Application of ANN and Fuzzy Systems to Some Power System Tasks
4. APPLICATION OF ANN AND FUZZY SYSTEMS TO SOME POWER SYSTEM TASKS

Introduction

On the occurrence of some fault or disturbance in the power system, it moves from normal state to abnormal state. Often this transition is reflected by the presence of some harmonics or negative sequence voltages or currents. There are conventional methods of detecting and filtering the harmonics using tank circuits tuned for selected frequencies. Also the phase sequence is conventionally detected with the use of moving disc or with two lamps and inductor connected in star to power supply. In this chapter these applications are addressed with artificial neural network and fuzzy system approaches. Firstly a simple task of phase sequence detection with feedforward ANN is presented; thereafter harmonic filtering with tapped delay ANN is attempted. Lastly the Harmonic filtering task is discussed with fuzzy approach.

4.1 Phase Sequence Detection with ANN

Phase sequence detection is necessary at the time of synchronizing an alternator to a power grid. Also as the direction of rotation of machine depends on the phase sequence of supply, prior information of this helps in proper connection of machines for rotation in intended direction, which is very important in pumps.

There are two basic conventional methods of detecting phase sequence. These are rotating and static types. In rotating method there are three coils 120° apart in space which are connected to three phase supply. These coils produce rotating magnetic field which acts upon aluminum disc and eddy voltages are induced. Because of the eddy voltages, eddy currents flow and a torque is produced. The disc revolves because of the torque and the direction
of rotation depends on the phase sequence of supply. On the other hand in static method two lamps and an inductor are connected in star and fed with three phase supply. If the inductor is designed to offer reactance equal to lamp resistance for the given supply frequency then for clockwise sequence one lamp glows dim and other bright; and for anticlockwise sequence glow of the lamps reverses [39].

To carry out the phase sequence detection with ANN, the power supply voltage has to be presented in the form of patterns. Therefore the voltages have been converted in the form of zeros and ones to form the patterns for each sequence. The three phase voltages have been sampled at equal time step and fed to feedforward neural network with self-feedback in the output layer.

### 4.1.1 Input Output Data Preparation

This section discusses the input data preparation for power supply phase sequence detection. Foremost requirement for input to ANN is that it should be in neuronal range i.e. between 0’s and 1’s or between –1’s and +1’s. For the application suggested here 0’s and 1’s have been chosen for pattern formation. Sinusoidal power supply voltage of 50 Hz can be converted into pulse of zeros and ones with the help of zero crossing detector and pulse generator. Zero crossing detectors generates triggering pulse at each zero crossing and pulse generator produces positive pulse of unit volt for the positive half of sinusoidal voltage and zero volt for negative half of the sinusoidal voltage. The waveforms thus generated for phase a, b and c can be shown as in Fig. 4.1. It is evident from the above figure that there is a relative change in a, b and c-phase voltages after 3.33 ms. Therefore the sampling time less than 3.33 ms would result in redundant information and sampling time more than 6.66 ms will lose some information. Thus it can be said three samples to six samples per cycle would result in patterns containing phase sequence features. Patterns generated above correspond to abc sequence; similarly patterns for acb
sequence can be generated. Target for abc sequence has been taken as +1 and for acb sequence as 0.

4.1.2 Training of ANN with Phase Sequence Patterns

Patterns generated in the above section have been presented to an ANN with three inputs, one for each phase and one output with delayed self-feedback: to flag the sequence as 1 for abc and 0 for acb. In the hidden layer 1-5 neurons were used convergence occurred for each case. The ANN
simulation has been done in MATLAB neural network toolbox [40]. Table-4-I below shows a set of sample input output set. While training the ANN phase sequence indicator, 12 such input sequences were presented. For mean square error of 0.000001, six samples per cycle, hyperbolic tangent transfer function in hidden layer, logarithmic transfer function in output layer and backpropagation learning algorithm (mathematical details are discussed in Chapter-5) convergence occurs in 8-9 epochs (see Fig. 4.2).

<table>
<thead>
<tr>
<th>Input No.</th>
<th>Input</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a - 1 1 1 0 0 0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>b - 0 0 1 1 1 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>c - 1 0 0 0 1 1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>a - 1 1 1 0 0 0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>b - 1 0 0 0 1 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>c - 0 0 1 1 1 0</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 4.2 Training Error Plot

4.1.3 Trained ANN Details

Following are the values of various weight matrices calculated after training the ANN based phase sequence indicator:
1. Weight matrix connecting phase-a samples to hidden layer neurons:

<table>
<thead>
<tr>
<th></th>
<th>0.8399</th>
<th>0.3009</th>
<th>-1.0247</th>
<th>-0.6523</th>
<th>-1.4521</th>
<th>-0.1309</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td>-2.9937</td>
<td>0.9468</td>
<td>0.6062</td>
<td>1.7521</td>
<td>-3.0971</td>
<td>-2.2954</td>
</tr>
<tr>
<td>3.</td>
<td>-2.0593</td>
<td>-1.0002</td>
<td>1.6417</td>
<td>1.6808</td>
<td>-1.2805</td>
<td>-2.0151</td>
</tr>
<tr>
<td>4.</td>
<td>-0.6474</td>
<td>-0.6852</td>
<td>-1.5058</td>
<td>-1.2268</td>
<td>-1.3642</td>
<td>-0.0605</td>
</tr>
<tr>
<td>5.</td>
<td>3.1190</td>
<td>-2.1867</td>
<td>-2.1194</td>
<td>-0.3655</td>
<td>1.7164</td>
<td>1.0896</td>
</tr>
</tbody>
</table>

2. Weight matrix connecting phase-b samples to hidden layer neurons:

<table>
<thead>
<tr>
<th></th>
<th>0.2465</th>
<th>1.6466</th>
<th>-0.0069</th>
<th>-2.1520</th>
<th>-3.3173</th>
<th>0.6447</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td>-1.5556</td>
<td>-0.7091</td>
<td>-4.1318</td>
<td>1.0508</td>
<td>0.3878</td>
<td>0.9353</td>
</tr>
<tr>
<td>3.</td>
<td>0.4477</td>
<td>1.3449</td>
<td>-1.0184</td>
<td>-2.6121</td>
<td>-3.6359</td>
<td>0.6881</td>
</tr>
<tr>
<td>4.</td>
<td>0.9034</td>
<td>0.8104</td>
<td>0.3425</td>
<td>-1.8601</td>
<td>-3.0942</td>
<td>-1.3082</td>
</tr>
<tr>
<td>5.</td>
<td>-0.7638</td>
<td>-1.3038</td>
<td>2.6537</td>
<td>0.4868</td>
<td>1.4760</td>
<td>-0.8652</td>
</tr>
</tbody>
</table>

3. Weight matrix connecting phase-c samples to hidden layer neurons:

<table>
<thead>
<tr>
<th></th>
<th>-3.1269</th>
<th>-1.2545</th>
<th>0.7868</th>
<th>2.9234</th>
<th>1.6900</th>
<th>0.2778</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td>0.3410</td>
<td>0.3032</td>
<td>-2.6078</td>
<td>-0.8711</td>
<td>-1.2085</td>
<td>-0.1240</td>
</tr>
<tr>
<td>3.</td>
<td>-1.3735</td>
<td>-3.6191</td>
<td>-2.0472</td>
<td>0.2880</td>
<td>1.2852</td>
<td>0.4787</td>
</tr>
<tr>
<td>4.</td>
<td>-1.5387</td>
<td>-2.4714</td>
<td>-1.1359</td>
<td>0.8599</td>
<td>1.4740</td>
<td>0.7384</td>
</tr>
<tr>
<td>5.</td>
<td>2.9605</td>
<td>2.3331</td>
<td>2.6433</td>
<td>-1.1346</td>
<td>-2.4722</td>
<td>-1.7071</td>
</tr>
</tbody>
</table>

4. Weight matrix connecting hidden layer neurons to output layer neuron:

|       | 1.2104  | 1.0675  | -4.8088 | 0.7913  | 3.8614  |

5. Weight matrix connecting output layer neuron with delayed self-feedback:

|       | -14.6978 |

6. Biases of hidden layer neurons:

|       | -3.5084 | -1.7586 | -3.5962 | -3.6494 | 2.3100  |
7. Bias of output layer neuron:
1.4461

With above matrices if the samples of power supply at any instant for abc sequence are given to the trained ANN we get the output in the range of 0.9 to 1.0 and for acb sequence we get the output in the range of 0.0001 to 0. These results indicate the correct classification for the two cases under consideration.

4.2 Harmonic Detection with ANN

4.2.1 Conventional Method

In a distorted sinusoidal wave the presence of harmonic and its magnitude is conventionally detected by LC tuned circuit shown in Fig. 4.3. If the values of L and C are such that reactance of these are equal for the frequency of interest; then on the presentation of distorted sinusoidal wave to this circuit, tuned frequency wave will appear at output and its magnitude will be proportional to the magnitude of this frequency component in input wave.

![Fig. 4.3, LC Tuned Circuit](image)

4.2.2 ANN Method

The ADALINE (Adaptive Linear Element) network shown in Fig. 4.4 with tapped delay line (see Fig. 4.5) [41] at input is proposed to simulate the harmonic detection model. If there are ‘R’ elements in the input vector of
ADALINE then ‘R − 1’ delay elements will be required. The samples of distorted sinusoidal wave containing 50, 100, 150 and 250 Hz components are input to the network.

\[ v(t) = \sin 314t + \sin 628t + \sin 942t + \sin 1570t \] (4.1)

Network trained for one frequency component as target and \( v(t) \) as input will serve as harmonic detector for that particular frequency. If the same network is trained for other frequency, it will detect this frequency component and also indicate the magnitude of this component though it has not been trained for all the combinations of magnitudes. It is important to note here that the ANN based harmonic detector is adaptive. It adapts to system changes very quickly. On the other hand in conventional harmonic detector, for every frequency component a new set of fixed L and C or variable L and C is required. To make it adaptive L and C values has to be function of system changes, which is sometimes difficult and calls for extra arrangements. However in ANN based detector, all these facilities are available in same detector except fresh training for each application.

Fig. 4.4, Adaptive Linear Element (ADALINE)
To train the network Widrow-Hoff \( \alpha \)-LMS rule has been used. This rule works on the principle of “minimal disturbance”. The weight update equation is written as:

\[
\mathbf{w}(m+1) = \mathbf{w}(m) + \alpha \mathbf{e}(m) \mathbf{a}(m) / |\mathbf{a}(m)|^2
\]  

(4.2)

Where \( m \) is adaptation cycle index, \( \mathbf{w}(m+1) \) is the next value of weight vector and \( \mathbf{a}(m) \) is the present input pattern. The present linear error (\( \mathbf{e}(m) \)) is defined as:

\[
\mathbf{e}(m) = \mathbf{d}(m) - \mathbf{y}(m)
\]  

(4.3)

Where \( \mathbf{y}(m) = \mathbf{w}(m)^T \mathbf{a}(m) \) and \( \mathbf{d}(m) \) is desired response.

\[
\mathbf{e}(m) = \mathbf{d}(m) - \mathbf{w}(m)^T \mathbf{a}(m)
\]  

(4.4)
\[ A e (m) = - a (m)^T \Delta w (m) \]  

(4.5)

From equation (4.2)

\[ \Delta w(m) = w(m + 1) - w(m) = \alpha c(m) a(m) / |a(m)|^2 \]  

(4.6)

\[ \Delta e(m) = - \alpha c(m) a(m)^T a(m) / |a(m)|^2 = - \alpha c(m) \]  

(4.7)

Hence the error reduces by a factor of \( \alpha \) and it will go on reducing in successive iterations.

### 4.2.3 Training and Testing Results for Harmonic Detection Task

Training input has been generated using the function as in equation (4.8)

\[ v(t) = \sin 314t + \sin 628t + \sin 942t + \sin 1570t \]  

(4.8)

![GENERATED TRAINING DATA GRAPH](image)

Fig. 4.6. Input Signal as in Equation (4.8)
Same training input signal has been used with different target to realize different harmonic detection. Time step in the graph is of 0.0005secs and has been kept same in all the following graph.

(i) **Second Harmonic Detection**

Target function:

\[ d(t) = \sin 628t \]  

(4.9)

Test input 1 function:

\[ v(t) = \sin 314t + 0.05\sin 628t + 0.05\sin 942t + 0.03\sin 1570t \]  

(4.10)

Test input 2 function:

\[ v(t) = \sin 314t + 0.01\sin 628t + 0.05\sin 942t + 0.03\sin 1570t \]  

(4.11)

Fig 4.7 to 4.10 show various graphs obtained for training and testing.

![Generated Test Data Graph](image)

*Fig. 4.7. Plot for Equation (4.10)*
Fig. 4.8

Fig. 4.9
(ii) Third Harmonic Detection

Target function:

\[ d(t) = \sin 942t \]  \hspace{1cm} (4.12)

Test input 1 function:

\[ v(t) = \sin 314t - 0.1 \sin 628t + 0.05 \sin 942t - 0.03 \sin 1570t \]  \hspace{1cm} (4.13)

Test input 2 function:

\[ v(t) = \sin 314t - 0.1 \sin 628t + 0.01 \sin 942t - 0.03 \sin 1570t \]  \hspace{1cm} (4.14)

Fig. 4.10 to 4.13 show the training graph and ANN output for the two test inputs in equation (4.13) and (4.14)
Fig. 4.11

Fig. 4.12
(iii) Fifth Harmonic Detection

Target function:

\[ d(t) = \sin 1570t \]  \hspace{0.5cm} (4.15)

Test input 1 function:

\[ v(t) = \sin 314t + 0.1 \sin 628t + 0.05 \sin 942t + 0.03 \sin 1570t \]  \hspace{0.5cm} (4.16)

Test input 2 function:

\[ v(t) = \sin 314t + 0.1 \sin 628t + 0.05 \sin 942t + 0.01 \sin 1570t \]  \hspace{0.5cm} (4.17)

Fig. 4.14 to 4.16 show the training and testing graphs for fifth harmonic detection.
Fig. 4.14

Fig. 4.15
4.3 Harmonic Detection with Fuzzy Approach

Detection of harmonics in power system signals with adaptive network fuzzy inference system (ANFIS) is presented in this section. This has resemblance with ANN computations. The succeeding sections include details of adaptive network, its extension to build ANFIS architecture and learning rules which are essential before applying ANFIS to harmonic filtering.

4.3.1 Adaptive Network and ANFIS Architecture

An adaptive network is a multilayer feedforward network in which a node performs a particular function on incoming signal and on parameters pertaining to this node. To have the adaptive capability in the network, both circle and square nodes are used. A square node is adaptive and has parameters: on the other hand circle node is fixed and has no parameters. Fig. 4.17 illustrates an adaptive network.
Following the basics of above network the ANFIS architecture for Takagi and Sugeno’s type if–then rules can be obtained. Let this fuzzy inference system has two inputs $x_1$ and $x_2$ and one output $y$. Suppose the rule base contains two rules,

Rule 1: If $x_1$ is $A_1^{(1)}$ and $x_2$ is $A_2^{(1)}$, then

$$y^{(1)} = p_0^{(1)} + p_1^{(1)} x_1 + p_2^{(1)} x_2$$  \hspace{1cm} (4.18)$$

Rule 2: If $x_1$ is $A_1^{(2)}$ and $x_2$ is $A_2^{(2)}$, then

$$y^{(2)} = p_0^{(2)} + p_1^{(2)} x_1 + p_2^{(2)} x_2$$  \hspace{1cm} (4.19)$$

The equivalent ANFIS architecture for above two rules, two inputs system is as depicted in Fig. 4.18. The node functions in a particular layer are of same family as described below:

Layer 1: Every node in this layer is a square node with node function $O_i^1 = m_{A_{1i}}(x_i)$, where $x_i$ is the input to node $i$ and $A_{1i}$ is linguistic label associated with this node. Alternatively it can be said that $O_i^1$ is the membership function of $A_{1i}$. Usually bell shaped membership function is used with maximum value equal to 1 and minimum value equal to 0 as below:

$$m_{A_{1i}}(x) = 1/[1 + ((x - c_i)/a_i)^2b_i]$$ \hspace{1cm} (4.20)$$
Where \( \{a_i, b_i, c_i\} \) is the parameter set. These parameter sets are referred as premise parameters.

Layer 2: Every node in this layer is a circle node labeled \( \Pi \) which multiplies the incoming signal and outputs the product, e.g.

\[
w^{(i)} = m_{A_1}(x_1) \times m_{A_2}(x_2), \quad i = 1, 2.
\]  

(4.21)

Each node output is the firing strength of a rule. In place of multiplication other operators that perform the AND operation can also be used as the node function for this layer.
Layer 3: Every node in this layer is a circle node labeled N. The \( \text{ith} \) node calculates the ratio of the \( \text{ith} \) rule weight to the sum of all the rule weights.

\[
\bar{w}_i^{(l)} = \frac{w_i^{(l)}}{w_i^{(l)} + w_j^{(l)}} , \quad i = 1,2.
\] (4.22)

Layer 4: Every node \( i \) in this layer is a square node with a node function.

\[
O_i^{(l)} = \bar{w}_i^{(l)} y_i^{(l)} = \bar{w}_i^{(l)} (p_0^{(l)} + p_1^{(l)} x_1 + p_2^{(l)} x_2) , \quad i = 1,2.
\] (4.23)

where \( \bar{w}_i^{(l)} \) is output of layer 3 and \( \{ p_0^{(l)}, p_1^{(l)}, p_2^{(l)} \} \) is consequent parameter set.

Layer 5: This layer has single node labeled \( \Sigma \). Output of this node is summation of all incoming signals i.e.

\[
O_i^{(5)} = \sum_i \bar{w}_i^{(l)} y_i^{(l)}
\] (4.24)

Thus the above constructed network is functionally equivalent to Takagi and Sugeno’s type fuzzy inference system.

4.3.2 Learning Rule

(i) Gradient Method: Consider a network with \( L \) layers and \( k^{th} \) layer has \( k \) nodes. Position of \( i^{th} \) node in \( k^{th} \) layer is given by \( (k, i) \) and the node output is denoted by \( O_i^{(k)} \). The node output is function of incoming signals and its parameter, then

\[
O_i^{(k)} = O_i^{(k)} (O_1^{(k-1)} , \ldots , O_{k-1}^{(k-1)}, a, b, c, \ldots)
\] (4.25)

Where \( a, b, c \) are parameters for this node.

Suppose the given data set has \( P \) entries. Error measure for the \( p^{th} \) (\( 1 \leq p < P \)) entry is defined as sum of squared errors
\[ E_p = \sum_{m=1}^{L} (T_{mp} - O_{mp}^L)^2 \] (4.26)

Where \( T_{mp} \) is the \( m \)th component of \( p \)th target output vector and \( O_{mp}^L \) is the \( m \)th component of actual output vector obtained by the \( p \)th input vector presentation. Hence overall error measure is \( E = \sum_{p=1}^{P} E_p \). Calculation of gradient requires error rate \( \partial E_p/\partial O \) for \( p \)th training data and for each node output. The error rate for the output node at \((L, i)\) will be,

\[ \frac{\partial E_p}{\partial O_{ip}^L} = -2(T_{ip} - O_{ip}^L) \] (4.27)

The error rate for internal node at \((k, i)\) can be calculated by using the error rate of \( k+1 \)th layer as below,

\[ \frac{\partial E_p}{\partial O_{ip}^k} = \sum_{m=1}^{k+1} \left( \frac{\partial E_p}{\partial O_{mp}^{k+1}} \right) \left( \frac{\partial O_{mp}^{k+1}}{\partial O_{ip}^k} \right) \] (4.28)

Equation (4.28) shows that the error rate of any internal node is expressed as linear combination of error rates of nodes in the next layer. Hence error rates of all the internal nodes can be calculated.

Now if \( \alpha \) is a parameter of the given adaptive network then,

\[ \frac{\partial E_p}{\partial \alpha} = \sum_{O^* \in S} \left( \frac{\partial E_p}{\partial O^*} \right) \left( \frac{\partial O^*}{\partial \alpha} \right) \] (4.29)

Where \( S \) is set of nodes whose output depend on \( \alpha \). Hence overall error measure with respect to \( \alpha \) will be,

\[ \frac{\partial E}{\partial \alpha} = \sum_{p=1}^{P} \left( \frac{\partial E_p}{\partial \alpha} \right) \] (4.30)
And the parameter update relation will be,

$$\Delta \alpha = - \eta \frac{\partial E}{\partial \alpha}$$  \hspace{1cm} (4.31)$$

Where $\eta$ is further expressed as,

$$\eta = k/\sqrt{\sum \left( \frac{\partial E}{\partial \alpha} \right)^2}$$  \hspace{1cm} (4.32)$$

Where $k$ is the step size.

(ii) Hybrid Method

The gradient method can be applied to update the parameters of adaptive network but it is very slow and suffers with the problem of local minima. Hybrid method is used to overcome these problems. This method combines the gradient method and least squares estimates (LSE) to update the parameter [27].

Let us assume a network with one output only, then,

$$\text{Output} = F(I, S)$$  \hspace{1cm} (4.33)$$

Where ‘I’ is the set of input variables and ‘S’ is set of parameters. If there exists a function $H$ such that the compound function $HOF$ is linear in the some elements of $S$. The parameter set can be decomposed in two sets.

$$S = S_1 + S_2$$  \hspace{1cm} (4.34)$$

Let $HOF$ is linear in the elements of $S_2$, then on applying $H$ to equation (4.33) we have
\[ H \text{ (output)} = H_0 F(I, S) \quad (4.35) \]

Which is linear in the elements of \( S_2 \). For given values of \( S_1 \) a matrix equation can be found for \( P \) training data

\[ AX = B \quad (4.36) \]

Where \( X \) is unknown vector whose elements are parameters in \( S_2 \). If \( |S_2| = M \) then dimension of \( A, X \) and \( B \) are \( P \times M, M \times 1 \) and \( P \times 1 \) respectively. Generally \( P \) is greater than \( M \) and there is no exact solution of (4.36). For this equation \( X^* \) a least square estimate of \( X \), can be calculated to minimize the squared error \( ||AX - B||^2 \). \( X^* \) is found using pseudo-inverse of \( X \):

\[ X^* = (A^T A)^{-1} A^T B \quad (4.37) \]

LSE of \( X \) can be found alternatively by sequential formula (4.38). Which overcomes the problem of singularity of \( A^T A \).

\[
\begin{align*}
X_{i+1} &= X_i + S_{i+1} a_{i+1} (b_{i+1}^T - a_{i+1}^T X_i) \\
S_{i+1} &= S_i - S_i a_{i-1} a_{i-1}^T S_i / (1 + a_{i-1}^T S_i a_{i-1}) \\
i &= 0, 1, \ldots, P-1
\end{align*}
\]

(4.38)

Where \( S_i \) is covariance matrix, \( a_i \) is \( i^{th} \) vector of matrix \( A \), \( b_i \) is \( i^{th} \) element of \( B \) and \( X^* \) is equal to \( X_p \). The initial values in (4.38) are \( X_0 = 0 \) and \( S_0 = \gamma I \), where \( \gamma \) is large positive number and \( I \) is \( M \times M \) identity matrix.

Gradient method and LSE is combined in the hybrid method. Each epoch of hybrid method consists of forward pass and backward pass. In the forward pass, with the presentation of input data each node output is calculated which
yields matrix A and B. Then the parameters in S2 are identified by equation (4.38). The functional signal goes forward till the error measure is calculated. In the backward pass the error rates with respect to each node output propagates from output side towards input side and parameters in S1 are updated by the gradient method as in equation (4.31).

4.3.3 ANFIS Application to Harmonic Filtering

In this section, detection of fifth harmonics often available in power lines has been attempted. A complex wave containing fundamental, second, third and fifth harmonic sinusoidal components with some delays have been used as input data and all components except one which is to be filtered out, forms the output data set. The hybrid method has been used to train the ANFIS in MATLAB fuzzy tool box. Here training of ANFIS means updating of parameters to achieve the performance goal.

Four delayed inputs lead to fairly accurate model of input output mapping. This inference system has 16 rules with four-premise variable and one consequent variable. Each input has been assigned with two-membership function of generalized bell shape. There are 55 nodes, 80 linear parameters and 24 nonlinear parameters in network. Training data pair used for training are 181, with time step of 0.0005 seconds.

Data used for fifth harmonic detection are detailed as below:

Fifth harmonic wave to be filtered: \( x(t) = \sin 1570t \)

(4.39)

Input complex wave (measured signal):

\( m(t) = \sin 314t + \sin 628t + \sin 942t + \sin 1570t \)

(4.40)

Output noise wave:
\[ n(t) = \sin(314t) + \sin(628t) + \sin(942t) \]  \hspace{1cm} (4.41)

Test input wave (Test measured signal):

\[ m(t) = \sin(314t) + \sin(628t) + \sin(1570t) + 0.001\sin(1570t) \]  \hspace{1cm} (4.42)

Test expected fifth harmonic wave:

\[ x(t) = 0.001\sin(1570t) \]  \hspace{1cm} (4.43)

With these data, results show that the fuzzy model successfully detects the presence of fifth harmonic in the measured signal even when it is slightly different than training signal.

Plot of membership function for input $-1$ after training is as shown below:

![Membership Function](Fig 4.19)

**4.3.4 Results of Harmonic Filtering with ANFIS**

Results of simulation are shown from Fig 4.20 to Fig 4.27 for training testing of ANFIS:
Fig. 4.20

Fig. 4.21

Fig. 4.22

Fig. 4.23
4.4 Conclusion

ANN offers a very attractive approach to solving some intractable problems. It can also be used to solve conventional problems to get some added advantage. It offers great breakthroughs in pattern recognition, prediction and forecasting. If the problem of electrical engineering is presented in the form of patterns then it can be used to solve variety of electrical engineering problem as attempted in this chapter, for phase sequence and harmonic detection. Results show the ability of ANN to classify the phase sequences and to detect the harmonics correctly with the use of input-output set. ANN performs well in spite of insufficient or noisy data, hence if number of inputs presented continuously is large and there is some error in sampling then also it gives the correct classification. One may not be able to mathematically model a system yet from the observed behaviour it is possible to apply ANN.

In this chapter, the use of ANFIS, to extend the harmonic detection task with fuzzy approach is also presented. Fuzzy modeling of any problem requires an expert's knowledge to generate rule base; while the ANFIS starts with some initial membership function and these functions are updated depending on input output set to meet the expected performance criterion. Hence no expert is needed; on the other hand expert's knowledge can be included for initial choice of membership function parameter and ANFIS will lead to further refined model. Also the expert's knowledge can be exploited in deciding the subdivision of input fuzzy space; otherwise by trial and error it can be achieved.

For the application presented here the ability of fuzzy systems to deal with vague data has been exploited. The fuzzy system arrives at the same result even when input is slightly different. This feature has been used in harmonic detection task. The input given to the fuzzy system contains all the frequency components and output contains all the components except the harmonic to be
detected. For fifth harmonic detection input has been varied by changing the magnitude of fifth harmonic and the fuzzy system gives the same output. Difference of the output with the input gives fifth harmonic wave. The fuzzy model used is capable of detecting the presence of even 0.1% of fifth harmonic thus leading to its measurement also. The amount of vagueness that can be added in the system to arrive at the same result needs further investigation.

The fuzzy model of this chapter, filters the fifth harmonic but the same can be trained for second, third, and other harmonics depending on the requirement, thus the training feature adds offline adaptivity which is absent in conventional methods of filtering.

Hence we conclude that in coming days the ANN and fuzzy approaches may find an important place in most of the areas of engineering and science.

**********