CHAPTER 5

COMPARISON OF ANN AND ANFIS BASED
ROTOR POSITION ESTIMATION

5.1 INTRODUCTION

In this Chapter, ANN based rotor position estimation algorithm is compared with ANFIS based rotor position estimation algorithm particularly with respect to execution time by implementing them on DSP, TMS320F2812 using CCS software. For the proposed comparison work, feed forward networks are used. The first layer has weights coming from the input. Each subsequent layer has a weight coming from the previous layer. All layers have biases. The last layer is the network output. Training is done with the TRAINLM training function which is a network training function that updates weight and bias values according to Levenberg-Marquardt optimization. The activation functions for the hidden layer neurons are tansig functions. The activation function for the neuron in the output layer is purelin function. Performance is measured in terms of mean square error (MSE).

The static flux linkage-current-rotor position characteristics of the solid rotor SRM are mapped using the feed forward networks. Flux linkage and current are used as inputs and rotor position is taken as output of ANN structure. The nonlinear flux linkage-current-rotor position characteristics can be obtained from finite element analysis (FEA), analytical model or from real time experiments. For this comparison work, the non-linear characteristics are obtained from real time experiments.
The test motor has to be locked at different rotor positions using the mechanical fixing arrangement specially designed for locking purpose. At locked rotor condition, DC voltage is applied across the windings for short duration and waveforms of phase currents and phase voltages are recorded using DSO. The mechanical fixing arrangement used for rotor locking purpose is shown in Figure 5.1.

![Figure 5.1 Mechanical fixing arrangement designed to lock the rotor at different positions](image)

The following data, time, phase voltage and phase current at different rotor positions ranging from 0 degree (aligned) to 30 degree (unaligned) are measured using DSO. To measure phase voltage and phase current, LEM make closed loop voltage sensors (LV20-P) and current sensors (LA 25-NP) are used.

Phase A is considered as reference phase. The captured data are transferred to PC for further processing and to obtain the static flux linkage-current-rotor position characteristics of the solid rotor SRM. The Microsoft excel sheet is used to find the flux linkage value by trapezoidal method as discussed by Xue et al. (2004).
\[ \psi_k(n) = \psi_k(n-1) + \Delta t(v_k(n-1) + v_k(n)) - (R_k*i_k(n-1) + i_k(n))/2 \]  

where, \( R_k \) is 5.4 ohms for the test motor. \( \psi_k(n) \) is the \( n \)th flux linkage sample of \( k \)th phase, \( v_k(n) \) is the \( n \)th voltage sample of \( k \)th phase, \( i_k(n) \) is the \( n \)th current sample of \( k \)th phase and \( \Delta t \) is the sampling interval. Equation (5.1) is used for obtaining flux linkage values. For further processing, Matlab version 7.3 is used.

Figure 5.2 shows the stator voltage pulse and stator current pulse measured at phase A aligned position using DSO. The recorded voltage and current pulses shown in Figure 5.2 are used for calculation of phase flux linkage. From DSO, the recorded samples can be transferred to PC.

![Figure 5.2 Stator voltage pulse and current pulse at phase A aligned position](image)

(Blue line shows stator voltage pulse and yellow line shows current pulse)

Figure 5.3 shows the static flux linkage-current-rotor position characteristics of the test motor (0 degree-aligned position, 30 degree-
unaligned position). The flux linkage characteristics are linear for low current values. The flux linkage characteristics are linear at and near unaligned rotor position. Using curve fitting tool box in Matlab, the static flux linkage-current-rotor position characteristics shown in Figure 5.3 have been obtained.

Figure 5.4 shows the structure of 2-5-1 ANN. The structure of 2-5-1 ANN can be detailed as follows, the 2-5-1 ANN structure has two inputs and a single hidden layer. Five hidden neurons are there in that single hidden layer. There is a single output from the structure. w11, w21, w12, w22, w13, w23, w14, w24, w15, w25, w41, w42, w43, w44 and w45 are weight values. B1, B2, B3, B4, B5 and B6 are bias values.

![Figure 5.3 Static flux linkage-current-rotor position characteristics (0 degree-aligned position, 30 degree-unaligned position)](image-url)
Figure 5.4 Structure of 2-5-1 ANN

Input \( x = \) [current, flux linkage], \( F_1, F_2, F_3, F_4 \) and \( F_5 \) are tanH functions. The bipolar tanH function can be expressed as follows, \( \text{tanH}(n) = \frac{2}{1+\exp(-2n)} - 1 \). Output from ANN structure is \( Y \) (rotor position).

5.2 ANN BASED ROTOR POSITION ESTIMATION

The flux linkage- current data are used as inputs and rotor position is taken as output for the following ANN structures, 2-2-1, 2-3-1, 2-5-1 ANN structures. From the literature, it is evident that accuracy of ANN structure is dependent on the number of neurons in the hidden layer. But there is limitation in increasing the number of neurons in hidden layer. It is limited by the execution time of the ANN based rotor position estimation algorithm on DSP. As execution time limits the accuracy of the rotor position estimation algorithm, execution time is considered as prime criteria for comparison.

Using Matlab version 7.3 software, ANN models are obtained. For training, 847 data collected from real time experiments are used. The number
of epochs in training is 10000. The simulink models obtained from Matlab software is implemented on DSP TMS320F2812 using CCS software. The source program is written in C file.

The number of instruction cycles required to implement each simulink model to estimate the rotor position from the phase current and flux linkage inputs are obtained using CCS. In earlier works the time needed to execute a sigmoid function is not discussed. In some of the earlier works, only lookup table method is used.

The analysis of execution time is done with a specific input data (current is $1.4A$ and flux linkage is $0.74wb$). To implement the following bipolar sigmoid function \((\text{tansig}(n)=2/(1+\exp(-2*n))-1)\) on DSP TMS320F2812, for the specified input data and weight values (when \(n=17.73\)), the number of instruction cycles needed is \(3103\) When \(n\) value is \(64.55\), the number of instruction cycles needed is \(3106\). The clock frequency of DSP, TMS320F2812 is \(150MHz\).

In Figure 5.5, the pulse width shows the time needed to execute the following unipolar sigmoid activation function, \(y = 1/(1+\exp(-1))\). It needs about \(19.56\ \mu s\).
Figure 5.5  Execution time of unipolar sigmoid function

(for the specific input data, Pulse width is 19.56 µs)

A total of 13016 instruction cycles are needed to execute the 2-3-1 ANN structure based rotor position estimation algorithm. As the execution time varies slightly with respect to input data, comparison is done for the specific set of input data. To execute a single bipolar sigmoid activation function, an approximate minimum of 3000 cycles are required when it is executed in C file.

From Table 5.1, it is clear that increasing the number of neurons in hidden layer is increasing the execution time. There is an increase of 26.68 µs for a single hidden neuron increase, when 2-2-1 ANN structure is compared with 2-3-1 ANN structure. There is a drastic increase of 146.23 µs for two hidden neurons increase, when 2-3-1 ANN structure is compared with 2-5-1 ANN structure. Thus the number of neurons used in hidden layers affects the sampling frequency. Hence the number of neurons used in hidden layers decides the accuracy of the ANN based rotor position estimation algorithm.
Table 5.1  Comparison of execution time of different ANN structures for the specific input data (implementation using C file on DSP TMS320F2812)

<table>
<thead>
<tr>
<th>Type of ANN Structure</th>
<th>Number of Instruction cycles</th>
<th>Execution time (μs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-2-1 feed forward network</td>
<td>9014</td>
<td>60.093</td>
</tr>
<tr>
<td>2-3-1 feed forward network</td>
<td>13016</td>
<td>86.773</td>
</tr>
<tr>
<td>2-5-1 feed forward network</td>
<td>34950</td>
<td>233</td>
</tr>
</tbody>
</table>

5.3  ANFIS BASED ROTOR POSITION ESTIMATION

A common way to integrate ANN and Fuzzy Inference System (FIS) is to represent FIS in a special ANN architecture. However, the conventional ANN algorithms cannot be applied directly to such a system, because the functions used in there usually non differentiable. This problem can be tackled by using differentiable functions in the FIS or by not using the standard learning algorithms.

As detailed by Ajith Abraham (2001), there are several approaches to integrate ANNs and FISs depending on the application. Some of the major works in this area are GARIC, FALCON, ANFIS, NEFCON, FUN, SONFIN, FINEST, EFuNN, dmEFuNN, evolutionary design of Neuro-Fuzzy systems and many others. Among the various Neuro-Fuzzy models only the hybrid integrated Neuro-Fuzzy model make use of the complementary strength of ANN and FIS( Faa-Jeng Lin and Po-Huan Chou (2009)).
The Neuro-Fuzzy model used in this work is ANFIS, the hybrid technology of integrated Neuro-Fuzzy model and a part of Matlab's Fuzzy Logic Toolbox. ANFIS implements a Takagi-Sugeno FIS and has a five layered architecture. The first hidden layer is for fuzzification of the input variables and T-norm operators are deployed in the second hidden layer to compute the rule antecedent part. The third hidden layer normalizes the rule strengths followed by the fourth hidden layer where the consequent parameters of the rule are determined. Output layer computes the overall input as the summation of all incoming signals.

ANFIS uses a hybrid learning algorithm that combines the back propagation gradient descent and least square methods to create a fuzzy inference system whose membership functions are iteratively adjusted according to a given training set of input and output.

ANFIS in comparison the other ones has high speed of training, the most effective learning algorithm and simplicity of the software. Although ANFIS is one of the first integrated hybrid Neuro-Fuzzy models, surprisingly it is the best function approximator among the several Neuro-Fuzzy models. ANFIS is faster in convergence when compared to the other Neuro-Fuzzy models. Furthermore, ANFIS provides better results when applied without any pretraining.

Most of the Neuro-Fuzzy systems are either based on Takagi-Sugeno or Mamdani type Fuzzy logic controller. Takagi-Sugeno type based Neuro-Fuzzy models are widely used for model-based applications. Takagi-Sugeno type combines the advantages of being general approximators that can reach high accuracy and being easy to interpret, since they are represented in a quite natural way.
The generality of Takagi-Sugeno type makes the data driven identification such systems very complex. Takagi-Sugeno type based Neuro-Fuzzy systems have high performance, but often requires complicated learning procedures and computational expensive. However, Mamdani type based neuro-fuzzy systems can be modeled using faster heuristics but with a low performance. Because of the above said merits, Takagi-Sugeno type neuro-fuzzy system is considered for mapping the nonlinear characteristics of SRM with a single output.

The sugeno type based Neuro-Fuzzy model is used for estimating rotor position from the active phase voltage and current samples. The membership functions values have been optimized by the ANFIS. The system is implemented using a DSP, TMS320F2812. The programming is done in .C source file using CCS software.

The following steps are followed to design and implement ANFIS based rotor position estimation:

1. Obtain Current-Flux linkage-rotor position characteristics of the test motor.
2. Use ANFIS tool box in Matlab to get different ANFIS structures for the corresponding characteristics.
3. Compare the performance of the different ANFIS structures with trained and untrained data.
4. Select the accurate ANFIS structure.
5. Implement the accurate ANFIS structure using .C file and execute the program on DSP.
6. Verify the program for trained and untrained data.
The ANFIS structure shown in Figure 5.6 uses five membership functions for each input. The inputs given to the ANFIS structure are the active phase current and its flux linkage. There are five gbell membership functions for each input. Hence ten input membership functions are used. Twenty-five rules are framed. The total linear output membership functions are 25. The corresponding output is the rotor position.

![Figure 5.6 Structure of ANFIS with two inputs and five membership functions for each input](image)

If the ANFIS structure uses two membership functions for each input, then a total of four input membership functions are used. Four rules are framed. The total numbers of linear output membership functions are four. The structure with increased membership functions is accurate when the membership functions used in the two different ANFIS structures are of same type. For example, an ANFIS structure with total ten input gbell membership functions is more accurate than an ANFIS structure with total four input gbell membership functions. But the number of computations involved in the implementation of the accurate ANFIS structure on DSP is more.
It is verified that the ANFIS structures with gbell type input membership functions have produced accurate results compared to other ANFIS structures with triangular and gauss type input membership functions for the nonlinear characteristics of the test motor.

A total of 8228 cycles are needed to execute a single gbell type input membership function (for the specific input data). The execution time for a single gbell type input membership function is 54.853 $\mu$s on DSP TMS320F2812 (for the specific input data set).

From Table 5.2, it is clear that increasing the number of rules by increasing the number of input membership functions increases the execution time. The error is minimized only when the numbers of rules are increased. For low speeds, ANFIS structure with more rules can be applied to estimate rotor position and at high speeds, ANFIS structure with less number of rules can be applied to estimate rotor position.

<table>
<thead>
<tr>
<th>Number of rules</th>
<th>Number of Instruction cycles</th>
<th>Execution time ((\mu)s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>38358</td>
<td>255.72</td>
</tr>
<tr>
<td>16</td>
<td>78557</td>
<td>523.713</td>
</tr>
<tr>
<td>25</td>
<td>119260</td>
<td>795.067</td>
</tr>
</tbody>
</table>

5.4 COMPARISON OF AI TECHNIQUES

To compare ANN and ANFIS based rotor position estimation algorithms with respect to execution time, the trained structures of ANN/ANFIS are implemented on DSP TMS320F2812 using .C file.
The details of execution time, Mean square error (MSE), error range and maximum sampling frequency of various AI techniques are tabulated in Table 5.3. If the execution time of the remaining part of the program except the rotor position estimation algorithm is considered as 40\(\mu\)s, then the maximum sampling frequency allowed is 10 kHz for 2-2-1 ANN structure. Mean square error is 0.4575 for 2-2-1 ANN structure.

**Table 5.3 Comparison of AI techniques (For the specific input data)**

<table>
<thead>
<tr>
<th>AI Structure</th>
<th>Execution time ((\mu)s)</th>
<th>MSE</th>
<th>Error range (mech. deg.)</th>
<th>Maximum sampling frequency (kHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-2-1 feed forward network</td>
<td>60.093</td>
<td>0.4575</td>
<td>-2.3546 to 1.2131</td>
<td>10</td>
</tr>
<tr>
<td>2-3-1 feed forward network</td>
<td>86.773</td>
<td>0.312</td>
<td>-1.5909 to 2.2194</td>
<td>7.89</td>
</tr>
<tr>
<td>2-5-1 feed forward network</td>
<td>233</td>
<td>0.08553</td>
<td>-0.7654 to 1.1863</td>
<td>3.66</td>
</tr>
<tr>
<td>ANFIS structure with 4 rules and 4 gbell input membership functions</td>
<td>255.72</td>
<td>0.7774</td>
<td>-1.9688 to 2.1268</td>
<td>3.38</td>
</tr>
</tbody>
</table>

The sampling frequency will be very much reduced for ANFIS structure based position estimation algorithm. The maximum sampling frequency allowed is about 3.38 kHz for ANFIS structure with 4 rules when it is implemented using .C file. Mean square error is 0.7774 for ANFIS structure with 4 rules.
In Table 5.3, the different ANN structures are compared with ANFIS structure with respect to execution time, Mean Square Error, error range and maximum sampling frequency. From Table 5.3, it is clear that ANN based rotor position estimation is better than ANFIS structure based position estimation with respect to execution time. It should be noted that there will be slight variations in execution time with respect to input data. Hence the comparison is done for the specific set of input data.

5.5 CONCLUSION

The key conclusions of this Chapter are listed as below:

1) The experimental procedure to obtain static flux linkage-current-rotor position characteristics of the test motor is explained.

2) The structure of ANN and ANN based rotor position estimation are discussed. The execution time of various ANN structure based rotor position estimation algorithms for the test motor, when the algorithms are executed on DSP TMS320F2812 using .C file is clearly detailed. The comparison is done for the specific set of input data. The time needed to execute unipolar and bipolar sigmoid activation functions is discussed. This idea encourages the implementation of these activation functions using .C file. Hence lookup table method can be avoided which has limitation over input data.

3) The structure of ANFIS and ANFIS based rotor position estimation are discussed. The execution time of various ANFIS structure based rotor position estimation algorithms for the test motor, when the algorithms are executed on DSP
TMS320F2812 using C file is compared. The time needed to execute a gbell type input membership function is discussed.

4) It is concluded that the ANN based position estimation technique is superior than ANFIS based position estimation technique with respect to execution time.