlab built-in functions to deal with Fractional Order models, and by using **fotf()** and **feedback()** functions, the closed loop transfer function of the AVR system with the ALO-FOPID controller is generated:

$$G(s) = \frac{0.11663s^{\{3.1832\}} + 11.663s^{\{2.1832\}} + 0.39616s^{\{2.1018\}} + 39.616s^{\{1.1018\}} + 0.24417s + 24.417}{4e-05s^{\{5.1018\}} + 0.00814s^{\{4.1018\}} + 0.4281s^{\{3.1018\}} + 11.663s^{\{2.1018\}} + 40.616s^{\{1.1018\}} + 24.417}$$

Using K=isstable (), the stability analysis of the MATLAB code gives that K=1. Hence the system is stable.

Bode frequency response of fractional dynamic systems is shown in the figure 4.4

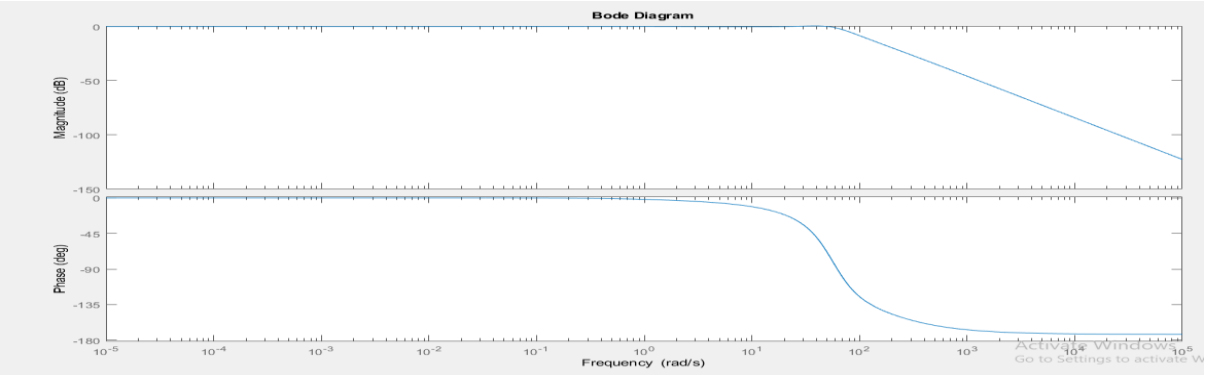


Fig 4.4 Bode diagram

4.7.3 Comparison with other FOPID Controllers:

In this section, we compare with other FOPID controllers. We have selected designed fractional controller available in [37] and [36]. In [37] the FOPID is designed based on Ant Lion Optimizer Algorithm (ALO); while in [36] the controller is designed based on Genetic Algorithm (GA).

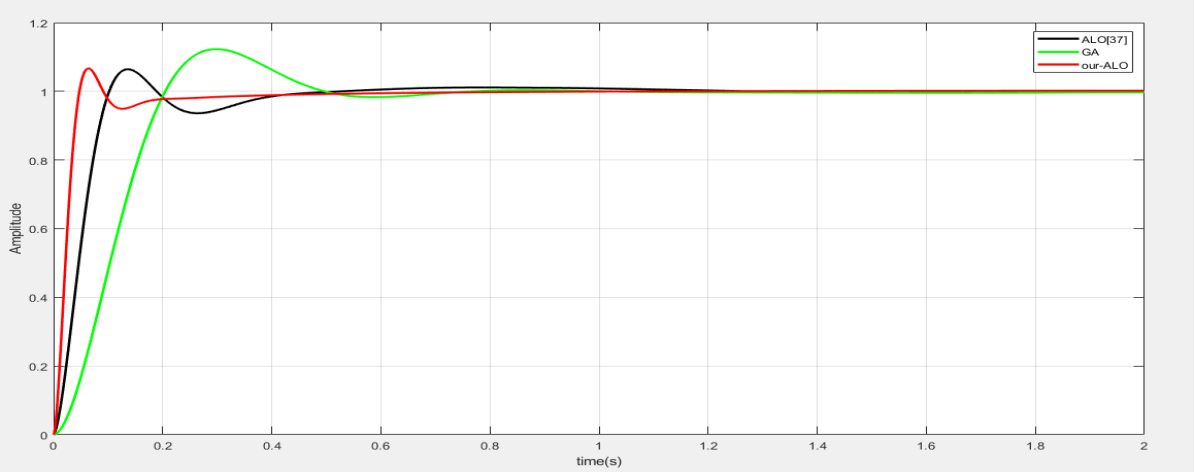


Fig 4.5 Terminal voltage step response of AVR system controlled by proposed ALO-FOPID,ALO-FOPID and GA-FOPID

The step responses of the AVR system controlled by proposed ALO-FOPID controller, ALO-FOPID controller and GA-FOPID controller are shown in Figure 4.5 . The performance indices are shown in Table 4.5. Figure 4.5 and Table 4.5 allow to say that the performances of proposed ALO-FOPID and those of ALO-FOPID are closer to each other, this difference is due to cost function used. However, they are better than GA-FOPID interms of all four performance indices.

Table 4.5 Best performance indices of different FOPID controllers

	Tr 2%	Ts 2%	PO 2%	Ess	Kp	Ki	Kd	λ	μ
proposed - ALO-FOPID	0.0322	0.276	6.487	0.002	3.9616	2.4417	1.1663	1.1018	1.0814
ALO-FOPID	0.0697	0.3737	6.6792	0.0023	3.407	1.0468	0.5568	1.2244	1.3918
GA-FOPID	0.1393	0.4623	12.5983	0.0027	1.6947	0.8849	0.3964	1.0284	1.1296

4.7.4 Robustness test

The voltage response characteristic of the AVR system for the different five sets of described values of $\boldsymbol{KG}$ and $\boldsymbol{\tau G}$ is shown in Fig4.6 Thus, ALO based FOPID controller has the ability to control the AVR system despite variation in the generator's parameters

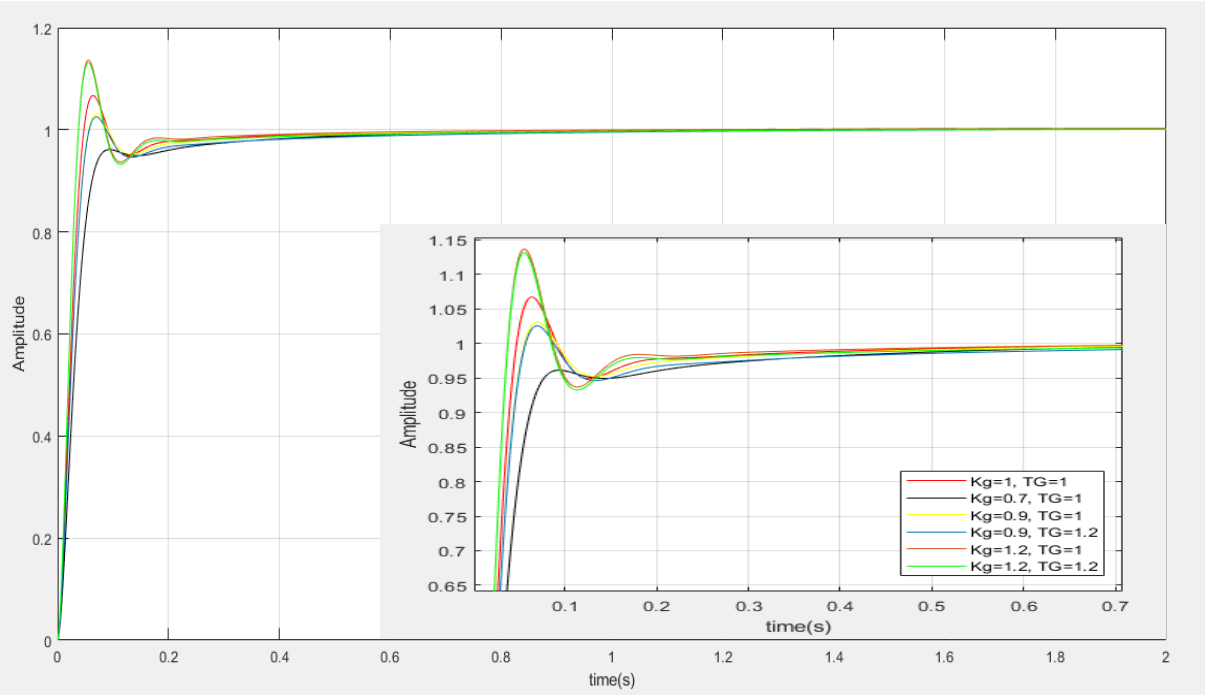


Fig 4.6 Voltage response profiles of the AVR system using

4.8 Conclusion

Fractional calculus can provide novel and higher performance extension for FOPID controllers. However, the difficulties of designing FOPID controllers increase, because FOPID controllers also take into consideration the derivative order and integral order in comparison with traditional PID controllers.

In this chapter, a novel optimization method for designing fractional order PID controllers based on the ant-Lion Optimization is used. The proposed method determines the parameters of FOPID controller through minimizing an objective function. the ALO-FOPID controller is also robust to model uncertainties.

General Conclusion

This project proposes the novel optimization algorithm in order to optimize the PID and FOPID controllers parameters in the AVR system. The proposed algorithm presents the compound of the Ant Lion Algorithm (ALO), Particle Swarm Algorithm (PSO) and Grey Wolf Algorithm (GWO). It is proved in the project that such an approach significantly accelerates the convergence of the algorithm. Furthermore, to determine optimal FOPID controller parameters, a new objective function is presented. The results obtained by applying the proposed algorithm with the new objective function introduced in this project provide significantly better voltage response of the AVR system compared to other considered algorithms. The robustness of the AVR system with such an obtained FOPID controller is tested by changing the AVR system's parameters. It is shown that in all examined cases, the step response of the AVR system has extremely small deviations compared to the nominal case, which means the system is robust to the uncertainties in the system. The mutual comparison shows that the ALO-FOPID controller is by far the best in rejecting all considered types of disturbances.

We think that in this way, the algorithm is improved, no matter what optimization problem is considered. In this project, we tested its applicability and efficiency on the problem of optimal FOPID design.