**NEURO FUZZY CONTROLLER BASED DIRECT TORQUE CONTROL**

**FOR SRM DRIVE**

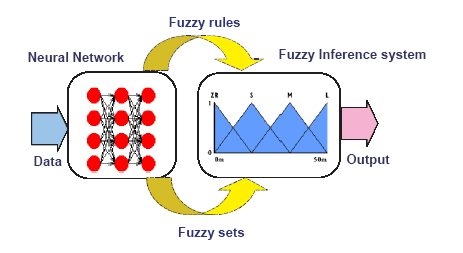
**Murugan.M**

***Abstract -*The integration of neural networks and fuzzy inference system could be formatted into three main categories: cooperative, concurrent and integrated neuro-fuzzy models namely fuzzy associative memories fuzzy rules extraction using self-organizing maps and systems capable of learning fuzzy set parameters. Mamdani and Takagi-Sugeno type integrated neuro-fuzzy systems were further introduced with a focus on some of the salient features and advantages of the different types of integrated neuro-fuzzy models that have been evolved during last decade.This work focus on the implementation of integrated neuro-fuzzy systems also called hybrid controllers. The Mamdani and Sugeno hybrid controllers are incorporated along with direct torque control to generate more accurate voltage space vectors. This helps in controlling the torque ripple and reduce its amplitude to a great extend. The detail description is given in the following sections. MATLAB design is done with the help of MATLAB Compilers from Math works and the results prove the better control of SRM with reduced torque and flux ripples.**

**I.INTRODUCTION**

In an integrated model, neural network learning algorithms are used to determine the parameters of fuzzy inference systems. Integrated neuro-fuzzy systems share data structures and knowledge representations, Bose (1994). A fuzzy inference system can utilize human expertise by storing its essential components in rule base and database; hence perform fuzzy reasoning to infer the overall output value. The derivation of *if then* rules and corresponding membership functions depends heavily on the priori knowledge about the system under consideration. However there is no systematic way to transform the experiences and knowledge of human experts to the knowledge base of a fuzzy inference system. There is also a need for adaptability or some learning algorithms to produce outputs within the required error rate. On the other hand, neural network learning mechanisms does not rely on human expertise. Due to the homogenous structure of neural network, it is hard to extract structured knowledge from either the weights or the configuration of the network. The weights of the neural network represent the coefficients of the hyper-plane that partition the input space into two regions with different output values. If we can visualize this hyper-plane structure from the training data then the subsequent learning procedures in a neural network can reduce. However, in reality, the prior knowledge is usually obtained from human experts, as it is most appropriate to express the knowledge as a set of fuzzy if-then rules, and encode into a neural network. The Figure1 shows the model of Neuro Fuzzy controller.

To a large extent, the drawbacks pertaining to these two approaches seem complementary. Therefore, it seems natural to consider building an integrated system combining the concepts of FIS and ANN modeling. A common way to apply a learning algorithm to a fuzzy system is to represent it in a special neural network like architecture Grabowski (1999). However the conventional neural network learning algorithms (gradient descent) cannot be applied directly to such a system as the functions used in the inference process are usually non differentiable. Functions in the inference system are not using the standard neural learning algorithm. This work discusses how to model integrated neuro-fuzzy systems implementing Mamdani and Takagi-Sugeno FIS.

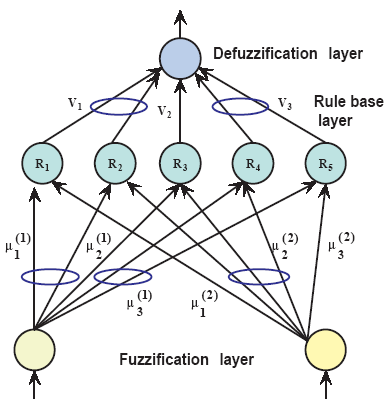


**Figure 1 Neuro Fuzzy Model**

**II.NEURO-FUZZY CONTROLLER (NEFCON)**

The learning algorithm defined for NEFCON is able to learn fuzzy sets as well as fuzzy rules implementing a Mamdani type FIS. This method also use reinforcement learning but need a previously defined rule base. Figure 2 illustrates the basic NEFCON architecture with 2 inputs and five fuzzy rules. The inner nodes R1……., R5 represent the rules, the nodes ξ1, ξ2, the inputs, η the output values, and μr, Vr is the fuzzy sets describing the antecedents and consequents respectively. In contrast to neural networks, the connections in NEFCON are weighted with fuzzy sets instead of real numbers. Rules with the same antecedent use, so called shared weights, which are represented by ellipses drawn around the connections. They ensure the integrity of the rule base. The knowledge base of the fuzzy system is implicitly given by the network structure. The input units assume the task of fuzzification interface, the interference logic is represented by the propagation functions, and the output unit is the defuzzification interface. The learning process of the NEFCON model can be divided into two main phases. The first phase is designed to learn the rule base and the second phase optimizes the rules by shifting or modifying the fuzzy sets of the rules.

Two methods are available for learning the rule base. Incremental rule learning is used when the correct output is not known and rules are created based on estimated output values. As the learning progresses, more rules are added according to the requirement. For decremented rule learning, initially rules are created due to fuzzy partitions of process variables and unnecessary rules are eliminated in the course of learning. Decrement rule learning is less efficient compared to incremental approach. However it can be applied to unknown processes without difficulty, and there is no need to know or to guess an optimal output value. Both phases use a fuzzy error E, which describes the quality of the current system state, to learn or to optimize the rule base. To obtain a good rule base it must be ensured that the state space of the process is sufficiently covered during the learning process. Due to the complexity of the calculations required, the decrement learning rule can only be used, if there are only a few input variables with not too many fuzzy sets. In larger systems, the incremental learning rule will be optimal. Prior knowledge whenever available could be incorporated to reduce the complexity of the learning. Membership functions of the rule base are modified according to the Fuzzy Error Back propagation (FEBP) algorithm. The FEBP algorithm can adapt the membership functions, and can be applied only if there is already a rule base of fuzzy rules. The idea of the learning algorithm is identical: increase the influence of a rule if its action goes in the right direction (rewarding), and decrease its influence if a rule behaves counter productively (punishing). If there is absolutely no knowledge about initial membership function, a uniform fuzzy partition of the variables should be used.



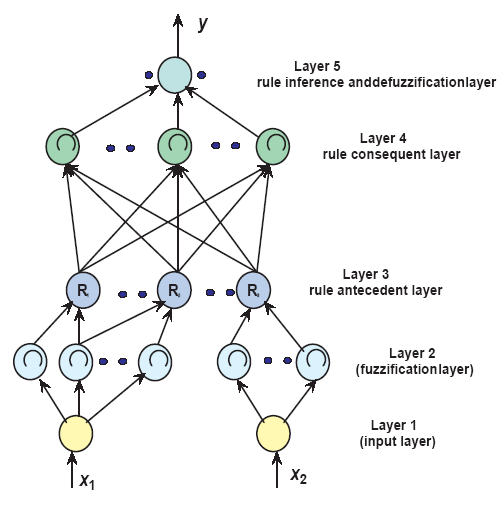
**Figure 2 Architecture of NEFCON**

**MAMDANI INTEGRATED NEURO-FUZZY SYSTEM**

A Mamdani neuro-fuzzy system uses a supervised learning technique (back propagation learning) to learn the parameters of membership functions. Architecture of Mamdani neuro-fuzzy system is illustrated in Figure 3. The detailed functions of each layer are as below.

***Layer 1*** (input layer): No computation is done in this layer. Each node in this layer, which corresponds to one input variable, only transmits input values to the next layer directly. The link weight in layer 1 is unity.

***Layer 2*** (Fuzzification layer): Each node in this layer corresponds to one linguistic label (PE,ZE,NE…) to one of the input variables in layer 1. In other words, the output link represents the membership value, which specifies the degree to which an input value belongs to a fuzzy set and is calculated in layer 2. A clustering algorithm will decide the initial number and type of membership functions to be allocated to each of the input variable. The final shapes of the MFs will be fine tuned during network learning.

****

**Figure 3 Mamdani Neuro-Fuzzy System**

***Layer 3*** (rule antecedent layer): A node in this layer represents the antecedent part of a rule. Usually a T-norm operator is used in this node. The output of layer 3 node represents the firing strength of the corresponding fuzzy rule.

***Layer 4*** (rule consequent layer): This node basically has two tasks. To combine the incoming rule antecedents and determine the degree to which they belong to the output linguistic value. The number of nodes in this layer will be equal to the number of rules.

***Layer 5*** (combination and defuzzification layer): This node does the combination of all the rules consequents using a T-conorm operator and finally computes the crisp output after defuzzification.

**III. TAKAGI-SUGENO INTEGRATED NEURO-FUZZY SYSTEM**

Takagi-Sugeno Neuro-Fuzzy systems make use of a mixture of backpropagation to learn the membership functions and least mean square estimation to determine the coefficients of the linear combinations in the rule’s conclusions. A step in the learning procedure got two parts: In the first part the input patterns are propagated, and the optimal conclusion parameters are estimated by an iterative least mean square procedure, while the antecedent parameters (membership functions) are assumed to be fixed for the current cycle through the training set. In the second part the patterns are propagated again, and in this epoch, back propagation is used to modify the antecedent parameters, while the conclusion parameters remain fixed. This procedure is then iterated. The detailed functioning of each layer (as depicted in Figure 4) is as follows. Layers 1, 2 and 3 function the same way as Mamdani FIS.

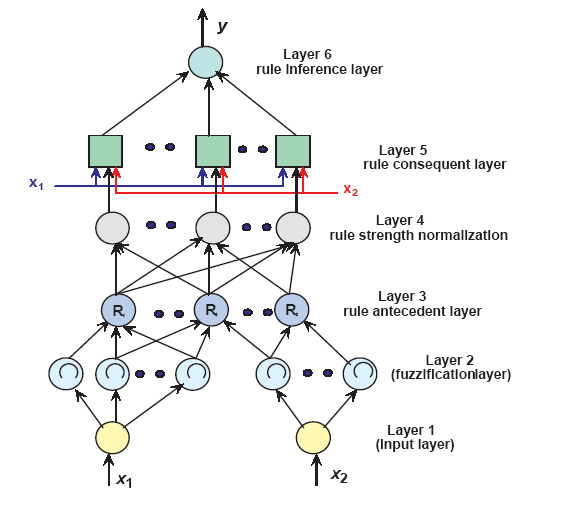
***Layer 4*** (rule strength normalization): Every node in this layer calculates the ratio of the ith rule’s strength to the sum of all rules firing strength The Figure 4 shows the architecture of Takagi-Sugeno Neuro-fuzzy systems.

 (1)

***Layer 5*** (rule consequent layer): Every node I in this layer is with a node function

 (2)

where  is the output of layer 4, and {pi, qi, ri) is the parameter set. A well-established way is to determine the consequent parameters using the least mean squares algorithm.



**Figure 4 Takagi Sugeno Neuro-Fuzzy System**

***Layer 6*** (rule inference layer): The single node in this layer computes the overall output as the summation of all incoming signals.

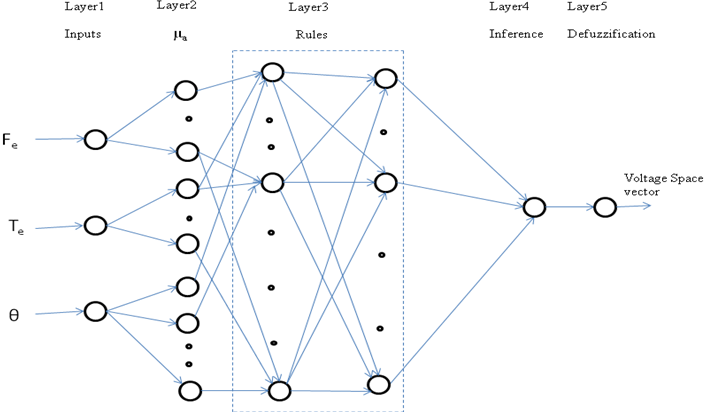
 (3)

**PROPOSED HYBRID NEURO-FUZZY CONTROLLER FOR DTC OF SRM**

Fuzzy logic and neural networks are complementary technologies in the design of intelligent systems. Artificial neural networks (ANN) are low-level computational algorithms that offer good performance with sensory data, while fuzzy logic deals with reasoning in a higher level than ANN. Mamdani and Takagi-Sugeno method, hybrid neuro-fuzzy systems were introduced with a focus on some of the salient features and advantages of the different types of integrated neuro-fuzzy models.

**III. PROPOSED MAMDANI NEURO-FUZZY DTC CONTROL**

In this DTC, Mamdani neuro-fuzzy controller is used to generate the voltage space vectors to control the switching of converter circuit in SRM. The third and fourth layer is combined as one single layer for rules. This system has three inputs - torque error, flux error and sector information. Triangular membership function is used for the antecedent part. The torque and flux errors are got as the difference between the reference value and actual value. The output of the control is the switching state (0,1,2,3,4,5,6) for the converter circuit. The flux errors (Fe) use three linguistic values: positive error (PE), zero error (ZE) and negative error (NE) with universe of discourse [-0.01 0.01]. For the torque error (Te) the universe of discourse is [-0.1 0.1] with three fuzzy aggregations positive (P), negative (N) and zero (Z). The flux linkage angle θ is divided into six aggregations of 60˚ each. The Mamdani neuro-fuzzy system is formed with the following layers as shown in Figure 5



**Figure 5 Neuro-Fuzzy Controller**

Here *Ii, Ij, Ik, Il*represents the inputs to the neuron and *Oi, Oj, Ok, Ol*represents the output of the neurons in layers 1,2, 3 and 4 respectively.

***First layer***: Comprises of the three inputs Fe, Te and θ. The *Ii1*are the inputs given (Xi) and Oi1 as the outputs of neurons in the first layer.

Oi1 = Ii1 = Xi (4)

***Second layer:*** This is the fuzzy layer that grades the membership function μA(Xi) and leads to 12 connections with value of one as the weights (wij) between the first layer and the second layer.

Ij2 = wij Oi1 (5)

Oj2 = µA(Xi) (6)

***Third layer:*** This layer defines the grade of corresponding rules of a fuzzy system with 54 connections combining the rule antecedent and consequent layer. Weights defined are one for connections between third and fourth layer. ****** (7)

Ok3 = Ik3 (8)

***Fourth layer:*** This fuzzy decision layer sums all the inputs to this layer.

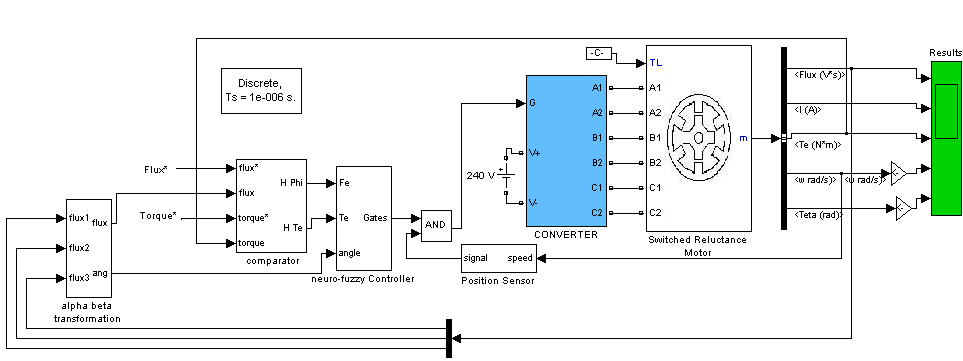
 (9)

***Fifth layer:*** It does defuzzification by adapting to the greatest grade of membership function and gives the single output as the choice of voltage space vector.

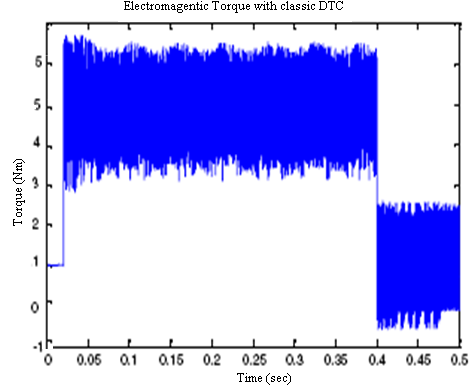
**Simulation Results**

To simulate the system a Matlab/Simulink closed loop model was constructed for the SRM and the Mamdani neuro-fuzzy control system as in Figure 5.6. The motor parameters such as torque, phase flux and position are obtained from the 3Φ SRM. The three phase-flux vectors are transformed on to a stationary orthogonal α-β reference frame to calculate the net flux. Artificial neuro-fuzzy inference system for voltage space vector generation is constructed. This controller combines *fuzzy* logic and artificial neural networks *(ANN)* with DTC for decoupled flux and torque control. Based on the present position of motor, torque error (Te) and flux error (ψe) the optimal selection of voltage space vector is done with the help of intelligent control. Thus the converter switches and hence the motor is controlled by DTC scheme. In this simulation test, the motor reference flux and torque were maintained at a constant of 0.3Wb and 5 Nm respectively.

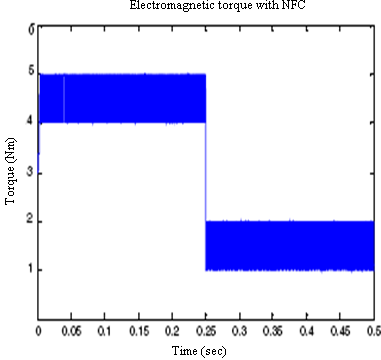
It is seen clearly that in steady state the neuro-fuzzy based DTC leads to regularly spaced flux linkage and currents. The individual flux linkages leads to smooth constant amplitude flux vector in the stator air gap. The torque results in Figure 8 shows lower ripple content and constant amplitude nature. The FFT analysis in Figures 9 as done with MATLAB shows lower ripples of torque for intelligent control of three-phase motor when compared to that of the conventional DTC of three-phase motor.



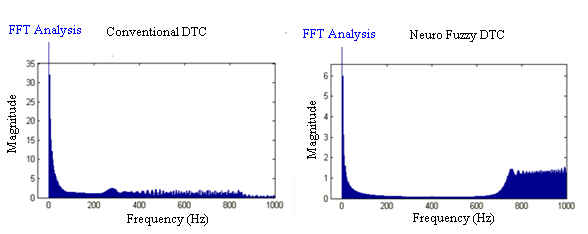
**Figure 6 Simulation Block Diagram of Neuro-Fuzzy Controller for Direct Torque Control Algorithm**



**Figure 7 Simulation Torque Characteristics for Classical DTC**

****

**Figure 8 Simulation Torque Characteristics of Neuro-Fuzzy Controller**



**Figure 9 FFT Analysis for Classical and Neuro Fuzzy DTC**

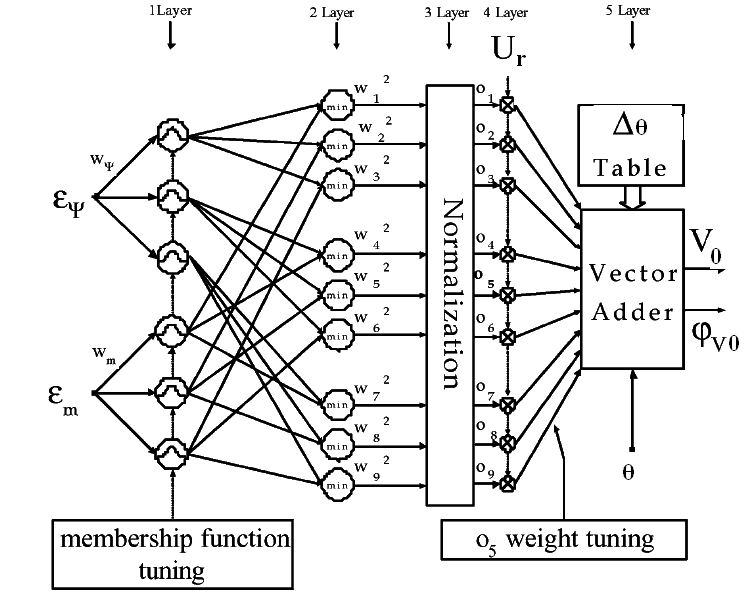
**PROPOSED SUGENO NEURO-FUZZY SYSTEM**

Already, fuzzy based SVM-DTC was proposed by Satean Tunyasrirut (2005). The proposed system uses Sugeno neuro-fuzzy system, an Adaptive NF Inference System (ANFIS) for voltage space-vector generation with SVM. It combines fuzzy logic and artificial neural networks for decoupled flux and torque control. In this scheme, shown in Figure 5.10 the error signals are delivered to the NF controller, which is also entered by the actual position of the stator flux vector. The NF controller determines the stator voltage command vector in polar coordinates for the SVM block. The scheme is characterized by a simple self-tuning procedure and good steady-state and dynamic performance. This system has two inputs: torque error, flux error which is got as the difference between the reference value and actual value.

Takagi-Sugeno Neuro-Fuzzy systems make use of a mixture of back propagation to learn the membership functions and least mean square estimation to determine the coefficients of the linear combinations in the rule’s conclusions. A step in the learning procedure got two parts: In the first part the input patterns are propagated, and the optimal conclusion parameters are estimated by an iterative least mean square procedure, while the antecedent parameters (membership functions) are assumed to be fixed for the current cycle through the training set.

In the second part the patterns are propagated again, and in this epoch, back propagation is used to modify the antecedent parameters, while the conclusion parameters remain fixed. This procedure is then iterated.

The flux errors (Fe) use three linguistic values: positive error (PE), zero error (ZE) and negative error (NE) with universe of discourse [-0.01 0.01]. For the torque error (Te) the universe of discourse is [-0.1 0.1] with three fuzzy aggregations positive (P), negative (N) and Zero (Z) (Table.2).The detailed functioning of each layer (as depicted in Figure 5.11) is as follows.



**Figure 10 ANFIS Controller**

Here *Ii, Ij, Ik, Il*represents the inputs to the neuron and *Oi, Oj, Ok, Ol*represents the output of the neurons in layers 1,2, 3 and 4 respectively.

**First layer:** In the first layer of the NF structure, flux error and torque error are multiplied by respective weights and are each mapped through three fuzzy logic triangular membership functions.

Ii1 = XiWi

Oi1 = µA (Ii1)  (10)

**Second layer:** This layer calculates the minimum value of the inputs

Ij2 = wij Oi1 (11)

Oj2 = min (Ij2) = wk  (12)

**Third layer:** This layer normalizes each input with respect to the others. The kth node output is the kth input divided by the sum of all the other inputs.

**** (13)

**Fourth layer:** This layer output is a linear function of the input and the ANFIS signals. The weight is the weight of both the incremental angle and the amplitude of desired reference voltage vector, so that:

| V0 | = Ok3 | Ur| (14)

 (15)

where, | V0 | is the amplitude of desired reference voltage, | Ur| is the maximum reference voltage amplitude,  is the angle of the desired reference voltage,  is the incremental angle and  is the actual angle of the stator flux vector.

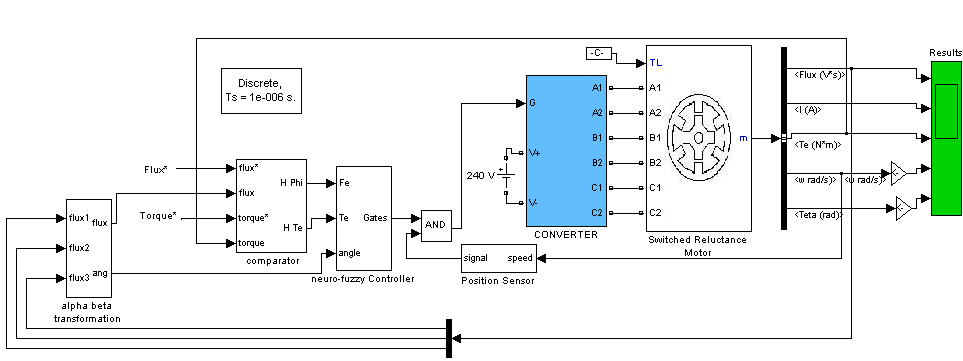
**Fifth layer:** This layer sums all the incoming signals.

 (16)

The components of the desired reference voltage vector are added to each other, which is delivered to the space vector modulator which calculates the switching condition of the converter. The network is tuned by least square estimation for output membership function and back propagation algorithm for output and input membership function.

**Simulation Results**

A Matlab/Simulink closed loop model was constructed for the SRM and Sugeno type hybrid neuro-fuzzy control system as in Figure 11. The motor parameters such as torque, phase flux and position are obtained from the 3Φ SRM.

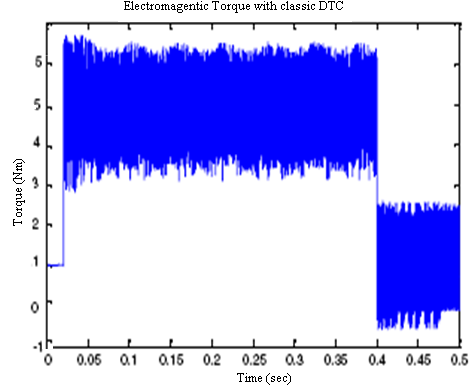


**Figure 11 Simulation Diagram of ANFIS Based SVM-DTC Controller**

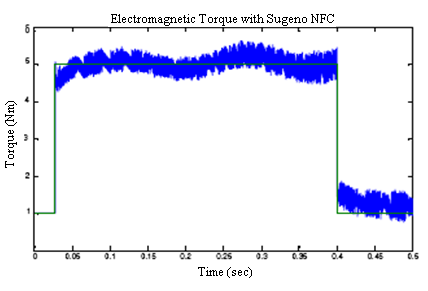
Adaptive Neuro-Fuzzy Inference System (ANFIS) for voltage space vector generation is constructed. This controller combines *fuzzy* logic and artificial neural networks *(ANN)* with DTC for decoupled flux and torque control.The torque and flux errors are generated based on the difference between reference and actual values of torque and stator flux respectively. Sampled fluxand torque errors, multiplied by weights, are delivered to the three membership functions in both inputs. Based on the present position of motor, torque error and flux error the optimal selection of voltage space vector is done with the help of ANFIS controller. Thus the converter switches and hence the motor is controlled by DTC scheme.

In this simulation test, the motor reference flux and torque were maintained at a constant of 0.3Wb and 5Nm respectively. The individual flux linkages leads to smooth constant amplitude flux vector in the stator air gap. The torque results in Figures 12-13 shows lower ripple content and constant amplitude nature. It is seen clearly from the spectrum analysis that in steady state the neuro-fuzzy based DTC leads to lower ripple and hence better performance.

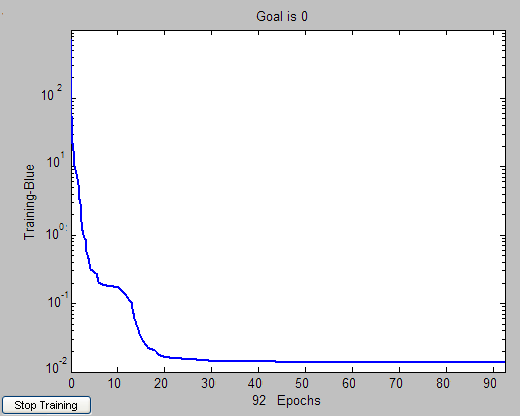
The number of epochs was 100 for training. The number of MFs for the input variables Teand Feis 3 and 3, respectively. The number of rules is then 9. The triangular MF is used for the two input variables. It is clear that the triangular MF is specified by two parameters. Therefore, the ANFIS used here contains a total of 39 fitting parameters, of which 12 (3\*2+ 3\* 2 = 12) are the premise parameters and 27 (3\*9=27) are the consequent parameters. The training and testing root mean square (RMS) errors obtained from the ANFIS are 4.7 × 10-6 and 5.3 × 10-6 respectively.



**Figure 12 Torque Developed in Conventional DTC**



**Figure 13 Torque Developed in Sugeno Neuro-Fuzzy Controller**



**Figure .14 Reduced Torque Errors Due to Training**

**CONCLUSION**

This paper explains the neuro -fuzzy logic based control. Mamdani NFC and Sugeno NFC based direct torque control are implemented for the motor. The adaptive neuro-fuzzy inference based SVM-DTC technique is implemented through Sugeno type control. The simulation results validate by producing less torque ripples for SRM drive.

REFERENCES

1. Bo Zhou and Xiao Fei Jing (2008), ‘Application of Particle Swarm Optimization on DTC for Induction Motors’, Fourth International Conference on Natural Computation pp. 472-476.
2. Cheng-Zhi Cao, Guang-Hua Wei, Qedong Zhang and Xin Wang (2004), ‘Optimization Design Of Fuzzy Neural Network Controller In Direct Torque Control System’ Proceedings of the Third International Conference on Machine Learning and Cybemetics, Shanghai, 26-29, pp. 378-382.
3. Grzesiak L.M., Meganck V., Sobolewski J. and Ufnalski B. (2007), ‘Genetic Algorithm for Parameters Optimization of ANN-based Speed Controller ‘EUROCON 2007 The International Conference on ‘Computer as a Tool’ Warsaw, September 9-12 pp.1700-1705.
4. Jinupun P. and Luk P.C.K. (1998), ‘Direct torque control for sensorless switched reluctance motor drives’, in Proc. 7th Int. Conf. Power Electron. Variable Speed Drives, pp. 329-334.
5. Lee B.S. (2000), ‘Linear Switched Reluctance Machine Drives with Electromagnetic Levitation and Guidance Systems’, Ph.D. thesis, The Bradley Department of Electrical and Computer Engineering, Virginia Tech., Blacksburg.
6. Liping Fan and Bin Li (2007), ‘Application of Fuzzy Neural Network in Direct Torque Control System’, IEEE International Conference on Control and Automation Guangzhou, CHINA - May 30 to June 1,   
   pp. 2186-2191.
7. Luis Romeral, Antoni Arias, Emiliano Aldabas and Marcel. G. Jayne (2003), ‘Novel Direct Torque Control (DTC), Scheme with Fuzzy Adaptive Torque-Ripple Reduction’ IEEE Transactions On Industrial Electronics, Vol. 50, No. 3, pp. 487-492.
8. San Dan, He Yikang and Zhi Dawei (2003), ‘Direct torque control of a permanent magnet synchronous motor base on fuzzy logic’, Trans. china Elec. Society, Vol.18, pp. 33-38.
9. Stankovic A.M., Tadmor G., Coric Z. J. and Agirman I., (1999), ‘On torque ripple reduction in current-fed switched reluctance motors’, IEEE Trans. Ind. Electron, Vol. 46, pp. 177-183.
10. Toufouti R., Meziane S. and Benalla H. (2006), ‘Direct Torque Control for Induction Motor Using Fuzzy Logic’, ICGST Trans. on ACSE, Vol.6, No.2, pp. 17-24.
11. Vedagarbha P., Dawson D.M. and Rhodes W. (2007), ‘An adaptive controller for a general class of switched reluctance motor models’, In Proc. 13th World Congr., Vol. F, pp. 471-476.
12. Xiying Ding, Qiang Liu, Xiaona Ma, Xiaoran He and Qing Hu (2007), ‘The Fuzzy Direct Torque Control of Induction Motor Based on Space Vector Modulation’, Third International Conference on Natural Computation (ICNC 2007), pp 125-131.