CHAPTER 4

FUZZY AND NEURAL NETWORK FOR SR MOTOR

4.1 Introduction

Fuzzy Logic control is based on fuzzy set theory. A fuzzy set is a set having uncertain and imprecise nature of abstract thoughts, concepts and without a clear or well defined boundary i.e. all elements of the fuzzy set belong to it to a certain degree given by the membership function (MF). A MF maps crisp input onto a normalized domain or fuzzy domain that has the characteristic function values in the interval (0, 1). In recent years, Fuzzy Logic systems have gained a lot of interest due to its ability to (1) Make correct decisions and carry out appropriate control actions to incorporate expert knowledge into the system design (2) efficiently handle vague, ambiguous and incomplete information.

The SRM is chosen based on the cost and controllability factors. SRM is used for low and high speed operations, especially automotive control applications because of its lower cost, reliable and robustness. But considering the controllability factor of SRM Drive exhibits the sensorless in existence. Even though, SRM has the ability to rapidly control the current, speed associated with conventional Motors, but it suffers torque ripples, acoustic noise.

In recent year, an Artificial Intelligent technique like Fuzzy logic is applied to improve the gains of the controllers to give better performance, Genetic Algorithm and Neural Networks are used to optimize the performance of the controller by tuning the membership functions and replacing the conventional controllers respectively. The choice of employing intelligent techniques depends on the machine characteristics and the desired
results. Several intelligent control techniques are discussed in previous research and used for several motors. The Knowledge-based fuzzy control system uses the knowledge, experience and intelligence of a human expert to make decisions about the behavior of the system. The Knowledge-based fuzzy control is depicted as the fuzzy rule-base with appropriate decisions to control the operation of the plant under study without human intervention.

Fuzzy Logic systems are rule based or knowledge based systems. It uses simple IF-THEN rules as an expert knowledge and store the information about the system to be controlled in a knowledge base and using the inference mechanism the fuzzy controller uses the knowledge base information to perform appropriate control action on the system under control.

The decisions made by the experts based on the predicted behavior of the system and knowledge gained is used to approximate the system behavior by means of linguistic values in the form fuzzy IF-THEN rules, which describes the control action that would be made by an expert operator. The rule base represents a static mapping between the antecedent and the consequent variables. Generally, Fuzzy Logic controller acts as a buffer between a nonlinear, highly complex system and desired control output, offering numerous advantages such as providing a model free approach, allowing human intelligence to be included in the control scheme and ability to perform any nonlinear control action as fuzzy systems are universal approximates.

The control strategy used for the design of the Fuzzy Logic Controllers is a Mamdani model which is a feedback controller.

In many processes, control algorithms based on fuzzy logic approach have been implemented. The objective for using such control techniques is due to the following reasons:

1. Compared to conventional control algorithms, a fuzzy logic system improves Robustness.
2. Control design is very simple for difficult systems
3. Easy implementation.

4.2 BENEFITS OF FUZZY LOGIC CONTROLLER DESIGN

Fuzzy logic is an alternative design methodology applied for developing both linear and non-linear systems. The non-linear and imprecise nature of the problem can be effectively solved by using fuzzy logic which gives better performance and reduces the development cost of the end product.

Conventional design for developing a controller requires knowledge about the underlying physical system. The development is a trial and error method which performs operations in an iterative process by checking the performance of the developed design. If the design performance is not up to the mark, then the modification is carried out in modeling and design. The controller is redesigned to repeat the process.

Using Fuzzy logic controller, improves the performance of the design by applying our knowledge and experience to understand the underlying characteristics of the system. Applying intelligent control techniques for problem related to uncertainty makes use of IF-THEN rules to describe the relationship between the inputs and outputs. Simulation is carried out to check the performance of the design and if the design is not up to the mark, modifications are made to tune the parameters of the controller by changing the fuzzy rules and the procedure is repeated for better results. The system design is not altered as conventional system, resulting in increased cost effective performance.

Fuzzy logic design has many advantages and they are:
1. Fuzzy reasoning is very simple and easy to understand.
2. Flexible in designing a system and easy to manipulate the functionality of the System.
3. Complex nonlinear functions can be modeled easily.
4. Gives better results for imprecise data and is fault tolerant.
5. Fuzzy controller will give optimized results when tuned using optimization techniques like genetic algorithm.

6. Fuzzy controller can be embedded with intelligence like neural network to understand the underlying system with the experience of experts.

7. Simplifies the implementation and reduces the product development cost.

**4.3 Basics of Fuzzy Logic Control**

The fuzzy set theory was introduced by Zadeh in 1965. He pointed out in “Fuzzy Sets”, such imprecisely defined sets or classes “play an important role in human decision making and it comes from the uncertain and imprecise nature of abstract thoughts and concepts”. Designing intelligent controllers using fuzzy logic has got much attention in recent years and researches have been carried out to develop complex industrial process controllers. The solutions to the real world problems are imprecise, vague and uncertain in nature.

Fuzzy logic is a powerful and efficient tool for qualitative modeling, which have been applied to a wide range of applications, such as automatic control, expert systems, pattern recognition, time series prediction and data classification. It uses basic measurement for meaningful representation of vagueness in natural or artificial languages. The fuzzy set theory uses linguistic values to represent an imprecise idea. The uncertainty in an event occurrence is described as randomness and the ambiguous nature of the event is represented as fuzziness. The fuzzy set is a simple extension of a classical set which is mathematically defined by assigning each possible individual element in a universe of discourse to a membership value between -1 and 1 using membership functions.

The membership function plays an important role in designing fuzzy systems (Passino et al 1998). The fuzziness in a fuzzy set whether the elements in the set are discrete (ordered or non-ordered) objects or continuous space with the mathematical formulation of fuzzy set theory, characterized by the membership function. The membership function essentially embodies all fuzziness for a particular fuzzy set and its
description is the essence of a fuzzy property or operation. Since all information contained in a fuzzy set is described by its membership function, it is useful to develop a lexicon of terms to describe various special features of this function. The membership of an object in a fuzzy set can be approximated. The membership to accommodate various “degrees of membership” on the real continuous interval is [-1, 1]. But there are infinite number of values in between the end points [-1, 1], which can represent various degrees of membership for an element. The main power and strength of membership functions are that it employs some amount of overlap.

Fuzzy logic is an intelligent computational technique that provides a strong framework for achieving robust and simple solution for non-linear systems. Fuzzy set is a collection of IF-THEN rules with uncertain and vague predictions that use a fuzzy reasoning model such as Sugeno (also TSK fuzzy model) and Mamdani models. The Sugeno type systems can be used to model any inference system in which the output membership functions are either linear or constant whereas Mamdani model produces either linear or nonlinear output.

The fuzzy logic controller consists of four stages: Fuzzification of input values, Knowledge and Rule Base, Inference Engine and Defuzzification as shown in Figure 4.1. The design of a fuzzy logic controller needs the selection of control elements and parameters and fuzzy reasoning operations, which include an implication operation, a compositional operation and aggregation operations of antecedents and consequences. The overall performance of the fuzzy logic controller depends on the configuration of the above factors.

In recent years, Fuzzy Logic Control (FLC) has emerged as a powerful technique and is being used in various applications, since nonlinear and automotive controls are frequent and uncertain in nature.
4.3.1 Linguistic Variables & Values

Fuzzy logic control systems consist of a knowledge base comprising of IF-THEN rules developed by expert members who prefer to use linguistic values to describe the behavior of the system. The linguistic variables and values are needed to specify fuzzy system's inputs, outputs and knowledge and rule base.

*IF pressure is high THEN volume is small*

The “pressure” and “volume” represent linguistic variables and “high” and “small” represents the corresponding linguistic values. Hence, linguistic variables can be said to be the variables which are described in terms of words instead of numeric values and the values assumed by these variables which describe their characteristics are termed as linguistic values.

4.3.2 Fuzzification

Fuzzification is the first step in the design of a FL controller and it refers to the process of converting a crisp or real value into a fuzzy variable. The crisp input values are transformed to linguistic variables. Many of the real world quantities or variables have a
considerable amount of uncertainty and this uncertainty happens because of imprecision, ambiguity and vagueness associated with those variables and probably they can be represented by a membership function (Sivanandam and Deepa 2010). Fuzzification is carried out by using the membership function which maps every crisp element (input values) in the universe of discourse onto a linguistic value with interval \([0, 1]\). The membership function gives the degree to which a certain element belongs and may be viewed as labels of the fuzzy set.

To design an efficient controller, it is very important for the designer to consider the following after completing the process of fuzzification using membership functions

1. It is required to carefully decide the number of fuzzy sets (membership functions) required for controller as input(s)/output(s) which corresponds to number of fuzzy rules,

2. Choose an appropriate shape, width and distribution of the membership functions.

### 4.3.3 Knowledge Base

Fuzzy rule base is considered as the core component of the Fuzzy Logic controller which stores all the information necessary to control the plant and it comes from the expert operator. Therefore, it is considered as the central part of the fuzzy logic controller as it helps the controller to take intelligent decisions by itself using the fuzzy rule base to make correct control action. Consider the following fuzzy rule as an example

*Fuzzy Rule: IF pressure is high THEN volume is small*

Where “pressure” is the crisp input from the process, “volume” is the fuzzy output and high and low represents the linguistic values for the linguistic variable pressure and volume respectively.
The fuzzy rule can be divided into two parts, i.e. IF (antecedent) THEN (consequent) where antecedent part defines the condition and consequent part gives the corresponding corrective control action. The number of rules in a rule base depends on the number of input and output variables and the number of membership functions attached to each variable.

The knowledge base consists of the database and the linguistic control rule base. The database contains information about the linguistic control rules and the data manipulation function of the controller. The rule base consists of IF-THEN rules which specify the control action defined by the expert member. The fuzzy logic controller looks at the input signals and by using the rules defined, it determines the appropriate corrective control output signals (control actions).

The main methods of developing a rule base are

1. Developing the rules using the knowledge of an expert for applying the control actions.
2. Modeling the control action of the operator.
3. Modeling the process.
4. Developing an optimized fuzzy controller.

The initial rules are obtained from the expert which is related to the physical consideration of the controller, the main objective to be considered by the fuzzy logic controller to achieve the control actions are:

1. Adjusting the control output by removing any significant error in the process output.
2. Corrective control actions are carried out to make a smooth transition of input to the nearest output reference value.
3. Keeping the process output value under control within the specified value, i.e., preventing it from exceeding the user specified limit.
4. By considering the two dimensional matrix of the input variables, each subspace is associated with a fuzzy output situation.

4.3.4 Fuzzy Inference Engine

The Mamdani inference system is the most widely used fuzzy inference model. The Fuzzy Logic engine infers the proper control actions based on the given fuzzy rule base.

\[ \mu_{FR_i}(x, y, z) = t(\mu_u(x), \mu_v(y)) \land \mu_w(z) \]

\[ = (\mu_{xi}(x) \land \mu_{yi}(y)) \land \mu_{zi}(z) \]  \hspace{1cm} 4.1

Where, FR\(_i\) represent the \(i^{th}\) fuzzy rule of the rule base consisting of \(n\) number of rules, the normalization is carried out using T-norm operator (t) gives the fuzzy output for the \(i^{th}\) rule Mamdani inference system. The T-norm operator (minimum) can be used for computing the antecedent part of the fuzzy rule for Mamdani implications.

4.3.5 Defuzzification

The control output from the fuzzy inference engine is fuzzy in nature; the fuzzy output is converted to crisp output to control the system under supervision. The technique which converts the fuzzy output of the controller into its corresponding crisp value is known as defuzzification process. There are many types of defuzzification methods available such as centre of gravity, bisector of area, mean of maxima, smallest of maxima and largest if maxima. However, centre of area is the most widely adopted defuzzification methods.

The reasons for using this method are:

1) It is fairly simple and requires less computational time and effort.
2) This method takes into account each and every activated rule of the fuzzy rule base as sum operator is used unlike other methods using max operator in which the rule with the highest value of the MF is considered for the output.

4.4 FUZZY LOGIC CONTROLLER

The traditional PI controllers are used for the control of a SR Drive but these controllers do not perform well during all conditions of the motor, when there is a large torque ripple. The fixed proportional (P) and Integral (I) gain controllers are optimized only for a small operating range and thus it deteriorates in system performance due to repetitive commutation failures and could not cope up with the sudden change in the operating condition of the system. The problem can be solved by updating the gain of the PI controllers with Fuzzy Logic control scheme.

The fuzzy logic controller uses a set of membership functions and linguistic rules obtained from the experience of the domain experts and the inference mechanism to determine the control action for a given process state. The Fuzzy membership functions can have different shapes depending on the preference of the human operators determined using their experience (Kaufman and Gupta 1985). The inference mechanism will perform a nonlinear mapping from its input to output using the if-then rules which describes the behavior of the mapping.

A Fuzzy Logic Controller has been designed has four stages, namely 1. Fuzzification 2. Knowledge base 3. Inference Engine and 4. Defuzzification.

4.4.1 Fuzzification

Fuzzification part maps the fuzzy logic controller inputs crisp values, scaled by input gains, into Fuzzy variables using normalized membership functions. The membership functions are used to convert crisp inputs into its fuzzy values, therefore; three triangular membership functions with a Universe of Discourse [-1, 1] are used for both input and output.
The input parameters are error and rate of change of error. Both the inputs are normalized and they fall within the range of the Universe of Discourse [-1, 1] and these normalized inputs are connected to the fuzzification block, consisting of three membership functions which are implemented in MATLAB.

4.4.2 Fuzzy Rule Base

Fuzzy rule base can be the central part of a fuzzy logic controller and it contains vital information which is necessary to control the operation of the plant under observation. The motivation of using the FL controller is to take intelligent decisions like an expert in order to achieve a precise control over the system. The fuzzy rules guide the controller to make proper decisions towards a correct control action. Fuzzy rule base is comprised of individual IF-THEN fuzzy rules of the form

Fuzzy Rule: IF x is high THEN y is medium

The fuzzy rule can be divided into two parts, i.e. IF (antecedent) THEN (consequent) where antecedent part defines the condition and consequent part gives the corresponding control action. The number of rules in a rule base depends on the number of input and output variables and on the number of membership functions attached to each variable.

In the proposed work, a rule base comprises of 3 X 3 i.e. 9 rules in the rule base which is used in corresponding two inputs fuzzified using three membership functions. The rule base is used to update the proportional and integral gains of the conventional PI controller [Munish 3]. In the proposed work, the N (negative), P (positive) and Z (Zero) represents the linguistic values. One of the rules is of the form

Rule: IF Δe is Zero and e is Negative then y is Positive
Where \( e, \Delta e \) represents the crisp input from the process, \( y \) is the fuzzy output. Overall, the fuzzy rule base provides the platform to include expert knowledge or human intelligence into the control system. The fuzzy rules are summarized in Table 4.1. The rules show that using three input membership functions (P, Z, N) and three output membership functions (P, Z, N) will provide the necessary functionality. The rules dictate that the results of each input membership function are combined using the AND operator, which corresponds to taking the minimum value.

<table>
<thead>
<tr>
<th>Error rate of change</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>Z</td>
<td>P</td>
</tr>
<tr>
<td>N</td>
<td>Z</td>
</tr>
</tbody>
</table>

Table 4.1: Fuzzy Logic Rules for Decision Making

The proportional and integral gain of the conventional PI controller updates using the rule base which is given in Table 4.2 and 4.3.
The number of linguistic regions is typically chosen and is alterable.

Table 4.2: Rule base for $\Delta K_p$

<table>
<thead>
<tr>
<th>$e/\Delta e$</th>
<th>Rate of change in Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Z</td>
</tr>
<tr>
<td>N</td>
<td>P</td>
</tr>
<tr>
<td>Z</td>
<td>P</td>
</tr>
<tr>
<td>P</td>
<td>Z</td>
</tr>
</tbody>
</table>

Table 4.3: Rule base for $\Delta K_i$

<table>
<thead>
<tr>
<th>$e/\Delta e$</th>
<th>Rate of change in Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Z</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Z</td>
<td>N</td>
</tr>
<tr>
<td>P</td>
<td>Z</td>
</tr>
</tbody>
</table>

4.4.3 Fuzzy Inference Engine

Fuzzy inference system deduces a meaningful interpretation or influence each IF-THEN rule comprised in a fuzzy rule base. Thus, fuzzy inference model produces a fuzzy
output for each activated rule. It gives the output of each individual rule in the rule base, but in order to reach to a correct control output, combined effect of entire rule base is required to be computed. As crisp measurement from the process (input) will be a part of more than one membership functions (fuzzy set) and it will result in activating a number of rules in the fuzzy rule base. Then the Fuzzy Logic engine infers the proper control actions based on the given fuzzy rule base.

\[ \mu_{FR_i}(\Delta e, e, u) = (\mu_{xi}(\Delta e) \land x_{yi}(e)) \land \mu_{zi}(u) \]  

Where, \( e \) and \( \Delta e \) represents the error and rate of change of error respectively, and \( u \) is the output. \( \mu_{xi} \) \( \Delta e \), \( \mu_{yi} e \) represents the membership value corresponding to the crisp input and \( \mu_{zi}(u) \) represents fuzzified control action taken as output.

### 4.4.4 Defuzzification

The Fuzzy Control action is translated to proper crisp values that are scaled by some appropriate output gains through the defuzzifier employing normalized membership function. The center of gravity method is used for defuzzifing process of output signals. The crisp output is given by

\[ u(\Delta e, e) = \frac{\sum_i^n u_i \min(\mu_{xi}(\Delta e), \mu_{yi}(e))}{\sum_i^n \min(\mu_{xi}(\Delta e), \mu_{yi}(e))} \]  

Where \((\Delta e, e)\) is the crisp output value taken as minimum value and \( u_i \) represents the centre of the output membership function.

### 4.4.5 Fuzzy Logic Controller – Design Steps

The fuzzy controllers are used increasingly in a wide area of applications to solve problems where system complexity, development time and cost are identified as a critical
issue. Fuzzy logic provides better results than conventional control methods and is used for
developing control algorithm for a variety of applications with uncertain and imprecise
nature. The fuzzy logic controllers are designed to control and maintain the torque without
much deviation in SRM. The fuzzy controller used in the proposed work to control the
speed and torque in the SRM Drive. The input parameters considered are the rate of change
of error (Δe), the error (e) and the output (u) to be applied to the system to get desired
output.

The fuzzy controllers are rule based systems which consist of fuzzy IF-THEN rules
formulated using linguistic values. The performance of the controller depends on the right
choice of membership function. Also, it is very difficult to represent the intelligence of the
expert perfectly by fuzzy rules. So, a trial and error method is carried out to vary the values
of the fuzzy parameters to keep the function of the controller within range to improve the
system performance.

The fuzzy rule base consists of many parameters and its function depends on how
to control and optimize the control system. The fuzzy logic controller contains a number of
parameters that can be varied to improve the performance of the controller. The parameters
are

a) The scaling factors for each variable
b) The meaningful fuzzy linguistic values and
c) The IF-THEN rules.

The following steps explain the working of the fuzzy logic controller:

(1) The initial values of the proportional (P) and integral gains (I) i.e. $K_{p0}$ and $K_{i0}$ of
the conventional PI controller are found using trial and error.

(2) Current error (e) and rate of change of this error (Δe) are taken as inputs to the
Fuzzy Logic controller. These inputs are normalized and then fuzzified using triangular
membership functions. Then, these fuzzified inputs are applied to the rule base for finding
Δ$K_p$ and Δ$K_i$. The output of the individual rules are aggregated which is fuzzy in nature,
and then defuzzification operation is applied using centre of gravity defuzzification method to obtain crisp values of $\Delta K_p$ and $\Delta K_i$. The gain of the conventional PI controller is updated using the obtained crisp output. (3) From the output ($u$), examine the system performance. The tolerance parameter is calculated to enhance the function of the controller to give best possible performance.

Fig.4.2: Fuzzy Logic Controller circuit for the speed control of SRM

4.5 HISTORY OF ANN & NEURAL NETWORK CONTROLLER

Artificial Neural Network (ANN) is different type, where each type is suitable for a specific application. The main interest here is applying the ANN for a non-linear mapping. Neural networks can be used to estimate input-output functions. ANNs are trainable dynamical systems. Unlike statistical estimators, they estimate a function without a mathematical model of how outputs depend on inputs. They are defined model free estimators. They learn from experience data with a numerical sample data.

Supervised forward models provide the most tractable and most applicable neural models. Assume that network has a set of observations. This set consists of a group of input and output value pairs. Each of these pairs is of the form $(x, y)$, where $x$ is the input and $y$ is the output. The set of these pairs inherits the mapping between the input and the output. The emphasis is to extract the closest mapping from the input domain to the output
range. The measure of this closeness can be chosen to conform to some appropriate form such as least squared error (though it is not the only function but it is simple value function). Therefore the objective is to estimate an unknown function $f(x+y)$ derived from observed set samples $(x_l, y_l)$, $(x_m, y_m)$ by minimizing an unknown expected error functional $E(w)$.

Error is defined as desired performance minus actual performance. Desired performance refers to the value $(y_i)$, while the actual performance is the network output to the input $(x_i)$. Supervision is used to the desired performance and actual performance of the network to provide an ever-present error or teaching signal. $E(w)$ defines an average error surface over the weight space. At each iteration, the current sample $(+y_i)$ and the previous initial conditions define an instantaneous error surface. They indirectly search $E(w)$ for the global minimum by using an optimization algorithm such as stochastic gradient descent. Due to the nonlinear nature of the problem, they often converge to a local minimum $(w^*)$. The local minimum $w^*$ may differ significantly from the global minimum of $E(w)$. Some shallow local minimum may be no better than expected error values determined by randomly picking network parameters. Since they do not know the shape of the $E(w)$, they do not know the depth of its local minimum. In general, nonlinear multi-variable systems define complicated, and bumpy, average error faces.

Consider a set of Input $(x_i)$ and Output $(y_i)$ which are derived from an actual measurement or simulation of a specific plant. It is desired to find a function which can resemble the whole plant based on this set of discrete vectors $(x_i, y_i)$. In other words they want to find a function which approximates the plant to a specific degree of accuracy instead of analyzing the nonlinear equation of the plants.
A typical feed-forward ANN is shown in fig. 4.3. It has one input node \( x \), and one output node \( o \) along with \( k \) hidden nodes, \( V_1 \) to \( V_K \). The objective is to produce the output \( o_i \) which is as close as possible to the desired target \( y_i \), when \( x_i \) is the input of the network.

An artificial neuron as shown in fig. 4.4 is the basic element of a neural network. It consists of three basic components that include weights, thresholds, and a single activation function.

4.6 Weighting Factors

The values \( W_1, W_2, W_3, \ldots, W_N \) are weight factors associated with each node to determine the strength of input row vector \( X = [x_1 \ x_2 \ x_3 \ldots \ x_n]^T \). Each input is multiplied by the associated weight of the neuron connection \( X^T W \). Depending upon
the activation function, if the weight is positive, $X^T W$ commonly excites the node output; whereas, for negative weights, $X^T W$ tends to inhibit the node output.

4.7 Threshold

The node’s internal threshold $\theta$ is the magnitude offset that affects the activation of the node output $y$ as follows:

$$y = \sum_{i=1}^{n} (X_i W_i) - \theta_k$$  \hspace{1cm} 4.4

4.8 Activation Function

In this subsection, five of the most common activation functions are presented. An activation function performs a mathematical operation on the signal output. More sophisticated activation functions can also be utilized depending upon the type of problem to be solved by the network. All the activation functions as described herein are also supported by MATLAB package.

4.9 Linear Function

A linear function satisfies the superposition concept. The function is shown in figure 4.5.

Fig. 4.5: Linear function
The mathematical equation for the above linear function can be written as

\[ y = f(u)\alpha \mu \]  \hspace{1cm} (4.5)

Where \( \alpha \) is the slope of the linear function in equation. If the slope \( \alpha \) is 1, then the linear activation function is called the identity function. The output function \( y \) of identity function is equal to the input function \( u \). Although this function might appear to be a trivial case, nevertheless it is very useful in some cases such as the last stage of a multilayer neural network.

**4.10 Threshold Function**

A threshold activation function is either a binary type or a bipolar type as shown in fig. 4.6 and fig 4.7, respectively. The output of a binary threshold function can be written as:

\[ y = f(u) = \begin{cases} 
0 & \text{if } u \leq 0 \\
1 & \text{if } u \geq 0 
\end{cases} \]  \hspace{1cm} (4.6)

![Fig. 4.6: Binary Threshold Activation Function.](image-url)
The neuron with the hard limiter activation function is referred to as the McCulloch-Pitts model.

### 4.11 Piecewise Linear Function

This type of activation function is also referred to as saturating linear function and can have either a binary or bipolar range for the saturation limits of the output. The mathematical model for a symmetric saturation function in Fig. 4.8 is described as follows:

\[
y = f(u) = \begin{cases} 
-1 & \text{if } u \leq -1 \\
\text{otherwise} & \\
1 & \text{if } u \geq 1
\end{cases}
\]  

4.7
4.12 Sigmoidal (S shaped) function

This nonlinear function is the most common type of the activation used to construct the neural networks. It is trained to derive in mathematical, differentiable and strictly increasing function. A sigmoidal transfer function can be written in the following form:

\[ f(x) = \frac{1}{1 + e^{-\alpha x}}, 0 \leq f(x) \leq 1 \]

4.8

Fig. 4.9: Continuous and Differentiable function
where $\alpha$ is the shape parameter of the sigmoid function. By varying this parameter, different shapes of the function can be obtained as illustrated in Fig.4.9. This function is continuous and differentiable.

The output of any hidden or output neuron is calculated on a weighted sum of the inputs to that neuron. In addition to the inputs to each processing neuron, a bias level $B$ (usually equal to one) may also be applied to each neuron. The bias is connected with an adjustable weight to each hidden and output neuron. The output of the neuron $V_k$ is generated by applying a sigmoid non-linearity as shown in figure 4.10 to the excitation. The same nonlinear function is also used for the output neuron.

![Sigmoid nonlinear function](image)

Fig. 4.10: Sigmoid nonlinear function

The sigmoid function used for this study has an input to output function given in equation. The output range of the function presented in equation is the closed interval $[0, 1]$ (continuous interval between zero and one, including zero and one). Therefore, the output of each processed node (hidden and output) lies in the continuous interval between zero and one. The parameter $\beta$ in equation decides the function's slope.

$$g[h] = \frac{1}{1 + e^{-2\beta h}}$$  \hspace{1cm} (4.9)

Using an optimization technique, the weights are adjusted such that the error for the entire input-output set becomes as low as possible. Usually the mean square error, equation, is chosen as the performance index or cost function:
\[ E[w] = \frac{1}{2} \sum_i (y_i - o_i)^2 \]  

4.10

Where \( i \) is the number of input pattern (\( i = 1, 2, \ldots, p \)) and \( o_i \) is the \( i^{th} \) computed output. We seek the weight vector \( w \) which results in a global minimum for \( E[w] \).

Learning or weight adjustment is carried out by determining the contribution of each connection to the output error and correcting that weight correspondingly. Applying the steepest descent algorithm, the adjustment in \( w_k \) yields as:

\[ \Delta w_k = -\eta \frac{\partial E}{\partial w_k} \]  

4.11

Where \( \eta \) is called the learning rate and is a very crucial parameter in the learning process. This procedure is also called back propagation, since the output error is back propagated through the network in order to determine the contribution of each single weight to it. It should be mentioned that \( w_k \) is chosen as an arbitrary weight, and the same derivation applies to all the weights either between two neurons or the weights between the bias and any neurons.

It is reported in the literature, that the cost function is usually full of valleys with steep sides but a shallow slope along the floor and the aforementioned method usually gets stuck in these regions and the learning process becomes too slow. There are a number of ways of dealing with this problem, including the replacement of gradient descent by more sophisticated minimization algorithms. However a much simpler approach, the addition of a momentum, is often effective and is very commonly used.

The idea is to give each connection some inertia or momentum, so that it tends to change in the direction of the average downhill force that it feels, instead of oscillating wildly with every little kick. Then the effective learning rate can be made larger without
deviated oscillations occurring. This scheme is implemented by giving a contribution from
the previous time step to each weight change:

\[ \Delta w_k (m + 1) = -\eta \frac{\partial E}{\partial w_k (m)} + \alpha \Delta w_k (m) \]  

4.12

Besides, it can be shown that given any \( \varepsilon > 0 \) and any function \( f; [0, 1]^n \prec R^n + R^n \), there exists a three layer back-propagation neural network that can approximate \( f (f E L2) \) to within \( E \) mean squared error accuracy". Here \( L_2 \) is the mathematical space of
functions that can be approximated by its Fourier series to any desired degree of accuracy
in the mean squared error sense.

Although the above statement guarantees the ability of a multi-layer network with
the correct weights to accurately implement an arbitrary function, it does not comment on
whether or not these weights can be learned using any existing learning law.

In addition, there is no guarantee that the function king approximated satisfies the
above \( L_2 \) condition. Such a function will not be amenable to approximate by an ANN. The
above theorem suggests that assuming a reasonable function, a three layer ANN should
normally suffice for most applications with variable number of hidden units. There is no
rule or theorem expressing the optimal number of hidden layer neurons, and is usually
derived from empirical results or trial-error method.

The update rule, as defined by the equation is written in the incremental form. In
other words, for each input-output pair (pattern \( i \)), the adjustment to individual weights are
derived from this equation. The pattern \( i \) is presented to the ANN network, and then all the
weights are updated before the next pattern is considered. This clearly decreases the cost
function (for small enough \( \eta \)) at each time step, and lets successive steps adapt to the local
gradient.
The developed neural network system can be trained and implemented in two different ways. In one approach, the network can be trained with a set of known input-output data pairs known as the training set and after some standard verification, can be used for the actual application. In this way the ANN network, after extracting the rules from examples or a training set is known to perform some sort of generalization whenever it comes across new inputs. This method is called off-line training because the weight adjustment is performed prior to implementing the network in the analysis.

In the second method, the learning can be done while the network is being implemented in the process. In this way the network corrects itself as it comes across new inputs; learning while new sequences are being presented rather than after they are complete. It can thus deal with sequences of arbitrary length and there is no requirement to allocate memory proportional to the maximum sequence length. This method is called on-line training. In this method there is no generalization and all the input-output pairs are members of the training set.

The neural network architecture used throughout this dissertation is as shown in fig.4.11 it is a two layer network with one input unit, two hidden units (adjustable) and one output unit. The weights are altered (learning process) in order to minimize the mean square error between the desired and actual outputs, using equation. This is done by performing a gradient descent algorithm on equation, which results in the normal back propagation algorithm (BP).

![Network Architecture](image-url)
This model is developed as a block and used in the digital simulation analysis program. Therefore at each time step an input $x_i$ is represented to the neural network model and the error between $o_i$ and $y_i$ is then used to adjust the weights with back-propagation algorithm.

For on-line training (incremental), the weight update is done once each time step. Thus for this particular implementation equation takes the form:

$$E[w] = (o_i - y_i)^2$$

4.13

Switched reluctance motors drive systems traditionally using PI controllers with fixed gains. Although such controllers have certain disadvantages, they are never-the-less rugged and operate satisfactorily for perturbations within a small operating range. On the other hand, ANN controllers have some specific advantages, whereby the use of ANN controllers has been shown to introduce flexibility and fault tolerance into the performance of the controllers. One of the most important features of Artificial Neural Networks (ANN) is their ability to learn and improve their operation using a neural network training data. The basic element of an ANN is the neuron which has a summer and an activation function. The mathematical model of a neuron is given by:

$$y = \Phi \sum_{i=1}^{N} w_i x_i + b$$

4.14

Where $(x_1, x_2, ..., x_N)$ are the input signals of the neuron, $(w_1, w_2, ..., w_N)$ are their corresponding weights and $b$ is bias parameter. $\Phi$ is a tangent sigmoid function and $y$ is the output signal of the neuron. The ANN can be trained by a learning algorithm which performs the adaptation of the weights of the network iteratively until the error between target vectors and the output of the ANN is less than a predefined threshold. The most popular supervised learning algorithm is back-propagation, which consists of a forward and backward action. In the forward step, the free parameters of the network are fixed, and
the input signals are propagated throughout the network from the first layer to the last layer. In the forward phase, they compute a mean square error.

$$E(k) = \frac{1}{N} \sum_{i=1}^{N} (d_i(k) - y_i(k))^2$$  \hspace{1cm} 4.15

Where $d_i$ is the desired response, $y_i$ is the actual output produced by the network in response to the input $x_i$, $k$ is the iteration number and $N$ is the number of input-output training data. The second step of the backward phase, the error signal $E(k)$ is propagated throughout the network in the backward direction in order to perform adjustments upon the free parameters of the network in order to decrease the error $E(k)$ in a statistical sense. The weights associated with the output layer of the network are therefore updated using the following formula:

$$w_{ji}(k+1) = w_{ji}(k) - \eta \frac{\partial E(k)}{\partial w(k)}$$  \hspace{1cm} 4.16

Where $w_{ji}$ is the weight connecting the $j^{th}$ neuron of the output layer to the $i^{th}$ neuron of the previous layer, $\eta$ is the constant learning rate. The objective of this Neural Network Controller (NNC) is to develop a back propagation algorithm such that the output of the neural network speed observer can track the target one. Figure 4.12 depicts the network structure of the NNC, which indicates that the neural network has three layered network structure. The first is formed with five neuron inputs $\Delta(\omega_{ANN}(K+1)), \Delta(\omega_{ANN}(K)), \omega_{ANN}, \omega_{SF}(K-1), \Delta(\omega_{SF}(K-2))$. The second layer consists of five neurons. The last one contains one neuron to give the command variation $\Delta(\omega_{SF}(K))$. The aim of the proposed NNC is to compute the command variation based on the future output variation $\Delta(\omega_{ANN}(K+1))$. Hence, with this structure, a predictive control with integrator has been realised. At time $k$, the neural network computes the command variation based on the output at time $(k+1)$, while the later isn’t defined at this time. In this case, it is assumed that $\omega_{ANN}(K+1)=\omega_{ANN}(K)$. The control law is deduced using the recurrent equation given by,
The proposed neural network controller is shown in Fig. 4.12. The proposed SRM Drive system with Neural Network Controller is shown in Fig. 4.13.

\[
\omega_j(k) = \omega_j(k-1) + G\Delta(\omega_j(k))
\]

4.17

Fig. 4.12: Neural network controller.

Fig. 4.13: SRM with Neural Network Controller.
4.13 THE FEED FORWARD ARCHITECTURE

Fig. 4.14 represents a feed forward NN with two inputs and one output. In a feed forward network, the output of the neurons in one layer acts as the input to the neurons of the following layers; with no feedback connection present in the network.

This topology is chosen as

- It is similar to FL architecture with two inputs and one output and hence can be combined to form a NF system.
- Learning algorithms such as steepest descent can be conveniently used with such network architecture.

![Feed forward ANN Architecture with two inputs one output.](image-url)

Fig. 4.14: Feed forward ANN Architecture with two inputs one output.

The proposed ANN has two inputs, either one or two hidden layers or one output neuron; the simplest architecture version consists of a single hidden layer. The input layer simply acts as a fan-out input to the hidden layer where two neurons are used. The outputs are transformed through a sigmoidal AF and fed to the output layer through their weights. The output layer has only one neuron with a sigmoidal AF and three inputs (two from the hidden layer and one constant bias). The output is multiplied by a constant scaling factor, $\pi$, to get the required alpha order. One of the three possible inputs is used to study the performance of this controller:

(i) Input 1: The reference current, $I_{ref}$, and bias,

(ii) Input 2: The measured dc current, $I_d$, and bias, or
(iii) Input3: The current error, $I_e = I_{ref} - I_d$, and bias.

4.14 Online reinforcement learning method

In this approach, an error reinforcement method is used to determine the target controller output (alpha order) from the SRM plant response ($I_d$). In the Back Propagation (BP) method, the controller output (alpha-order) is compared with the desired known alpha-order and the error is back propagated to control the output. Therefore the use of conventional BP requires a prior knowledge of the alpha order. In the reinforcement learning method used here, the current error, $I_d$, is used to adjust the weights of all the ANN layers. The SRM plant current output, $I_d$, which is the performance measure, is controlled by an alpha order resulting from the neuron in the output layer. Since the SRM plant performance is directly related to the output (alpha order) of the ANN, this critical block has been investigated in detail in this work.

4.15 Weight adjustment /learning

The error function, $E$, to be minimized for a given input pattern is given by

$$E = \frac{1}{2} (I_{ref} - I_d)^2$$ \hspace{1cm} 4.18

The change in weights of the input and the output layers is adjusted according to the Generalized Delta rule in the negative direction as:

$$dw \alpha \frac{\partial E}{\partial w}$$ \hspace{1cm} 4.19

Where

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial v_i} \cdot \frac{\partial v_i}{\partial net_i} \cdot \frac{\partial net_i}{\partial w}$$
Where for the $k^{th}$ layer of the $i^{th}$ neuron $v_i = \frac{1}{(1 + e^{-net_i})}$ (For asymmetrical sigmoid), $net_i = \Sigma w_{ji} y_j$ (weighted sum at the $i^{th}$ neuron), $w_{ji} = \text{weights connected between } j^{th} \text{ to the } i^{th} \text{ Neuron and } y_j = \text{output from } j^{th} \text{ to the } i^{th} \text{ neuron.}$

Note that although the second and third partial derivatives can be explicitly evaluated, the first term cannot be evaluated explicitly. As derived in equation, the adjustment of weights using the standard BP algorithm requires a priori knowledge of the system, i.e., the ANN requires the error to be back propagated through the dc system. Therefore, the term $\frac{\partial E}{\partial v_i}$ is expressed as being proportional to the current error, $I_c$. Hence the change in weights for both the output and hidden layers is:

$$\Delta w_{ij}^{(t+1)} = -\eta \times I_c \times f'(net_i) \times y_i + \Delta w_{ij}^{(t)} \times \mu$$ \hspace{1cm} (4.20)

Where, $\eta$ is the learning parameter, $I_c$ is the error, $f'(net_i)$ is the derivative of the sigmoidal function, $y_i$ is the input to the $i^{th}$ neuron, $\mu$ is the momentum constant and $\Delta w_{ij}^{(t+1)}$ and $\Delta w_{ij}^{(t)}$ are the change in weights at the instants $(t+1)$ and $(t)$ respectively.

### 4.16 NEURAL NETWORK DESIGN FOR SR MOTOR ESTIMATION

A typical neural performance for Switched Reluctance Motor rotor position estimation is shown in Fig.4.15. Basically, the non-linear input reduces the hidden layer from the characteristic design. The standard design rules for neural network establish that neuron exists on the input layer for every input. Therefore, an extra neuron exists in the input layer for the reduced network as opposed to a characteristic one is shown in Fig. 4.15(a). The non-linear input is chosen as the product of the other two inputs so that it can be developed in real time and minimizes the effects of the overall computational.
Each neuron contains an activation function and selected function for the input neurons represented by the single-sided sigmoidal function as given in eqn. [4.21],

$$S(x) = \frac{1}{1 + e^{-x}}$$  \hspace{1cm} 4.21

The output neuron is a directly linear arrangement of its inputs and weighting factors. In addition to the activation functions, each neuron contains a threshold that adjusts the receptiveness of the neuron. If it counts the number of operations required by a typical neuron with two inputs and requires two multiplications, two additions (two inputs and threshold) and activation function estimations.

### 4.17 ANN MODEL OF SRM WITH ROTOR POSITION

ANNs are systems that are with intent constructed to make use of some organizational principles like those of the human brain and are widely used a lot in engineering fields after the expansion of computer technologies. Neural Networks have a
nonlinear, adaptive and parallel distributed memory. Because of learning, generalization and highly computing ability, ANNs are widely used in many fields. One of these fields is automotive control. In general, SRMs are worked in saturation, and consequently there are nonlinear relationships among the rotor parameters.

The recognizing of mathematical modeling of the SRMs is difficult because of nonlinear characteristic (for example, the flux linkages as a function of current and rotor position). Because of using the artificial intelligence techniques in control fields which have ability of modeling of the SRM characteristics are realized and based on ANN and fuzzy inference system. The relationship between flux linkages and current in SRM is shown in Fig. 4.16.

![Fig.4.16: Relationship between flux linkages and current in SRM](image)

The ANN is realized that a feed forward multi layer perceptron (MLP) structure is trained by using Levenberg-Marquardt learning algorithm. In order for each layer’s tangent sigmoid, purelin, tangent sigmoid and purelin transfer functions are used respectively. The training of the ANN is realized by using a data set which has (80 values for each) voltage, current and rotor position. The normalized voltage and current values are given as input and rotor position values are given as output to the ANN. The ANN has an input layer, four hidden layers with in order seven-five-three-five neurons and an output layer with a neuron as will be shown in Fig. 4.17.
The trained ANN is tested by a test data set which has also 80 normalized voltage and current values and the real rotor angle values are compared with obtained rotor position values from the ANN in Fig. 4.18.

Fig. 4.18: Comparing the trained ANN outputs and real rotor position values
ANN-BASED ROTOR POSITION ESTIMATOR IN SR MOTOR

The rotor estimation in SRM is performed in this study to design the position observer built on the 4 phases, 3000 rpm and 8/6 poles SRM as shown in Fig.4.19.

Fig. 4.19: Cross section of 8/6 poles SRM

The network structure used for estimating the rotor position by using ANN is shown in Fig.4.20. The inputs of the networks \( i_j \) and \( \psi_j \) are the phase current and the flux linkage data, obtained from the non-linear full model of the SRM. The output of network is the actual rotor position per phase \( \theta_j \) computed according to the inputs. \( \theta_j \) is the desired rotor position per phase, while \( e \) is the error between actual and desired rotor position values. The network is composed of 4 layers: an input layer (P), two hidden layers (R, S) and an output layer (T).

Fig. 4.20: Architecture of the neural network for modeling of the inductance and flux linkage
In nodes at the $R$, $S$, and $T$, the output of the nodes is calculated by using activation function given as follows:

$$y(x) = \exp\left(-\frac{(x-c)^2}{\sigma^2}\right)$$  \hspace{1cm} 4.22

Here, $x$ represents input of the nodes, $y$ represents output of the nodes related to $x$, $c$ center of Gaussian function, and $\sigma$ its width. Feed forward model of the layers can be described as follows.

4.19 Feed forward Algorithm

P layer: It is the input layer and the entry of this layer, $i_j$ and $\psi_j$ are the values of the current and flux linkage data, respectively. The inputs and the outputs of this layer are obtained as follows,

$$x_p = \{i_j, \psi_j\}, \text{and} \quad y_p = x_p \text{ Where } p = 0...P$$ \hspace{1cm} 4.23

R layer: It is the first hidden layer, the inputs and the outputs of this layer are obtained as follows,

$$x_r = \sum_{p=0}^{P} y_p \cdot w_{rp}, y_r = y(x_r) \quad \text{ Where } r = 0...R$$ \hspace{1cm} 4.24

S layer: It is the second hidden layer, the inputs and the outputs of this layer are obtained as follows,

$$x_s = \sum_{r=0}^{R} y_r \cdot w_{rs}, y_s = y(x_s)$$

where $s=0..S$ \hspace{1cm} 4.25
T layer: It is the output layer, the inputs and the outputs of this layer are obtained as follows,

\[ x_i = \sum_{x=0}^{S} y_s w_s, \quad \text{And} \quad y_i = y(x_i) \quad 4.26 \]

While the terms above layers are combined; inductance model can be expressed as follows,

\[ \hat{\theta}_j = y \left( \sum_{x=0}^{S} y \left( \sum_{r=0}^{P} y \left( \sum_{p=0}^{P} y_p w_{pr} w_{rs} w_s \right) \right) \right) \quad 4.27 \]

The estimated rotor position model obtained as a result of the feed forward algorithm has been completed and the back propagation learning algorithm is realized for the optimization of the weights in the network.

4.19.1 Back propagation Learning Algorithm

The learning algorithm of the ANN using the supervised gradient method, the energy function \( E \) is chosen as follows,

\[ E(k) = \frac{1}{2} e^2 (k), \quad \text{Where, k=1….K} \quad 4.28 \]

K denotes total number of input-output patterns and error value for each pattern,

\[ e(k) = \theta_j (k) - \hat{\theta}_j (k) \quad 4.29 \]

Where \( (k) j \theta \) is the desired value \( \hat{\theta} \) (k) actual value.
4.20 IMPLEMENTATION OF PROPOSED WORK

Many industrial applications have shown interest in applying intelligent control technique like Fuzzy Logic, Neural Networks, Genetic Algorithms, and Adaptive Neuro-fuzzy inference systems etc., to improve the performance and efficiency of the system. Recently developed control systems based on Artificial Neural Network, Fuzzy system and Genetic algorithm are fast, reliable, can be used for sr drive for nonlinear and automotive control applications. A hybrid scheme using Fourier linear combiner and a fuzzy expert system for controlling the torque ripples in SR dive is presented.

A Fuzzy logic based scheme has been used to solve the problem of fixed gain of the conventional PI controller by suitably adjusting the gains of the PI controller with changing motor condition for fast stabilization of torque ripples. FL is used to provide smooth transition from one control mode to another mode. In a neural network controller is employed to replace a conventional PI controller in SRM drive.

Fuzzy Logic control and Artificial Neural Network based control are the two popular control methodologies used for SR Drive control. The main advantages of using Fuzzy based controls are

1. Accurate mathematical model of the plant is not required to estimate the control input
   Under disturbance conditions; it requires only general knowledge about the plant.

2. The controller uses simple if-then rules to incorporate knowledge of the plant.

3. The controller uses simple inference mechanism to take appropriate control actions on the plant.

The Neural network based controllers have higher efficiency and have the ability to learn any kind of nonlinearity [Munish]. The NN based controllers are highly adaptive and their parallel processing capability gives faster response. An intelligent control system can
be developed by combining the advantages of both Fuzzy logic and neural network to have a more intelligent and efficient control over the plant which leads to the development of Neuro-Fuzzy based controller systems. Recent researches on Neuro-fuzzy systems showed the potential goodness of these systems [Munish]. A survey on application of Neuro-fuzzy systems to the SRM Drive problem is presented.

4.21 Neuro-Fuzzy Systems

Fuzzy Logic and neural networks are complementary techniques used in the design of an adaptive intelligence system for SR motor. Artificial Neural Network (ANN) learns from scratch by adjusting the interconnections (weight) between layers. Fuzzy Inference System (FIS) is a well-known computational framework dealing with non-linearity and uncertainty and uses the concept of fuzzy set theory, if-then rules and reasoning.

A Neuro-Fuzzy controller system combines the learning power of artificial neural network and the knowledge representation feature of fuzzy logic of into one system. The aim of combining the inherent features of Neural-network and Fuzzy logic is to inherit the advantages of both control techniques and overcome their individual disadvantages. FL controllers are easy to design and implement, have powerful knowledge representation in the form of simple if-then rules, do not require a mathematical model of the plant, and they are robust and flexible. But the disadvantage of these controllers is, it does not have any learning capability. Since Fuzzy Controllers are not adaptive to the changing environment, the trial and error method is used for tuning the parameters such as membership functions and rule base of the controller and it consumes more time to tune these parameters if the rules are relatively large in number. The fuzzy system requires an expert input or instructions in order to define fuzzy rules.

Artificial Neural Network based controllers have the capability of handling any kind of nonlinearity quite efficiently. The neural-network based controller uses massive parallelism to learn and is highly adaptable to changing system conditions by adjusting the connection weights of the neurons. The connection weights can be updated online by using
some learning algorithm or offline by training the network with training data. The neural network based controller is applied in a SRM Drive system incorporate flexibility and fault tolerance to improve the performance of the controllers.

The ability and performance of a NN controller is greatly influenced by the weight adaptation algorithm and the amount of noise in the data and it suffers from a large number of training cycles and computational burden. The main drawback of neural- network based controllers are, 1) once trained, these controllers act as a black box and it is not possible to extract or add any information to the already trained neural networks, 2) has slow convergence and inability to deal with linguistic form of information.

The Neuro-Fuzzy based controllers have the learning ability, adaptability to changing environment and have the capability of dealing with both numerical and linguistic data which is missing in Fuzzy Logic and Artificial Neural Network controllers respectively. A Neuro-fuzzy system makes use of fuzzy inference system which is trained by a learning algorithm derived from neural network theory.

A Neuro-fuzzy system is a fuzzy inference system trained using the neural network learning algorithm. The learning algorithm fine-tunes the fuzzy system parameters (membership function and rule base) by removing incomplete information of the rule base and defines suitable criteria for selection of membership function and provides various levels of degree of overlapping and quantization. The neural technique is used to solve these problems for fuzzy reasoning. The Neuro-fuzzy system combines the advantages of neural-network and fuzzy system. The Neuro-fuzzy system applies both these technologies for four cases

1. The task of designing and fine tuning the membership functions of fuzzy systems can be automated using neural networks.
2. The learning capabilities of fuzzy system and neural network acts separately.
3. Neural network acts as corrective mechanisms for fuzzy systems.
4. Based on users’ preferences and individual needs the standard system can be customized using neural network

The Neuro-fuzzy system integrates the advantages of both neural-network and fuzzy system. This controller is much more effective than Fuzzy logic and neural network based controller, since it has the capability of self-learning the gain values and adapts accordingly to situations, thereby adding more flexibility to the controller.

The Neuro-Fuzzy controller is used as a better alternative to tune the PI gains of the conventional PI controller in the SR Drive. The use of properly-designed Neuro-fuzzy logic controllers has shown at least marginal improvement in the operation of motor compared with conventional constant-parameter PI controllers. Since, the tuning of constant-parameter PI controllers is a compromise between the speed of response and transient stability of the controller after the occurrence of small disturbances and has least robustness to tolerate large disturbances due to faults.

The Neuro-Fuzzy Inference System (NFIS) is a fuzzy logic based paradigm that uses the learning capability of neural network to enhance the performance of the intelligent system using a priori knowledge acquired from the expert knowledge. For a given data set, the NFIS constructs a fuzzy logic system whose membership functions’ parameters are adjusted using multi layer perceptron. The levenberg marquardt algorithm allows the fuzzy logic system to learn information about a data set, to compute the membership function parameters that allow the associated fuzzy inference system track the given input/output data.

4.22 Neuro Fuzzy Controller Architecture

The neuro-fuzzy system consist of 2 input, 1 output with 4 layers neural network, where each layer represents particular fuzzy logic operation from fuzzification to defuzzification. Layer 1 acts as the fuzzification layer, layer 2 contains the fuzzy rules and layer 3 acting as consequent and layer 4 as defuzzification layer. The network designed
thus takes the advantages of parallelism and learning ability from neural network and simple representation of data in the form of linguistic variables in the hidden layer uses fuzzy logic. The architecture of the Adaptive neuro-Fuzzy controller is shown in fig.4.21.

![Neuro-Fuzzy Controller model](image)

**Fig.4.21: Neuro-Fuzzy Controller model.**

The proposed NFIS control employs the Multi-layer perceptron and applies levenberg marquardt learning algorithm, which is the variant as a nonlinear extension of least-square methods. The output of an adaptive network is linearized with respect to its parameters and levenberg marquardt algorithm is employed to update all parameters.

The overlapping region of the constant torque and constant power is separated into a constant torque region with a feedback network using Neuro Fuzzy logic. The controller is important for the proper operation of the SRM. It gives the precision control of firing angle for smoother operation. The controller can also be used for current control to provide control over back EMF and ripple reduction. The same is implemented using neural and fuzzy logic (neurofuzzy controller). The prediction advance “Turn ON angle” is done by neural logic and implemented by fuzzy logic. The desired current level based on speed and load is calculated and the required current level is controlled by the phase angle done in
ON-OFF region. The graphical representation of the proposed work (Dotted Line) is shown in Fig. 4.22.

![Graphical representation](image1)

Fig. 4.22: Graphical representation

The proposed architecture is implemented in MATLAB and the comparative graph with and without (existing) feedback control circuit is shown in Fig. 4.23.

![Block Diagram](image2)

Fig. 4.23: Block Diagram of the proposed architecture
One of the greatest advantages of the proposed method is the torque ripple reduction. Since, it is based on and controlled by controlling current; the proposed method can be implemented for any application generically. But the other characteristics have also been performed and it outperforms the existing method.

4.23 PWM SWITCHING FOR SR MOTOR CONTROL

Pulse-Width Modulation (PWM) is further “feed-forward” approach to motor control. The duty cycle (the ratio of the “on-time: to the “off-time” during a given switching stage) of the phase voltage is tuned to control the average phase voltage applied to the phase, This average voltage then determines the phase current.

PWM is fairly less responsive to changes in phase current than hysteresis control, but it is well suited for fixed speed or variable speed drives where the dynamic load changes are not extreme. PWM uses a very high switching frequency to lower the current ripple and to permit smaller sized energy storage elements to be used (inductors and capacitors reduce in size with increased operating frequency for a specified current and voltage rated device).

As well, higher switching frequencies agree to the harmonic content to occur at fixed multiples of the fundamental switching frequency (at very high frequencies, in other words), allowing easy harmonic filter implementation (again with smaller components). PWM contain two variations on switching methods. SRM control circuit have two switches per phase, which are controlled separately.

Soft-chopping PWM maintain one of the phase segments switches on while switching the other switch on and off in order to control the current. The resulting rate of change of current is measured; voltage stresses, noise, and current ripple are minimal.

Hard chopping PWM keeps switching both of the switches on and off at the same time, producing more quick changes in the phase current and potentially faster
response time. The disadvantage of hard-chopping is higher stress on the components, higher current ripple, torque ripple and acoustic noise.

PWM Switching is the control method by which the transistor operates in either saturation or cut off region. When the transistor is saturated for a long time, is known as ON-time, if it is driven to the cut off region then it is called as OFF-time. During the ON-time $V_{CE}$ is very small approximately zero for simple circuits. During the OFF-time the transistor is cut off, which results in open circuit between the emitter and collector. Thus transistor operates as a switch during ON and OFF time. The sum of ON and OFF times is the period (T) of the switch as shown in Fig. 4.24

![Transistor ON and OFF times](image)

Fig. 4.24: Transistor ON and OFF times

The switching action repeats for every T. The reciprocal of the period is the switching frequency, the rate at which the transistor is switched

$$f = \frac{1}{T}$$  \hspace{1cm} 4.30

The voltage across the load is not constant and has DC components. The average value of the output voltage is determined from,

$$V_o = \frac{1}{T} \int_{t_{on}} V_s dt$$  \hspace{1cm} 4.31

Hence,

$$V_o = \left( \frac{t_{on}}{T} \right) V_s$$  \hspace{1cm} 4.32
The equation (4.32) reveals that the average output voltage is proportional to the ON time. The longer the switch stays closed, the larger the DC component of the output voltage. The ratio of ON time to the switching period is called the duty ratio.

\[ D = \frac{t_{on}}{T} \]  \hspace{1cm} (4.33)

Average Output voltage is given by,

\[ V_o = D V_s \]  \hspace{1cm} (4.34)

As \( 0 \leq D \leq 1 \), the average voltage varies from zero to source voltage.

The PWM method is also effective as a linear controller of the average power delivered to the load. The RMS value of the output voltage is determined from,

\[ V_{RMS} = \sqrt{\frac{1}{T} \int_0^{t_{on}} V_s^2 dt} \]  \hspace{1cm} (4.35)

From equation (4.14) in (4.15)

\[ V_{RMS} = V_s \sqrt{D} \]  \hspace{1cm} (4.36)

The average power absorbed in the PWM controlled load is

\[ P = D \frac{V_s^2}{R} \]  \hspace{1cm} (4.37)

Since we have derived the average power equation (4.37) it is necessary to control the average torque of the motor.
4.23.1: Limitations of simple PWM controlled Switching

PWM controlled circuit performs linear control over the load power and is effective in the case of incandescent lighting and LED displays as well as to the energy applied to heating elements. These devices suffer no problems when the current is periodically interrupted. However in case of logic circuits it results in complete loss of functionality. In order to provide effective constant output the PWM circuit is modified with energy storage components which will sustain the output voltage during OFF time in power switch.