Chapter One

Artificial Intelligence Optimization Techniques

1.1 Introduction

From an engineering perspective, the description of artificial intelligence (AI) may be summarized as: "the study of representation and search through which intelligent activity can be enacted on a mechanical device". This perspective has dominated the origins and growth of AI. The first modern workshop/conference for AI practitioners was held at Dartmouth College in the summer of 1956, [George F Luger,2005]

According to), [D.A. Linkens, 1996]intelligent control shows high performance control over a wide range of operating conditions(e.g. parameters uncertainties) . It is defined as systems that have the ability to emulate human capabilities (planning, learning and adaptation).

Unlike conventional control, intelligent control uses tow sources of information (learning from process and designer/skilled operator knowledge) to form the corresponding relationship between inputs and outputs. The most widely used intelligent control schemes are fizzy logic control (FLC), artificial neural networks (ANN). These techniques are used in the field of electrical drives for control process, estimation, system identification and optimization problems however genetic algorithms (GA's) [Eldissouki, 2002] and particle swarm optimization are used to solving optimization problems.

There are two large electrical drive manufactures, which incorporate AI into their drives there are Hitachi and Yaskawa. In addition to this, Texas Instruments (TI) has built a fuzzy controlled induction motor drive using the TMS320C DSP. The main conclusion obtained by TI agrees with results obtained from various fuzzy-control implementations: the development time of the fuzzy-controlled drive is significantly less than the corresponding development time of the drive using classical controllers.SGS Thomson has also built some DC and AC drives incorporating fuzzy logic control, [P.Vas et al, 1996].

In all drives, but especially in electrical vehicles, energy is a crucial factor. However, by using AI it may be possible to improve the efficiency.

This chapter introduces the trends of intelligent control techniques and its application to AC drives for efficiency optimization. The study focuses on fuzzy control, genetic algorithms and PSO technique.

1.2 Artificial Neural Networks

Artificial Neural network (ANN) resembles human brain in learning through training and data storage. They can approximate complicated functions using several layers of neurons structured in a way similar to the human brain; so, ANN acts as a universal approximator. ANN has learning capability and generalization property. Because of its learning capability, ANN is very powerful in control applications where the dynamics of a plant or process control is partially known or the mathematical representation is very complicated. The generalization property is very useful because it allows training of the neural networks with a limited training data set.

Artificial neural network consists of a number of interconnected information-processing elements called neurons. It has certain performance characteristics in common with the biological neural networks, **[**Bose, 2006**].** A neuron can be modeled to perform a mathematical function such as a pure linear function, step function, tan-sigmoid function etc. These neurons can be interconnected to establish a variety of network architecture. The attractive feature of the neural network is that it can be trained to solve complex nonlinear functions with variable parameters, which may not be attainable by conventional mathematical tools, [B.Kosko,1992]

1.3 Particle Swarm Optimization (PSO)

Particle Swarm Optimization is an evolutionary computation technique introduced in 1995; its idea is based on simulation of social behavior of animals such as bird flocks or fish schools searching for food. PSO is another form of evolutionary computation it is population-based method, like genetic Algorithm. However, the basic concept is cooperation instead of competition. It is also very similar to GA, but it does not have genetic operators.

In this algorithm, each individual is referred to as a particle and presents a candidate solution to the optimization problem. Unlike other population-based methodologies, every agent moves along its velocity vector, which is updated using two different best experiences; one is the best experience, which a particle has gained itself during the search procedure and the other is the best experience gained by the whole group. Combination of these experiences can provide useful information for each particle to explore new positions in the domain of the problem.

1.3.1. Original PSO Algorithm

The basic PSO algorithm consists of the velocity and position equations

For each particle i, the velocity and the position of particles can be updated by the following equations, [Taher Niknam,2010]:

$$\begin{cases} V_i\,(k+1) = V_i\,(k) + rand1i\big(p_i - x_i(k)\big) + rand2i\big(G - x_i(k)\big) & (1.1) \\ x_i\,(k+1) = x_i\,(k) + v_i\,(k+1) & (1.2) \end{cases}$$

Where *i* is the index of each particle, *k* is the discret time index, *rand1* and *rand2* are random numbers between 0 and 1.

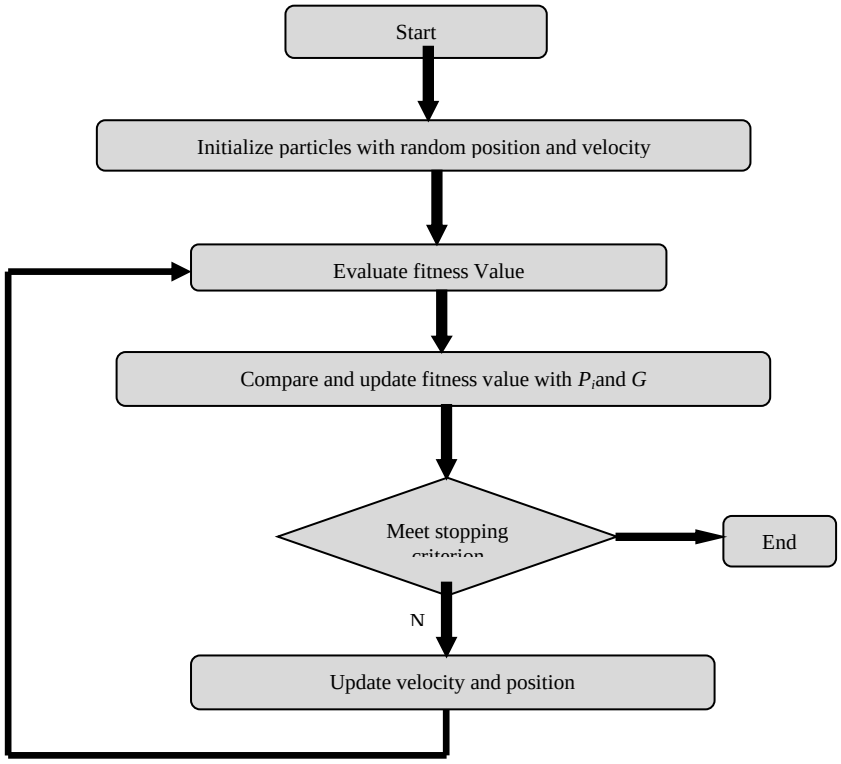
P_i is. The best position found by ith particle, *G* is the global best particle among the entire population.

The most used PSO algorithm form is including an inertia term and acceleration constants as, [Brian .Brige,2003] :

$$V_i\,(k+1) = \emptyset_i(k)\,V_i\,(k) + \alpha_1[rand1i\big(p_i - x_i(k)\big)] + \alpha_2[rand2i\big(G - x_i(k)\big)] \qquad (1.3)$$

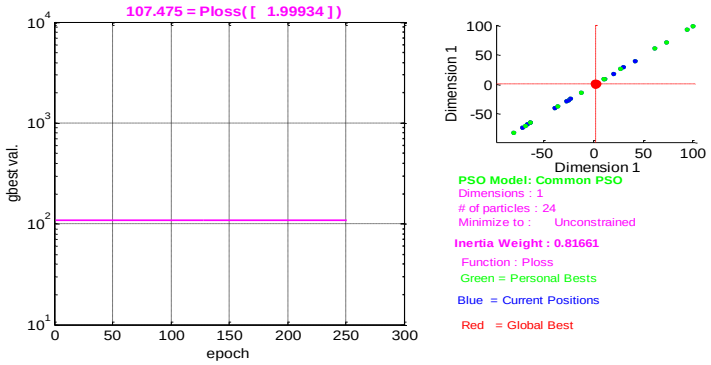
ϕ is an inertia function and $\alpha_{1,2}$ are acceleration constants.

PSO optimization procedure is established according to the flowchart shown in Figure 1.1..



Fig(1.1). PSO Flowchart

A simple example would be find the minimum of the loss function given in chapter three for a given load T_l=25% T_{IN}. A particle swarm optimization toolbox developed by [Brian Brige,2003] was used obtained result is shown in Fig (1.2) bellow:



Fig(1.2). Visualization of PSO process

1.4 Fuzzy Logic System

Fuzzy logic systems (FLSs) are else a universal function approximators. The heart of fuzzy logic system is linguistic rule-base, which can be interpreted as the rules of a single "overall" expert, or as the rule of "subexperts" and there is a mechanism (inference mechanism), where all the rules are considered in an appropriate manner to generate the output,[Zadeh,1965],[Mamdani and Assilian,1975],[Lee,1990] and [Kusko,1992]

Fuzzy logic control has found many applications. This is so largely employed because this fuzzy logic control has the capability to control non-linear, uncertain systems even in the case where no mathematical model is available for the controlled system.

The application of fuzzy logic has some advantages:

- When parameters change
- When existing traditional controllers must be augmented or replaced (e.g. to provide self-tuning, to give more flexibility of controller adaptability, etc...).
- If sensor accuracy (or price) is a problem (fuzzy logic can handle imprecise measurements, uncertainties)
- To obtain solutions when solutions are not possible by using other technique.

1.4.1 Conventional and Fuzzy Sets

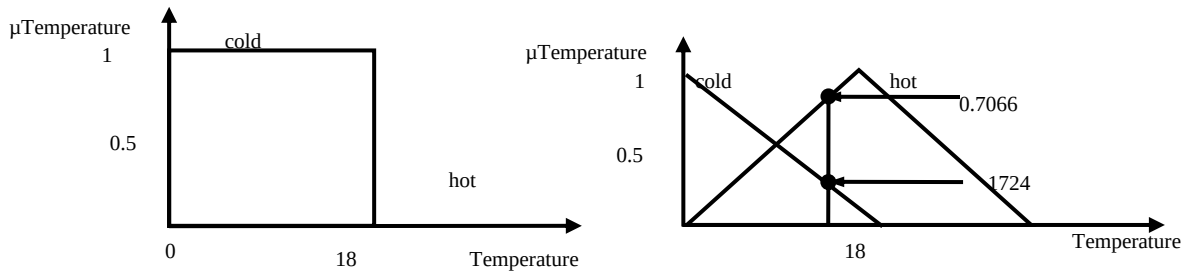
Fuzzy set theory resembles human reasoning in its use of approximate information and uncertainty to generate decisions. It was specifically designed to mathematically represent uncertainty and vagueness and provide formalized tools for dealing with the imprecision intrinsic to many problems.

Within the framework of classical logic, a proposition is either true or false (1 or 0).to clarify this concept the example bellow is used:

For example, classical logic can easily divide the temperature of a room into two subsets, "less than 18° and 18° or more than 18°. Figure (1.2). (a) Shows the result of this partition. All temperatures less than 18 are then considered as belonging to the set "less than 18". They assign a value of 1 and all temperatures reaching 18 or more are not considering as belonging to the set "less than 18". They are assigned a value of 0.

However, human reasoning is often based on knowledge or inaccurate, uncertain or imprecise data. A person placed in a room in which the temperature is either 17.95° or 18.05°, will certainly distinguish between these two temperature values. This person will be able to tell if the piece is "cold "or "hot" without accurate temperature indication.

Fuzzy logic is used to define subsets, such as "cold" or "hot" by introducing the possibility to a value belonging more or less to each of these subsets.



(a) Two sets according to classic logic (b) Two sets according to fuzzy logic

Fig (1.2). Temperature classification of a room in two sets

In fuzzy set theory, the concept of characteristic function is extended into a more generalized form, known as *membership function*: $\mu A(x)$: $U \rightarrow [0, 1]$. While a characteristic function exists in a two-element set of {0, 1}, a membership function can take up any value between the unit interval [0, 1]. The set which is defined by this extended membership function is called a *fuzzy set*. In contrast, a classical set which is defined by the two-element characteristic function, is called a *crisp set*, [M.N Cirstea et al, 2002]

According to the definition above a fuzzy set from Fig (1.2) can be defined as follows. Let *U* be a set, called the Universe of Discourse and u be a generic element of *U* ($u \in U$). A fuzzy set *A* in a universe of discourse *U* is a function that maps *U* into the interval [0, 1]. The fuzzy set *A* is characterized by a membership function (MF) $\mu_A(x)$ that takes values in the interval [0, 1].

1.4.2. Linguistic Variables and Values

Words are constantly used to describe variables in human's daily life. Similarly, words are used in fuzzy rules to formulate control strategies. Referring to the above example, words like "room temperature is hot" can be used to describe the state of a system (in the current case, it is the state of the room). In this example, the words "*cold*" and "*hot*" are used to describe the variable "temperature". This means that the words "*cold*" and "*hot*" are the values of the fuzzy variable "temperature". Note that the variable "temperature" in its turn, can also take crisp values, such as 18°, 15.6°, 0°, etc.

If a variable is assigned some crisp values, then it can be formulated by a well established mathematical framework. When a variable takes words as its values instead of crisp values, there is no formal framework to formulate it in the classical mathematical theory. The concepts of Linguistic Variable and Value were introduced to provide such a formal framework. According to these concepts, if a variable can take words in natural languages as its values, then that variable is called *Linguistic Variable*. The words that describe the value of that linguistic variable are defined by fuzzy sets in the universe of discourse in which the variable is defined [L.-X. Wang, 1997]. These words are called *Linguistic Values.*

In general a linguistic variable is characterized by (1) a name, (2) a term, and (3) a universe of discourse. For example on Figure 1.1. (b), the variable "temperature" is a linguistic variable with 2 linguistic values, namely "*cold"* and "*hot*". The variable "temperature" can be characterized in the universe of discourse U = [-18°, +18°], corresponding to minimum and maximum temperature of the room, respectively. The linguistic values "*cold*" and "*hot*" can be characterized by the fussy sets described in Figure (1.2. b) or by any other set (depending on the application and the designer's choice).

These definitions show that linguistic variables are the necessary tools to formulate vague (ill-defined) descriptions in natural languages in accurate mathematical terms. They constitute the first step to incorporate human knowledge into engineering systems in a systematic and efficient manner, [L.-X. Wang, 1997].

1.4.3. Membership Functions (MFs)

There are many other choices or shapes of MFs besides the ones described in Figure (1.2). A graphical illustration of typical and commonly used ones in literature is shown in Figure 1.3., [K.M. Passino, 1998]

The simplest and most commonly used MFs are the triangular types due to their simplicity and computation efficiency,[K.M. Passino, 1998],[Bose,2006]. A singleton is a special type of MF

that has a value of 1 at one point on the universe of discourse and zero elsewhere. The L-function and sigmoid types are mainly used to represent saturation of variables.

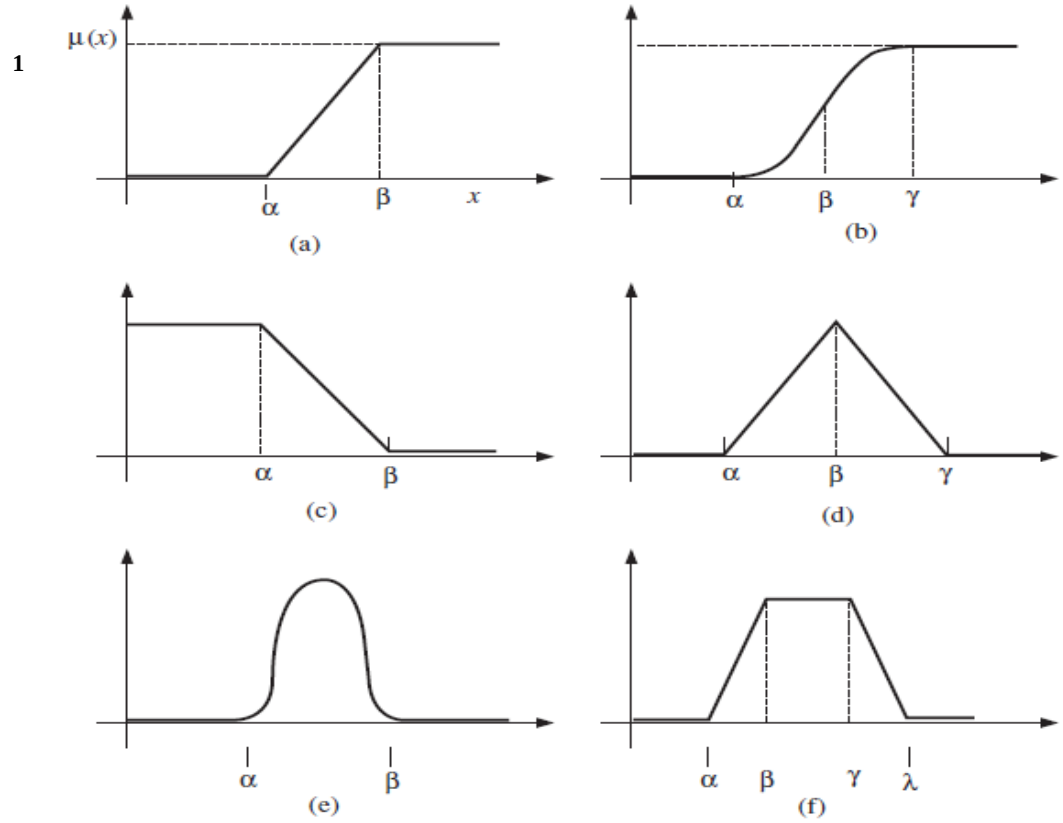


Fig. (1. 3). Typical shapes of MFs (a) ..,(b) sigmoid,(c) L function,(d) Triangular,(e) Gaussian function,(f) Trapezoidal,[M.N Cirstea, 2002]

1.4.4. Fuzzy Rules and Fuzzy Implication

A "fuzzy If-Then- rule" , also known as " fuzzy rules", "fuzzy implication", or "fuzzy conditional statement" assume the form :

If x is A then y is B

Where A and B are linguistic values defined by fuzzy sets on universes of discourse X and Y, respectively. Often "x is A" is called antecedent or "premise", while "y is B" is called the "consequence" or "conclusion".

Examples of fuzzy if-then rules are widespread in our daily linguistic expression, such as the following:

- If pressure is high, then volume is small
- If road is slippery, then driving is dangerous

- If speed id high , then apply the brake is little

The expression "if x I A the y is B", is sometimes abbreviated as $A \rightarrow B$

The procedure for assessing these influences is called "*Fuzzy Implication"*. Since fuzzy propositions and relations are expressed by MFs, fuzzy implications also imply MFs as a method of interpretation.

In literature, there are a number of implication methods. The frequently used ones are [BK.Bose 2002], [L.-X. Wang, 1997], [Z. Kovacic, 2006]:

1) Zadeh implication,

2) Mamdani implication,

3) Godel, implication,

4) Lukasiewicz implication,

5) Sugeno implication,

6) Larsen implication, etc

The differences between these methods are summarized in [Ajit.K.mandal.; 2006], [S.N.Sivanandam,2007], [Kwang.H.Lee,2005]. Their mathematical functions indicate that the Mamdani implication is the most suitable for hardware implementation, [K.M. Passino, 1998]. It is also the most commonly used in control system applications.

1.5. Fuzzy Logic Controller (FLC)

Usually a control strategy and a controller itself is synthesized on the base of mathematical models of the object or process under control. The models of an object under control involve quantitative, numeric calculations and commonly are constructed in advance, before realization. Since fuzzy logic control is based on human knowledge and experience, it doesn't need an exact mathematical model, it is an automatic control strategy based on "IF-THEN" rules.

The FLC can be viewed as a step toward a rapprochement between conventional precise mathematical control and human-like decision making.

The principal structure of a fuzzy controller is illustrated in Figure 1.3.. It consists of normalization factors, fuzzification of inputs, inference or rule firing, defuzzification of outputs, and denormalization.

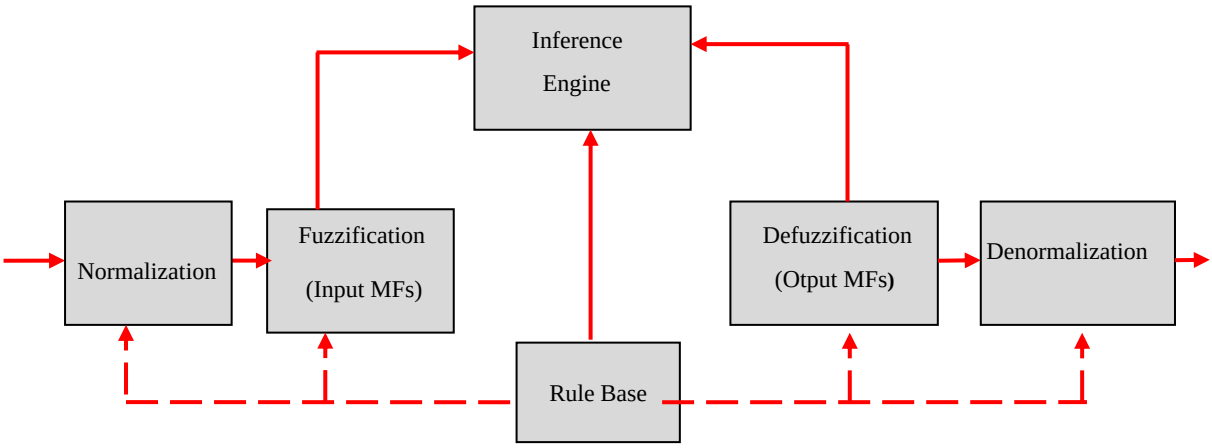


Fig. (1.4). Basic structure of FLC

1.5.1 Steps of Fuzzy Logic Controller Design

- Initially choose the number of inputs/outputs
- Fuzzify the real inputs using appropriate membership functions
- Create the IF THEN rules using AND/OR operator
- Defuzzify the output fuzzy to get the corresponding crisp output

1.5.2. Fuzzification Interface

The fuzzification interface transforms input crisp values into fuzzy values and it involves the following functions,[Kwang H. Lee,2005].

- Receives the input values
- Transforms the range of values of input variable into corresponding universe of discourse
- Converts input data into suitable linguistic values (fuzzy sets).

1.5.3. Rule Base

Although differential equations are the language of conventional control, the dynamic behavior of a system is characterized by a set of linguistic descriptions in terms of fuzzy rules in FLCs Fuzzy rules serve to describe the quantitative relationship between the input and the output variables in linguistic terms such that, instead of developing a mathematical model that describes a system, a knowledge-based system is used.

Fuzzy rules are the core of the FLC. Generally the dynamic behavior of a fuzzy controller is characterized by a group of fuzzy rules which follows the format:

If antecedents Then consequence.

The antecedents can be joined by union (OR) or by intersection (AND). In the speed control of induction motor for example typical rule reads as:

If "*the speed error"* is positive small (PS) AND "*change in speed error"* is negative small (NS) THEN *u* is negative small (NS).

1.5.4. Inference Engine

The function of the inference engine provides a way to translate the input of a fuzzy set into the fuzzy output. It determines the extent to which each rule is relevant to the current situations as characterized by inputs. The inference engine is the decision-making logic of an FLC. It has the capability of simulating human decision-making based on fuzzy concepts and inferring fuzzy control actions using fuzzy implication and the rule of inference in FL.

Normally this mechanism consists of set of logic operations.

There are several ways to implement a fuzzy inference: the Mamdani fuzzy reference system and the Sugeno reference system are two commonly used.[Hung T. Nguyen.,2003]

1.5.5. Defuzzification Inference

The result of implication and aggregation steps in the inference engine is a fuzzy output. This output is the union of all the outputs of individual rules that are validated [Bose 2002]. The conversion of this fuzzy output set to a single crisp value (or a set of crisp values) is referred to as *Defuzzification*. Hence, this latter interface generates the output control variables as a numeric value.

Defuzzification can be implemented in different ways, general methods include MOM (mean of maximum), COA (center of area), and COM (center of maximum), [Hung T. Nguyen., 2003], [Bose, 2002].

1.6 Genetic Algorithms

Genetic algorithms (GAs) are global optimization techniques developed by John Holland in 1975. They are perhaps the most widely known type in the family of evolutionary algorithms,[Mitsuo. Gen, 2000]. These algorithms search for solutions to optimization problems by "evolving" better and better solutions. A genetic algorithm begins with a "population" of solutions and then chooses "parents" to reproduce. During reproduction, each parent is copied, and then parents may combine in an analog to natural cross breeding, or the copies may be modified, in an analog to genetic mutation. The new solutions are evaluated and added to the population, and low-quality solutions are deleted from the population to make room for new solutions. As this process of parent selection, copying, crossbreeding, and mutation is repeated, the members of the population tend to get better. When the algorithm is halted, the best member of the current population is taken as the global solution to the problem posed,[Mitsuo.,Gen 2000], [Kwang,Y. Lee,2008];[Goldberg,1989].

There has been widespread interest from the control community in applying the genetic algorithm (GA) to problems in control systems engineering. Compared to traditional search and optimization procedures, such as calculus-based and enumerative strategies, the GA is robust, global and generally more straightforward to apply in situations where there is little or no a priori knowledge about the process to be controlled. As the GA does not require derivative information or a formal initial estimate of the solution region and because of the stochastic nature of the search mechanism, the GA is capable of searching the entire solution space with more likelihood of finding the global optimum.

The process of evolution is based on the following principles:

- Individuals in a population compete for resources and mates.
- The most successful individuals in each generation will have a chance to produce more offspring than those individuals that perform poorly.
- Genes from 'good' individuals propagate throughout the population so that two good parents will sometimes produce offspring that are better than either parent. Thus each successive generation will become more suited to their environment.

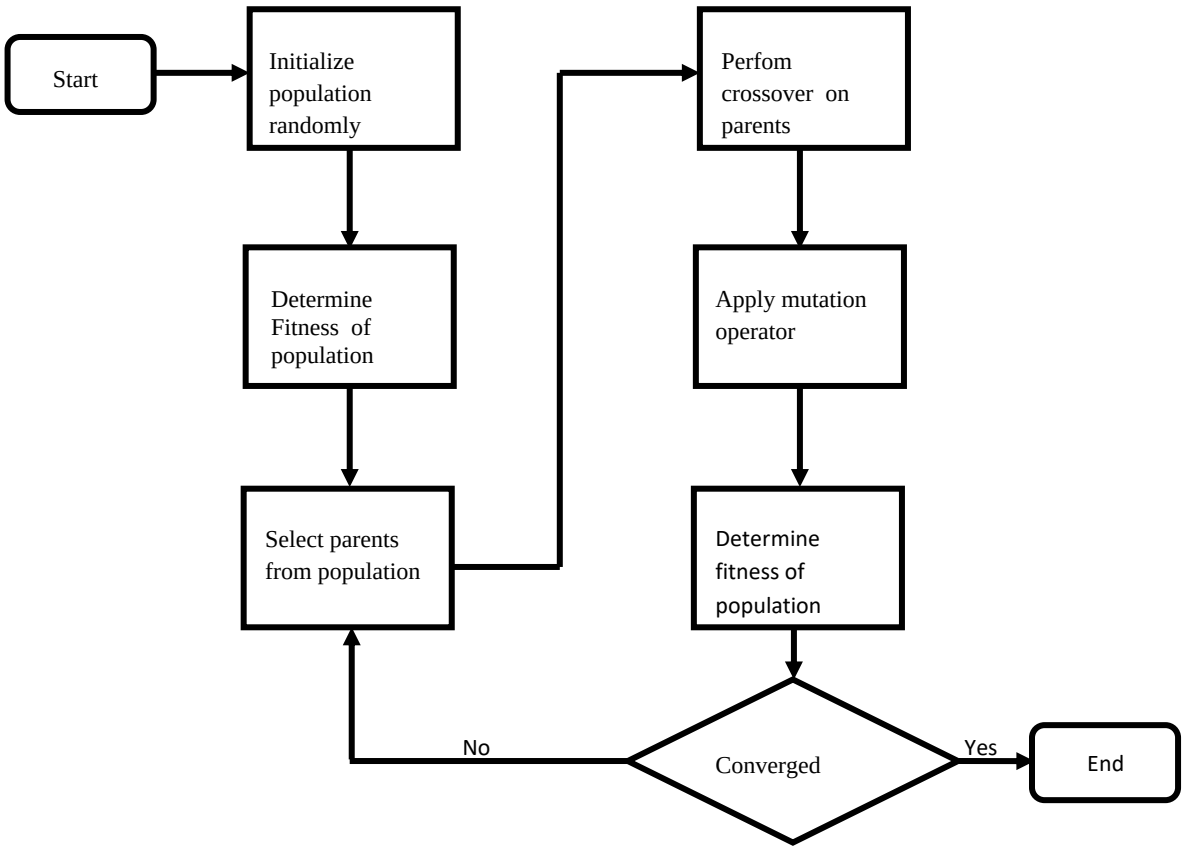
1.6.1 Implementation Details

A population of individuals is maintained within search space for a GA, each representing a possible solution to a given problem. Each individual is coded as a finite length vector of characters. A fitness value is assigned to each solution representing the ability of an individual

to 'compete'. The goal is to produce an individual with the fitness value close to the optimal. By combining information from the chromosomes, selective 'breeding' of individuals is utilized to produce 'offspring' better than the parents. Continuous improvement of average fitness value from generation to generation is achieved by using the genetic operators. The basic genetic operators are:

- Selection: used to achieve the survival of the fittest.
- Crossover: used for mating between individuals.
- Mutation: used to introduce random modifications.

The genetic operators are used in the GAs optimization procedure according to the flowchart given in Figure (1.5).



Fig(1.5). Genetic algorithms flowchart

1.6.1.1. Selection

The selection mechanism favors the individuals with high fitness values. It allows these individuals better chance for reproduction into the next generation while reducing the

reproduction ability of least fitted members of population. Fitness of an individual is usually determined by an objective function.

1.6.1.2. Crossover

The crossover operator divides a population into pairs of individuals and performs recombination of their genes with a certain probability. If one-point crossover is performed, as shown in Figure (1.6)., one position in the individual genetic code is chosen. All gene entries after that position are exchanged among individuals. The newly formed offspring created from this mating are put into the next generation. Recombination can be done at many points, so that multiple portions of good individuals are recombined, this process is likely to create even better individuals. The crossover operator roughly mimics biological recombination between two single−chromosome (haploid) organisms.

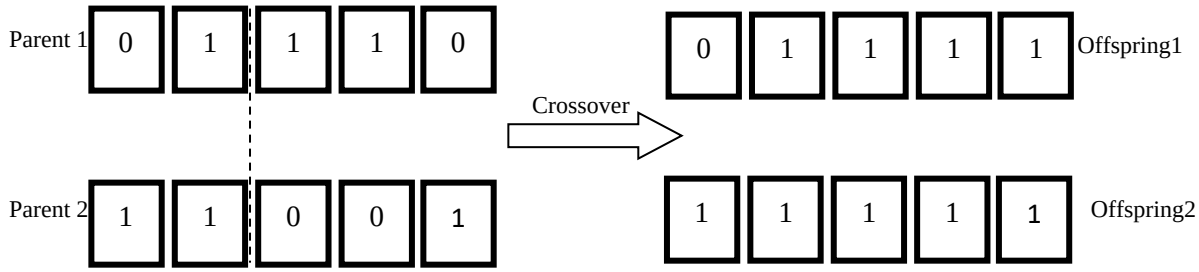


Fig (1.6). One-point crossover example

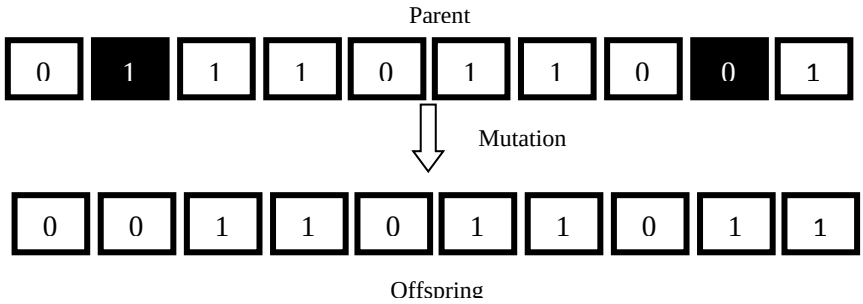
1.6.1.3. Mutation

When using mutation operator a portion of the new individuals will have some of their bits flipped with a predefined probability. In Figure (1.7). Mutation operator is applied to the shaded genes of the parent. The purpose of mutation is to maintain diversity within the population and prevent premature convergence. The usage of this operator allows the search of some regions of the search space which would be otherwise unreachable.

The described operators are basic operators used when the individuals are encoded using binary alphabet. Operators for real valued coding scheme were developed by Michalewicz [xx]. The following operators are defined: uniform mutation, non-uniform mutation, multi-non-uniform mutation, boundary mutation, simple crossover, arithmetic crossover and heuristic crossover.

- Uniform mutation randomly selects one individual and sets it equal to an uniform random number.

- Boundary mutation randomly selects one individual and sets it equal to either its lower or upper bound.
- Non-uniform mutation randomly selects one variable and sets it equal to a non uniform random number.
- Multi-non-uniform mutation operator applies the non-uniform operator to all of the individuals in the current generation.
- Real-valued simple crossover is identical to the binary version.
- Arithmetic crossover produces two complimentary linear combinations of the parents.
- Heuristic crossover produces a linear extrapolation of the two individuals.



Fig(1.7). Mutation example

As an example of application of the two intelligent techniques used in this thesis an optimal fuzzy controller based on GA is developed in next section.

1.7. Induction Motor Speed Regulation

This section will be dedicated to investigate the speed regulation of IM using both techniques focused in this chapter mainly fuzzy logic and genetic algorithms, in the first step an FLC is used to regulate motor speed than it is performances are optimized by using GA'.

1.7.1 Speed Control using FLC

The components of the FLC will be introduced by using speed control of induction motor problem. Fig (1.9). depicts the typical response to step consign. One can see there are four zones:A1/ rise, A2**/**overtake,A3/ damping and A4/ steady state regions.

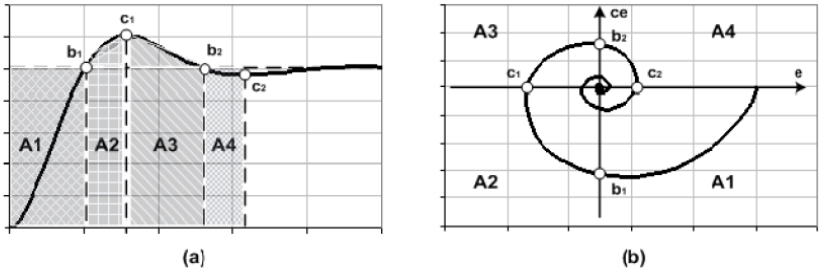


Fig. (1.8). System step response

Most closed-loop speed control systems react to the error (*e (t)*) between the reference speed and the output speed of the motor. When controlling processes, human operators usually compare the actual output of the system with the desired (reference) output and observe the evolution of this difference. This is why in most FLCs, including the controllers proposed in the input variables are the system error, *e(t),* and the change-in-error, *Ce(t),* to complete the initial description of the investigated speed control closed loop, let *u*(*t*) be the FLC output variable, i.e. the process input signal (which consist of the torque current component namely I_{sq}) .

As the first step is the fuzzification hence it:

(1) Measures the values of the input variables (*e(t) and Ce(t)* for the presented example,

(2) Performs a scale mapping of the measured crisp values of the input variables into the universes of discourse of these input variables, and

(3) Converts the input values into linguistic values compatible with the fuzzy set representation in the rule base. The three operations are performed as follows. Just as (*e(t)* and/or *Ce(t)*) take on values of, for example 0.2p.u at time instant *t*, linguistic variables also assume linguistic values at every time instant *t*. The values that linguistic variables take on over time change dynamically.

Let's suppose, for the presented example, that *e(t)*, *Ce(t)*, and *u* take on the following values: "Negative Big" or NB, "Negative Small" or NS, "Zero" or Z, "Positive Small" or PS, and "Positive Big" or PB. The meanings of these linguistic values are quantified by their respective MFs. For close-loop speed control, each of the following statement quantifies some of different configurations of the system:

- The statement "*e(t)* is PB" can represent the situation where the output speed is significantly smaller than its reference.

- The statement "*e(t)* is NS" can represent the situation where the output speed is just slightly over the reference, but not too close to it to justify quantifying it as Z and not too far to justify quantifying it as NB.
- The statement "*e(t)* is PB" and "*Ce(t)* is PS" can represent the situation where the speed is significantly below the reference, but while "*Ce(t)* is PS", the motor speed is away from its reference value.

These statements indicate that in order to fuzzify the dynamics of a process successfully, one must first have a good understanding of the physics of the underlying process. Moreover, the accuracy of the FLC is built on the shape, the number and the distribution of linguistic values or MFs used.

Figure (1.10). Shows the fuzzy sets and the corresponding triangular MF description of each signal. The universe of discourse of all the variables, covering the whole region and all the MFs are asymmetrical because near the origin (steady state), the signal requires more precision. This completes the first step of FLCs according to Figure (1.4)

.

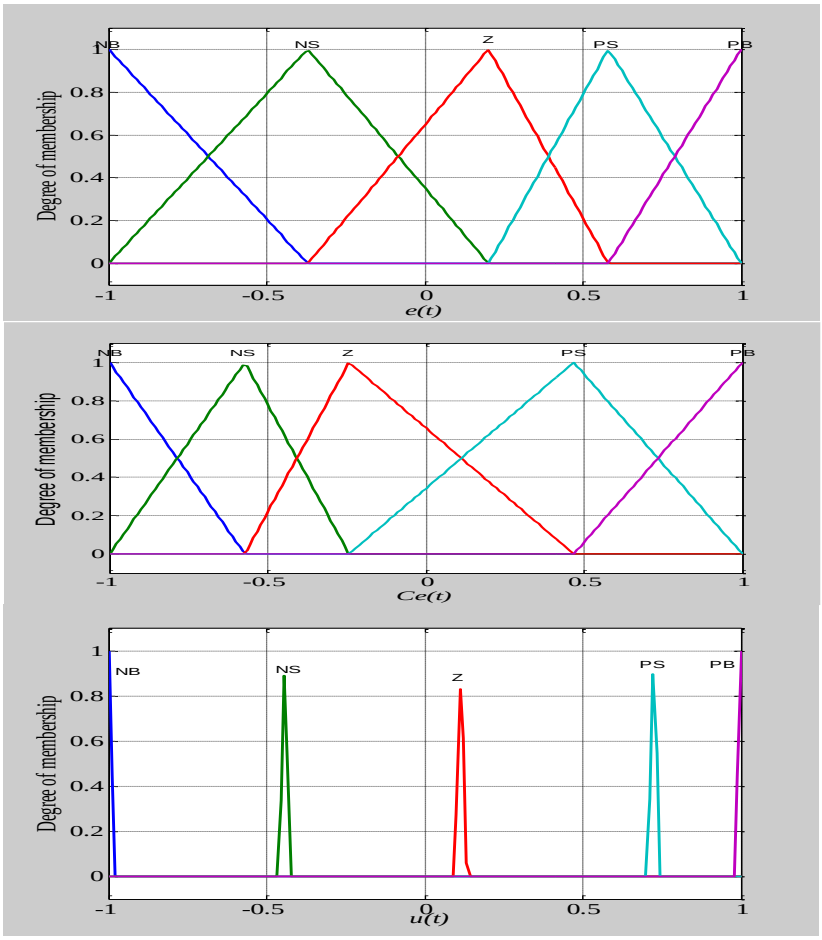
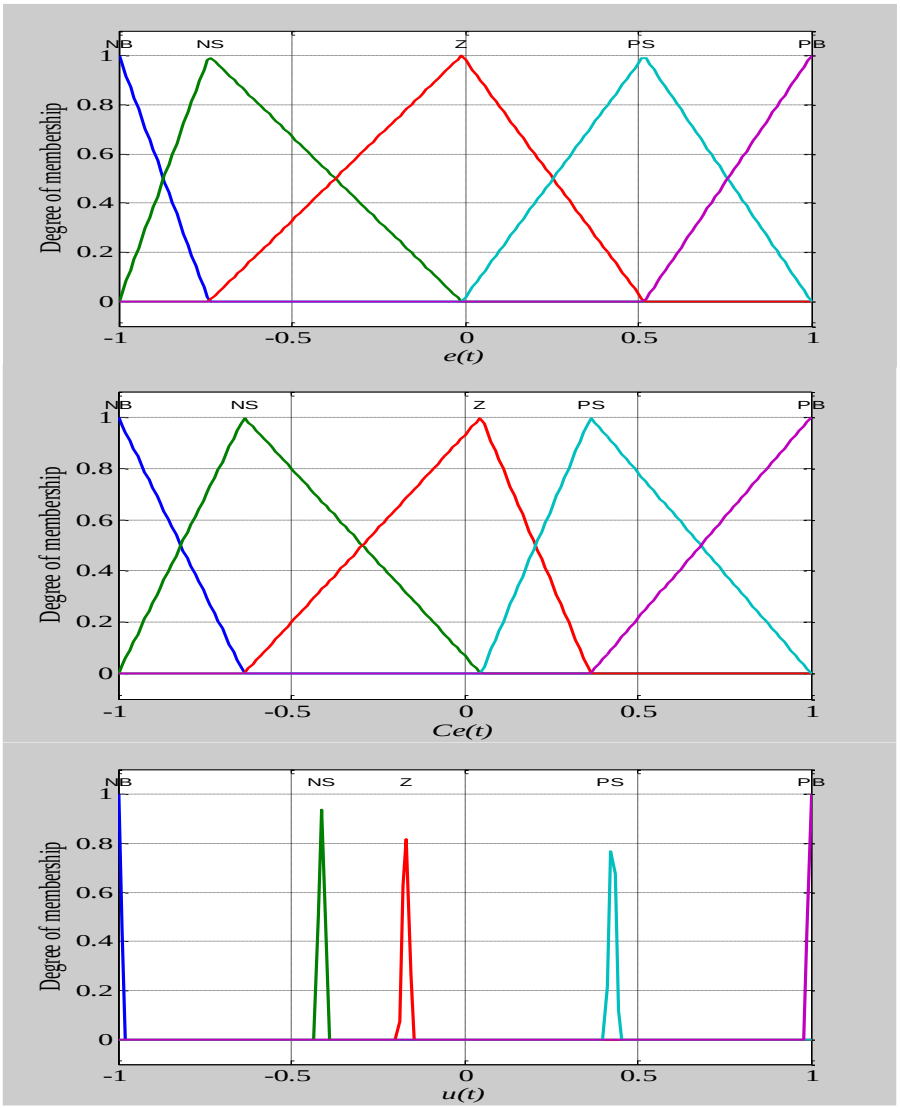


Fig (1.10). Inputs and output MFs of the induction motor speed control

1.7.2 Optimizing an FLC using GA's

Many methods for fuzzy control use genetic algorithms to search the fuzzy controller structure or parameters. However, one of the drawbacks of applying genetic algorithms in optimal fuzzy controller design is a lack of the theoretical knowledge. Each application has a different strategy to represent the fuzzy controller by chromosomes.

In this part; we will be interested by optimizing the MF's of the speed controller used in section 1.3 to achieve this task the fuzzy controller (MF's plus normalization gains) formed the chromosomes while the fitness was the square error of the speed. Genetic operator parameters used probabilities of crossover and mutation are respectively set to 0.8 and 0.005. Optimized MF's obtained are shown in Figure (1.11). We can see that the MF's intervals are changed compared with those in Figure (1.10).



Fig(1.11). Optimized inputs and output MFs of the induction motor speed control

Figure (1.12). shows the obtained motor speed response for the reference speed of 157rad/s. Thus we can see that the optimized speed controller gives the best response compared to IP and fuzzy controllers. Also it has a very fast perturbation rejection.

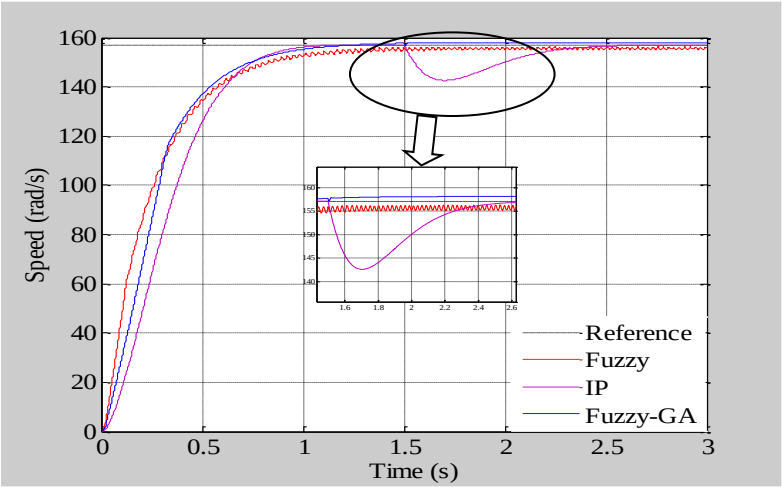


Fig (1.9). Induction motor speed response

1.8. Summary

This chapter has summarized the principles of some artificial intelligent techniques which are used to efficiency drives optimization. Attention has been focused on the two intelligent techniques investigated in this thesis namely fuzzy logic and genetic algorithms. The search of the IM speed response was investigated by both fuzzy logic and genetic algorithms to show the effectiveness of these two techniques.

The investigation of fuzzy logic and genetic algorithms in order to optimize efficiency of the induction motor drive is described throughout the next chapter.

Listes des figures

Table des matières