DESIGN OF AIRFOIL USING BACKPROPAGATION TRAINING WITH PENALTY TERM

K. Thinakaran, Dr. R. Rajasekar

1Associate Professor, Sri Venkateswara College of Engineering & Technology, Thiruvarur
2Professor, Srinivasan Engineering College, Perambalur

E-mail: thina85@rediffmail.com, Rajasekar080564@yahoo.com

ABSTRACT

Here, we investigate a method for the inverse design of airfoil sections using artificial neural networks (ANNs). The aerodynamic force coefficients corresponding to series of airfoil are stored in a database along with the airfoil coordinates. A feedforward neural network is created with input as aerodynamic coefficient and the output as the airfoil coordinates. In existing algorithm as an FNN training method has some limitation associated with local optimum and oscillation. In this paper new cost function is proposed for hidden layer separately. In the proposed algorithm the output layer is incorporated into to the cost function having linear and non linear error terms. The functional constraints penalty term at the hidden layer are incorporated into to the cost function having linear and nonlinear error terms. Results indicate that optimally trained artificial neural networks may accurately predict airfoil profile.

Keywords: Linear Error, Nonlinear Error, Steepest Descent Method, Airfoil Design, Neural Networks, Backpropagation, Penalty Term.

1. INTRODUCTION

In order to develop aircraft components and configurations, automated design procedures are needed. At present, there are two main approaches to the design of aircraft configurations. The first is direct optimization, where an aerodynamic object function, such as the pressure distribution is optimized computationally by gradually varying the design parameters, such as the surface geometry. The second is the inverse design methods. Here, a two dimensional airfoil profile is obtained for the given coefficient of lift ($C_L$) and the coefficient of drag($C_D$).

But the backpropagation takes long time to converge the computational effort can therefore be excessive. Some focused on better function and suitable learning rate and momentum (18-22). Details on the use of numerical optimization in aerodynamic design can be found in works by Hicks and other authors. (1-5). In inverse design methods, the aim is to generate geometry for airfoil. There are many inverse techniques in use, for example, hodo-graph methods for two dimensional flows (4-6) and other two dimensional formulations using panel methods (7-8). The above methodologies have been extended to the three dimensional case (12-14). To design fast algorithm, Abid et al. proposed a new algorithm by minimizing sum of squares of linear and nonlinear errors for all output (22). Kathirvalavakumar proposed new efficient learning algorithm for training ANN (21). The hidden layer and output layer was trained separately to speed up the convergence. Many constrained learning algorithm with functional constraints into neural networks have been proposed (24). Jeong et al. proposed learning algorithm based on first and second order derivatives of neural activation at hidden layers (23).
Han et al. proposed two modified constrained to obtain faster convergence (25). The additional cost terms of the first algorithm are selected based on the first order derivatives of the activation functions of the hidden neurons and second order derivatives of the activation of the output neurons. Second one are selected based on the second order derivatives of the activation functions of the hidden neurons and first order derivatives of activation functions of the output neurons.

The objective of this work is to show that artificial neural networks can be trained to design airfoil profiles. We have proposed new cost function by incorporating functional constraints proposed by Han et al and the cost function proposed by Abid et al(25). In the proposed algorithm the output layer is incorporated into to the cost function having linear and non linear error terms. The functional constraints penalty term at the hidden layer are incorporated into to the cost function having linear and nonlinear error terms. It converges faster than the non linear method. ANNs are able to perform even in the presence of noise in the data.

2.ANN TRAINING METHOD

\[ n_i := g(\sum_j w_{ij}n_j - \mu_i) \]

\[ \sum \]

\[ \mu_i \]

\[ n_i \]

Figure 1. Schematic Diagram for a Simple Neuron

Here, \( n_i \) is called the state or activation of the neuron \( i \). \( g() \) is a general non linear function called variously the activation-function. The weight \( w_{ij} \) represents the strength of the connection between neurons \( i \) and \( j \). \( \mu_i \) is the threshold value for neuron \( i \), the general architecture of a two layer neural network with feed-forward connections and one hidden layer is shown in Figure 3. The input layer is not included in the layer count because its nodes do not correspond to neural elements. The weighted sum of the inputs must reach the threshold value for the neuron to transmit. One drawback associated with neural networks is that it is normally very difficult to interpret the values of the connecting weights \( w_{ij} \) in terms of the task being implemented.

Neural networks offer a very powerful and general framework for representing nonlinear mappings from several input variables to several output variables. Since the goal is to produce a system which makes good predictions for new data. Training generally involves minimization of an appropriate error function defined with respect to the training set. Learning algorithms such as the back-propagation algorithm for feed-forward multilayer networks (16) help us to find such a set of weights by successive improvement from an arbitrary starting point.

An airfoil profile can be described by a set of \( x \)- and \( y \)-coordinates, as illustrated in figure 2.

![Figure 2. Flow Field And Airfoil Data](image)

ANN Architecture Used

![Figure 3. The Neural-Network Model Trained To Predict Surface Y-Coordinates And Cl, Cd And X Coordinates Are The Inputs](image)
The aerodynamic force coefficients corresponding to series of airfoil are stored in a database along with the airfoil coordinates. A feedforward neural network is created with input as a aerodynamic coefficient and the output as the airfoil coordinates. This is then trained to predict the corresponding surface y-coordinates.

We used the below sigmoidal activation function to generate the output.

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

(1)

For a neuron j at output layer L, the linear outputs given by Abid et al. is

\[ u^L_j = \sum w_{ji} y^H_i \]  

(2)

Where \( w_{ji} \) is the weight connection between the output neuron j and hidden neuron i. And \( y^H_i \) is the output of neuron i at hidden layer H.

And the non linear output given by Abid et al. is

\[ f(u^L_j) = \frac{1}{1 + e^{-u^L_j}} \]  

(3)

The non linear error is given by

\[ e^L_j = d^L_j - y^L_j \]  

(4)

Where \( d^L_j \) and \( y^L_j \) respectively is desired and current output for \( j^{\text{th}} \) unit in the \( L^{\text{th}} \) layer.

The linear error is given by

\[ e^L_j = ld^L_j - u^L_j \]  

(5)

Where \( ld^L_j \) is defined as below

\[ ld^L_j = f^{-1}(d^L_j) \]  

(Inversing the o/p layer)

The learning function depends on the instantaneous value of the total error. As per the modified backpropagation by Abid et al. the new cost term to increase the convergence speed is defined as

\[ E_p = \frac{1}{2} (e^L_j)^2 + \frac{\lambda}{2} (e^L_j)^2 \]  

(6)

Where \( \lambda \) is Weighting coefficient

Now, this new cost term can be interpreted as below

\[ E_p = (\text{nonlinear error} + \text{Linear error}) \]

To achieve low input and output mapping the error must be reduce by derivative of cost function When the value of \( u^L_j \) becomes larger. So the proposed algorithm has penalty term defined by Jeong and Lee[24] at the hidden layer in addition with cost function having linear and non linear term, which is defined as follows

\[ \text{penalty term} = \mu_{\cdot H} \gamma_{\cdot H} \int_{n_{H}} f^{-\lambda} (\mu_{\cdot j} H) \]

We have proposed new cost function for MBFC. This procedure, the dependence of the learning function is on the instantaneous value of the total error thereby leading to faster convergence.

MBFC

The new cost term to increase the convergence speed is defined as

\[ E_p = \frac{1}{2} (e^L_j)^2 + \frac{\lambda}{2} (e^L_j)^2 - \text{penalty} \]

\[ E_p = \text{(nonlinear error+ Linear error - penalty)} \]

Learning of the Output Layer
Now, the weight update rule is derived by applying the gradient descent method to Eq. (8).

Hence we get weight update rule for output Layer L as
\[ \Delta w_{ji} = -\mu_L \frac{\partial E_p}{\partial w_{ji}} \]

Where \( \mu_L \) is the network learning parameter

\[ \Delta w_{ji} = -\mu_L e^L_i \frac{\partial y^L_i}{\partial w_{ji}} + \mu_L e^L_i \frac{\partial u^L_i}{\partial w_{ji}} \]

\[ \Delta w_{ji} = -\mu_L e^L_i \frac{\partial y^H_i}{\partial w_{ji}} + \mu_L e^L_i \frac{\partial u^L_i}{\partial w_{ji}} + \mu_L e^L_i y^H_i \]

\[ \Delta w_{ji} = -\mu_L e^L_i f'(u^L_i) y^H_i + \mu_L e^L_i y^H_i \]

**Learning of the hidden layer**

And we get the weight update rule for the hidden layer \( H \) as

\[ \Delta w_{ji} = -\mu_H \frac{\partial E_p}{\partial w_{ji}} \text{ - Penalty} \]

\[ \Delta w_{ji} = -\mu_H e^H_i \frac{\partial y^H_i}{\partial w_{ji}} + \mu_H e^H_i \frac{\partial w^H_{ji}}{\partial w_{ji}} \text{ - Penalty} \]

\[ \Delta w_{ji} = -\mu_H e^H_i \frac{\partial y^H_i}{\partial a^H_{ji}} + \mu_H e^H_i y^H_i \]

\[ \Delta w_{ji} = -\mu_H e^H_i f'(u^H_i) y^H_i + \mu_L e^H_i y^H_i \]

The network learning parameter \( \mu \) is initialized, which plays an important role in minimizing the error. Then the network is trained with corresponding change of weight for both hidden and output layer. And the weight is determined by calculating linear, nonlinear error. Which play an important role in minimizing the error. Then the network is trained with new cost function.

**Proposed Algorithm**

In the Proposed Algorithm the network learning parameter \( \mu \) is first initialized. Here, the change of weight for output layer and hidden layer is determined using new cost function equation (7) and (8) respectively.

- **Step 1**: Initialize the parameter \( \mu \) to some random values
- **Step 2**: Assign Threshold value to a fixed value based on the sigmoid function.
- **Step 3**: Calculate linear output using equation (2)
- **Step 4**: Calculate Non-Linear output using sigmoid function as in the equation (3)
- **Step 5**: Calculate the below values for output layer
- **Step 6**: Calculate the below values for hidden layer
- **Step 7**: Calculate the weight change for hidden layer using the equation (8)
- **Step 8**: If the mean square error value is greater than threshold value, then the above steps from 3 to 7 is repeated
- **Step 9**: If the mean square error value is less than threshold value, then declare that the network is trained

**3. RESULTS AND DISCUSSION**

In our investigation of neural network models for inverse design, we found that satisfactory results were obtained by using a feedforward neural network with a penalty term in the hidden layer learning. In our case, it was found that twenty hidden nodes could adequately capture the nonlinear relationship between the airfoil profiles.

As mentioned previously we have a database comprised of 26 upper and lower-surface x and y coordinates, together with the corresponding coefficient of lift\((C_L)\) and the coefficient of drag\((C_D)\). There were 1422 patterns in total. The main goal is to determine the airfoil profile for a given conditions. This is the “inverse” problem.
The network was trained to minimum error (using 1000 training patterns) on a test set (comprising 422 patterns) which was not used in the training process. The computed profiles show good agreement with the actual profiles. The new airfoil is tested again for the same flow conditions in Xfoil tool to compare $C_l$, $C_d$.

In Table I, we have given the values of stored $y$ coordinates, the values of calculated $y$ coordinates for a pattern and also the difference between these values. Just for sample, we have given 7 coordinates out of 26 coordinates for a pattern. From this table, we can say that the computed profiles generated during the test process show good agreement with the actual profiles.

**Table I - Profile Comparison Between Calculated & Stored Y Coordinates**

<table>
<thead>
<tr>
<th>Y coordinate in database</th>
<th>Y coordinate calculated using proposed algorithm in test phase</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0027</td>
<td>0.0030</td>
<td>-0.0003</td>
</tr>
<tr>
<td>0.0081</td>
<td>0.0080</td>
<td>0.0001</td>
</tr>
<tr>
<td>0.0154</td>
<td>0.0156</td>
<td>-0.0002</td>
</tr>
<tr>
<td>0.0256</td>
<td>0.0249</td>
<td>0.0007</td>
</tr>
<tr>
<td>0.0337</td>
<td>0.0350</td>
<td>-0.0013</td>
</tr>
<tr>
<td>0.0420</td>
<td>0.0447</td>
<td>-0.0027</td>
</tr>
<tr>
<td>0.0525</td>
<td>0.0529</td>
<td>-0.0004</td>
</tr>
</tbody>
</table>

This figure 4 is prepared using the profile stored in database and using the profile generated from the proposed algorithm during test phase. The red color airfoil is drawn using the profile namely naca0012 stored in database. The green color airfoil is drawn using profile naca0012 generated by proposed algorithm during test phase. From this figure, we can say that the computed profiles generated during the test process show good agreement with the stored database profiles.

From figure 4, we can say that the computed profiles generated during the test process show good agreement with the stored database profiles. Next we compute the convergence rate at training phase. To do this, we noted down the MSE error at each epoch and plotted it in the graph in Figure 5. The blue line indicates the error. It is clear that our approach converges quickly and in this approach the error is less at the converging stage. It shows how the training decreases mean square Error (MSE) with the epoch. From this figure it is obvious that using penalty term increase the converges speed and without oscillation of learning.

![Convergence History](image)

**Figure 5. Convergence Of Trained Neural Network.**

Next the error in each epoch obtained from non linear and non linear with penalty approach are compared and tabulated in Table-III. From this table, it is clear that this method predicted...
approximately correct airfoil without oscillation in learning. This proves that the proposed approach results in less error and takes less time to predict the airfoil for the given CL & CD.

Table II - Comparison Table

<table>
<thead>
<tr>
<th>Epoch</th>
<th>Training the Neural Network with penalty Term</th>
<th>Without penalty term</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2.58363</td>
<td>3.97564</td>
</tr>
<tr>
<td>1200</td>
<td>0.00163</td>
<td>0.00159</td>
</tr>
<tr>
<td>2400</td>
<td>0.00147</td>
<td>0.00147</td>
</tr>
<tr>
<td>3600</td>
<td>0.00133</td>
<td>0.00135</td>
</tr>
<tr>
<td>4800</td>
<td>0.00112</td>
<td>0.00121</td>
</tr>
<tr>
<td>6000</td>
<td>0.00086</td>
<td>0.00098</td>
</tr>
<tr>
<td>7200</td>
<td>0.00060</td>
<td>0.00063</td>
</tr>
<tr>
<td>8400</td>
<td>0.00038</td>
<td>0.00035</td>
</tr>
<tr>
<td>9600</td>
<td>0.00024</td>
<td>0.00020</td>
</tr>
<tr>
<td>10800</td>
<td>0.00017</td>
<td>0.00013</td>
</tr>
<tr>
<td>12000</td>
<td>0.00016</td>
<td>0.00011</td>
</tr>
<tr>
<td>13200</td>
<td>0.00016</td>
<td>0.00010</td>
</tr>
<tr>
<td>14400</td>
<td>0.00015</td>
<td>0.00009</td>
</tr>
<tr>
<td>15600</td>
<td>0.00014</td>
<td>0.00009</td>
</tr>
<tr>
<td>16800</td>
<td>0.00011</td>
<td>0.00008</td>
</tr>
<tr>
<td>18000</td>
<td>0.00009</td>
<td>0.00008</td>
</tr>
<tr>
<td>19200</td>
<td>0.00007</td>
<td>0.00008</td>
</tr>
<tr>
<td>20400</td>
<td>0.00005</td>
<td>0.00008</td>
</tr>
<tr>
<td>21600</td>
<td></td>
<td>0.00009</td>
</tr>
<tr>
<td>22800</td>
<td></td>
<td>0.00008</td>
</tr>
<tr>
<td>24000</td>
<td></td>
<td>0.00008</td>
</tr>
<tr>
<td>25200</td>
<td></td>
<td>0.00007</td>
</tr>
<tr>
<td>26400</td>
<td></td>
<td>0.00006</td>
</tr>
</tbody>
</table>

Figure 6 contains the airfoils naca0012, naca0014 and naca1017 which are generated by proposed algorithm in test phase. From this figure, we can say that profile generated from proposed algorithm in test phase matches with that of stored database profiles. Thus we observe that the adaptive learning rate approach generated comparatively the correct airfoil profiles.

A measure of the accuracy of the results obtained can be inferred from examination of error which is defined as:

$$\text{Error} = \frac{y_i(\text{actual}) - y_i(\text{computed})}{\text{airfoil thickness ratio}} \times 100$$

Where $y_i(\text{actual})$ is the actual y-coordinate of the section at location i, $y_i(\text{computed})$ is the computed y-coordinate. Table-III shows the maximum error in percentage for the airfoil profiles naca0012, naca0014 and naca1017 which are generated by proposed
algorithm in test phase. From this table, we can conclude that the our approach predicated comparatively the correct airfoil profiles.

Table III - Maximum Error For Airfoil Profiles Generated By Proposed Algorithm.

<table>
<thead>
<tr>
<th>Airfoil</th>
<th>Maximum error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NACA0012</td>
<td>0.001799</td>
</tr>
<tr>
<td>NACA0014</td>
<td>0.001493</td>
</tr>
<tr>
<td>NACA0017</td>
<td>0.003778</td>
</tr>
</tbody>
</table>

4. CONCLUSIONS

In this paper, we have used an inverse design methodology using artificial neural networks which is used for the design of airfoil profiles. The results indicate new cost function for backpropagation increase the convergence speed. In the proposed algorithm the functional constraints penalty term at the hidden layer are incorporated into the cost function having linear and non linear error terms. The output layer incorporated into cost function having linear and nonlinear error terms. Results indicate that optimally trained artificial neural networks may accurately predict airfoil profile without oscillation in learning.

REFERENCES


