Scientia Iranica B (2017) 24(5), 2490–2500

SCIENTIA IRANICA

Sharif University of Technology
Scientia Iranica
Transactions B: Mechanical Engineering
www.scientiairanica.com

Research Note

An optimum neural network for evolutionary aerodynamic shape design

N. Timnak[a], A. Jahangirian[a,*] and S.A. Seyyedsalehi[b]

a. *Department of Aerospace Engineering, Amirkabir University of Technology, Tehran, Iran.*
b. *Department of Bio-Engineering, Amirkabir University of Technology, Tehran, Iran.*

Received 24 October 2015; received in revised form 10 January 2016; accepted 26 September 2016

KEYWORDS
Airfoil design;
Genetic algorithm;
Neural network;
Pruning approach;
Navier-Stokes solver.

Abstract. Two new techniques are proposed to enhance the estimation abilities of the conventional Neural Network (NN) method in its application to the fitness function estimation of aerodynamic shape optimization with the Genetic Algorithm (GA). The first technique is pre-processing the training data in order to increase the training accuracy of the Multi-Layer Perceptron (MLP) approach. The second technique is a new structure for the network to improve its quality through a modified growing and pruning method. Using the proposed techniques, one can obtain the best estimations from the NN with less computational time. The new methods are applied for optimum design of a transonic airfoil and the results are compared with those obtained from the accurate Computational Fluid Dynamics (CFD) fitness evaluator and with the conventional MLP NN approach. The numerical experiments show that using the new method can reduce the computational time significantly while achieving improved accuracy.

1. Introduction

Among different methods for aerodynamic shape optimization, the Genetic Algorithm (GA) is a popular method that has been widely used by researchers [1,2]. The specifications of GA cause its superiority to other optimization methods. The most important characteristic of GA is that it does not need computing the sensitivity of the derivatives, because calculating the gradient information especially for non-linear functions is very complicated. GA also works well when the design parameters are increased. Another important feature of GA is searching the design space in a population of points, not one special point, which results in a greater likelihood of finding the global optimum point [3]. In addition, GA is an attractive method for finding the optimum value in the non-smooth spaces.

The other important key of these algorithms is using a combination of *exploitation* of the positive characteristics of the existing set of solutions and *exploration* of other areas of the design space to find the optimum solution [3,4]. The applications of this method to the airfoil shape optimization are presented in [1,2,5-7]. However, the unfavorable key about GA is the computational time consumed in aerodynamic shape optimization problems when Computational Fluid Dynamic (CFD) methods are used for fitness function calculation. For example, the computational time required for optimization with GA where a viscous flow solver is considered for the fitness function calculation would be about several days on a traditional instrument (PC) for 70-80 generations including 20 members. Therefore, several approaches, such as adaptive range GA [8] or parallel processing [9], are presented in order

*. *Corresponding author. Tel.: +98 21 64543223;*
E-mail address: ajahan@aut.ac.ir (A. Jahangirian)

doi: 10.24200/sci.2017.4308

to decrease the required computational time. These methods try to decrease the overall time by modifying the GA or with the help of advanced hardware.

Another method to decrease the GA optimization time is to reduce the number of expensive fitness evaluations by CFD solver. One of these techniques is known as Neural Network (NN) that is suitable for estimation of the objective functions in the optimization problems. Actually, these methods are capable of approximating the fitness function by the help of available information from the sequential generations during the optimization.

Different research studies have been carried out using NN in aerodynamic optimizations [10-12]. However, before full application of these methods in the field of optimization, there are two important problems that need to be resolved. First, the model should be strong enough to map the inputs to the outputs in an accurate manner. Second, the number of training data should be as low as possible. Due to these two constraints, even some new NNs are not appropriate for airfoil shape optimization. For example, deep neural networks cannot be trained conveniently. Thus, these networks have problems in the training process and fall in the local minimum [13]. The other new neural networks are convolution ones. These networks have good estimation ability and a good generalization. However, due to their large structure, these networks require many training data, which is in contrast with the second constraint [14]. Some works have been carried out in order to increase the efficiency of the neural networks for airfoil shape optimization, such as Inexact Pre-Evaluation (IPE) by Karakasis and Giannakoglou [15] or applying normal function distribution by Shahrokhi and Jahangirian [16].

In the present work, a pre-processing method is initially proposed that normalizes the MLP input data in a special manner. This helps the MLP training process to become smoother while increasing the training accuracy. Then, in order to upgrade the efficiency of the NN, a new structure for the network is presented so that the required learning data will be reduced while higher accuracy will be achieved. The methods will be explained in detail and the results are presented for a transonic airfoil design problem.

2. Aerodynamic optimization with GA

In the present study, a simple GA is applied for optimum design of a transonic airfoil. Actually, GA is a collection of generations that contain several chromosomes. In airfoil shape optimization, each chromosome indicates an airfoil and the genes of each chromosome are the airfoil parameters that are obtained from PARSEC parameterization, which contains the leading edge radius (r_{LE}), upper and lower crest locations (X_{UP}, Y_{UP}, X_{LO} , Y_{LO}) and curvatures $Y_{xx\mathrm{UP}}$, $Y_{xx\mathrm{LO}}$) trailing edge coordinate Y_{TE}, direction α_{TE}), and trailing edge wedge angle β_{TE}). According to PARSEC, by solving a system of linear equations as follows one can obtain the shapes of the airfoil:

$$Y_k = \sum_{n=1}^{6} a_{n,k} X_k^{\frac{n-1}{2}}, \qquad k = 1, 2, \tag{1}$$

where $k = 1$ is considered for the upper surface and $k = 2$ is related to the lower surface of the airfoil. The coefficient 'a_n' is related to the defined geometric parameters. The PARSEC parameters are bounded in order to avoid impractical shapes. The boundaries are introduced in Section 6 for airfoil shape optimization. More information about PARSEC parameterization can be found in reference [6].

Thus, the genetic algorithm starts with a collection of chromosomes and generates new chromosomes from previously generated members using GA operators. In this work, selection of the chromosomes for the next generation is done by the tournament operator [3] with an elitist strategy, where the first and the second best chromosomes in each generation are directly transferred into the next generation. Also, one-point crossover operator with an 80% probability of combination is used [17] and the mutation probability is set 10%. The aerodynamic efficiency factor (C_L/C_D) is determined as the objective function. A restriction is applied to limit the airfoil maximum thickness not to be less than a prescribed value (i.e. 11%) in order to avoid very thin shapes. The total population of each generation is set to be 20.

3. Numerical flow solver

The airfoil shapes (chromosomes) that are generated by the GA should be evaluated by a proper fitness function. As mentioned in Section 2, the objective function assumed in this paper is (C_L/C_D), which is supplied from a CFD flow solver. This solver is based on the numerical simulation of turbulent viscous flow governed by Reynolds averaged Navier-Stokes equations. Triangular unstructured grids are utilized for discretization of the computational field. Since most of the computational time in such an optimization problem is consumed by the CFD solver, it must have a high efficiency and convergence rate. In order to achieve this goal, a dual-time implicit method proposed by Jahangirian and Hadidolabi [18] is used for unstructured grids.

Due to the high necessity for running the CFD, generating high-quality grids is of outmost significance. Therefore, a successive refinement method presented by Jahangirian and Johnston [19] is used for unstructured grid generation. This method produces high-quality stretched cells inside the boundary and shear layers as well as isotropic cells outside these regions.

4. Fitness approximation with a neural network

As previously mentioned, NN is an alternative approach to CFD in order to reduce the computational time required for the fitness function evaluation. However, the accuracy of the available NNs is highly dependent on the number of training data, which are in turn supplied from accurate CFD calculations and, hence, require extra computational effort. Thus, more sophisticated NN methods are required for full application of evolutionary algorithms in aerodynamic shape optimization, which requires lower training data for the NN, having the higher estimation ability. Two new approaches are presented in this work to enhance the estimation ability of the conventional Neural Network (NN) method.

4.1. Pre-processing the training data

In the case of the present work, the inputs of the NN are the PARSEC parameters and, as mentioned before, we expect the network to predict the value of (C_L/C_D) for each airfoil. According to PARSEC parameterization, input layer of the network contains 10 components that can change within their own boundary values. Actually, the boundary values of the input parameters have widespread orders. For example, the domain of changes for the first PARSEC parameter (r_{LE}) is [0.006-0.0115] where the domain of the fifth parameter (α_{TE}) is [0-10]. This causes the effects of the higher-order parameters to be much more than those of the lower-order parameters during the training process. This means that when an NN faces parameters with different orders, while mapping these inputs to their own fitness function, it will give more weight to the parameters with higher orders; thus, it is not a smooth training. During the training process of the NN, the differences in weight matrices of the first hidden layer are calculated by the delta rule with the following relation:

$$\Delta V = \eta X^T \times \overline{\delta_y}, \tag{2}$$

where, $X^T$is the transpose of the input matrix, $\overline{\delta_y}$ is the back propagated error signal matrix of the hidden layer, and η is the learning step (learning rate). Thus, the edition weight matrix has a direct relationship with the inputs $(\Delta V \sim X)$. Therefore, the network will give more weight to the bigger inputs. For instance, the relative importance of the PARSEC parameters in the NN for the mentioned database is as follows:

$$\beta_{TE}, \alpha_{TE} >>> X_{UP}, X_{LO} > Y_{xxUP}, Y_{xxLO}$$
$$>> r_{LE}, Y_{UP}, Y_{LO}, Y_{TE}.$$

However, it is obvious that all of these parameters have an important role in determining the (C_L/C_D) factor for an airfoil. This serious challenge causes two problems in the results: first, training does not correctly occur; second, estimating the results of the test data has large error and is not acceptable. Therefore, it is very important to modify the inputs by a pre-processing operation in order for each parameter to be seen equivalently. To achieve this aim, the following formula is used to modify each input "X_i" as:

$$\frac{X_i - \overline{X}}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(X_i - \overline{X}\right)^2}}, \tag{3}$$

where $\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$ and it is important to note that the average of the new data is 'zero', i.e. $\overline{\left(X_i - \overline{X}\right)} = 0$. Therefore, by this modification, the entire training data of the NN will be of the same order and will be seen equivalently.

4.2. The new neural network structure

The most common structure for neural network is the feed forward Multi-Layer Perceptron (MLP) and the most popular method for training the network is the back propagation method (BP) [20]. In general, the structure of a network contains the input, output, and hidden layers. Dimensions of the input space and dimensions of the fitness function space determine the number of neurons in the input and the output layers. But, what about the number of neurons in the hidden layers? In the MLP structures, the number of hidden neurons is usually set to be constant at first; but, it continuously changes to reach a better output. In fact, it is a trial and error approach that does not guarantee the best efficiency for the MLP networks. Minsky and Papert [21] showed that MLP networks were not efficient enough to cover all problems.

In summary, small networks with inadequate hidden neurons cannot work efficiently as it is expected. On the other hand, large networks with too many neurons can cause over-fitting, which can decrease the estimation ability [22]. Thus, the efficiency of the network in both situations is weak. Recently, researchers have investigated two different ways for choosing the network topology:

1. Pruning algorithms which start with a large network and optimize it;
2. Constructive algorithms which start with a small network and try to optimize it by adding neurons to the hidden layers [23].

One can see the examples of these methods, which are applied in biomedical engineering problems, in [24].

In this paper, a new structure related to "growing and pruning method" is presented. In the present work, the pruning stage method is extended in order to have more accurate estimations in less computational time. This new idea is based on the *independence of the hidden neurons*, which significantly helps the network to increase the accuracy and estimation ability.

4.2.1. Growing approach

In the first stage, the network starts with only one hidden neuron and, then, develops the network structure by neurons growing based on the network errors during the training epochs and splitting the algorithm based on maximum error of the hidden neurons. This means that in each step, the neuron with the maximum error is found and, then, it is split in two neurons in order to share the task of the previous neuron between the two new ones. The growing algorithm starts with one neuron in the hidden layer and, then, during the NN training, if the number of hidden layers cannot reduce the network error, the neuron in the hidden layer with the maximum back propagated error signal over all training data is found by the following calculation:

$$\left|\delta_{y_\xi}\right| = \max\left\{\sum_{i=1}^{N_p}\left|\delta_{y1}\right|_i, \sum_{i=1}^{N_p}\left|\delta_{y2}\right|_i, \ldots, \sum_{i=1}^{N_p}\left|\delta_{yN2}\right|_i\right\}, \quad (4)$$

where δ_y is the signal error vector of the hidden layer and is calculated as follows:

$$\delta_y = \left[\delta_{y_1} \quad \delta_{y_2} \quad \cdots \quad \delta_{y_{N2}}\right]^T. \quad (5)$$

Now, according to the fact that the neuron in "ξ" position has a high error, we spilt it in two neurons in order to share the tasks of the "ξ " neuron between two new neurons, which are labeled by "ξ_1" and"ξ_2". It is obvious that in this case, two neurons are more capable of doing the tasks of one neuron.

As it is obvious in Figure 1, the second step is finding the input/output weight vectors for the two new neurons. Note that in Figure 1 and Eq. (6), ε is a small number.

As the weight vector for each neuron determines a hyper plane in the input space, the input weight vector must be as follows:

$$\begin{cases} V(\xi_1) = V(\xi) \\ V(\xi_2) = V(\xi) + \varepsilon \end{cases} \quad (6)$$

For determining the output layer weights, one should notice that the desired outputs must not be changed. Thus, the new neurons must also generate the previous output. Therefore, the output weights are discrete between two new neurons, i.e.:

$$\begin{cases} W(\xi_1) = \frac{1}{2}W(\xi) \\ W(\xi_2) = \frac{1}{2}W(\xi) \end{cases} \quad (7)$$

4.2.2. The new pruning approach

Generally, the pruning method summarizes the hidden neurons with different ideas; but, in the present work, we carry out the second stage (summarizing the hidden layers) based on the linear dependency of the neurons of the hidden layers. To have a brief explanation, it should be noted that in comparison to the conventional pruning approach, for each hidden layer, we add a virtual layer too and, then, we train each virtual layer in a special manner in order to find the neurons that have a linear correlation with the other ones. This structure is shown in Figure 2. By the help of this idea, the input weights are edited in a way that the hidden neurons have the least linear correlation and if a neuron has a high correlation with the other ones, it will be omitted. Pruning in the proposed way causes the decision region to have higher generalization ability and have better estimation for the output.

As mentioned before, this layer does not belong to the neural network but is a virtual layer. We call this layer the "*linear dependent detector*" because the task of this layer is to determine how much a neuron in the hidden layer is dependent on the other neurons. Thus, this connection determines the dependence/independence of hidden neurons. It is important to notice that if a neuron is linearly dependent, the other neurons can do its task. Thus, the existence of this dependent neuron only needs additional computations and causes the network to be large, hence, decreasing the efficiency of the network. Subsequently, the dependent neuron is omitted from

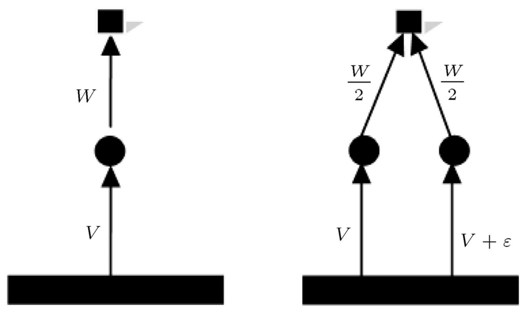


Figure 1. The process of splitting the neurons.

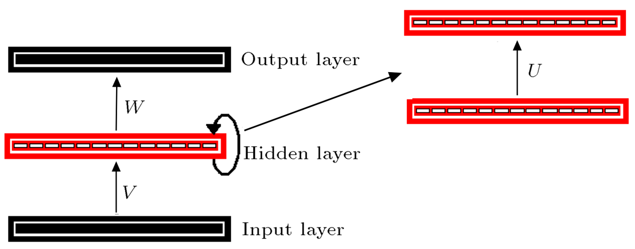


Figure 2. The structure of extended pruning process.

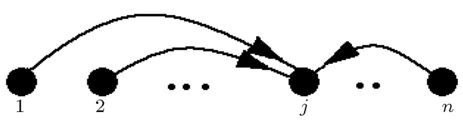


Figure 3. Neurons connection in the recurrent layer.

the list of the hidden neurons and its task is divided between the other neurons in the hidden layer. As mentioned before, the training of this virtual layer is carried out in a special manner that is explained in the following:

In Figure 3, it is obvious that each neuron has only one connection with the others and not to itself. Thus, in the weight matrix, the diagonal components are zero.

$$U=\begin{bmatrix} 0 & u_{12} & u_{13} & \cdots & u_{1,N2} \\ u_{21} & 0 & u_{23} & \cdots & u_{2,N2} \\ u_{31} & u_{32} & 0 & \cdots & u_{3,N2} \\ \vdots & & & \ddots & \vdots \\ u_{N2,1} & u_{N2,2} & u_{N2,3} & \cdots & 0 \end{bmatrix}_{N2\times N2}$$

Now, for each neuron in the“*linear dependent detector*”, the error is calculated by:

$$e_{jw}=y_j-\sum_{\substack{i=1\\ i\neq j}}^{N2} y_i u_{ij} \qquad j=1,2,\cdots,N2. \tag{8}$$

Then, among the hidden neurons, a neuron with the minimum {e} is found because:

$$\text{if } \ e_{\mathrm{jw}}\to 0; \ \text{ then } y_j\to\sum_{\substack{i=1\\ i\neq j}}^{N2} y_i u_{ij},$$

this means that y_j is linearly dependent.

Then, y_i is a linear combination of other neurons. In other words, it is dependent. Thus, this neuron is a candidate for pruning:

$$|e_{\xi w}|=\min_{j=1,2,\cdots,N2}\{|e_{jw}|\}. \tag{9}$$

Eq. (8) means that y_ξ goes to the elimination list. Therefore, this neuron and all its connections are eliminated from the network. After eliminating a neuron, the weighted matrix is updated by the new dimension of networks.

According to the structure of the network in the second stage, updating the weights requires three steps, which are demonstrated in the following Eq. (10)-(17). The purpose of the following relations is to mathematically show the effects of the recurrent layer. It is obvious that removing the dependent neurons develops the network in terms of generalization. Note that, in the following equations, α is the momentum coefficient and the others have been defined before.

1. Weight edition of the output layer:

$$\Delta w_{jk}=\eta y_j\delta_{z_k}+\alpha\Delta w_{jk} \quad k=1,2,\cdots,N3, \tag{10}$$

$$w_{jk}=w_{jk}+\Delta w_{jk} \qquad j=1,2,\cdots,N2. \tag{11}$$

2. Weight edition of the hidden layer:

$$\delta_S=\varepsilon\times(y-yW), \tag{12}$$

$$\Delta u_{jt}=\eta\, y_j\delta_{S_t}+\alpha\Delta u_{jt} \quad j=1,2,\cdots,N2, \tag{13}$$

$$u_{jt}=u_{jt}+\Delta u_{jt} \quad t=1,2,\cdots,N2. \tag{14}$$

3. Weight edition of the input layer:

$$\delta_y=\ \delta_y-\ \delta_S, \tag{15}$$

$$\Delta v_{ij}=\eta\, x_i\delta_{y_j}+\alpha\,\Delta v_{ij} \qquad j=1,2,\cdots,N2, \tag{16}$$

$$v_{ij}=v_{ij}+\Delta v_{ij} \qquad j=1,2,\cdots,N1. \tag{17}$$

Finally, it is crucial to state that this virtual connection decreases the components that explain the model (map each airfoil to its aerodynamic efficiency factor (C_L/C_D)). This means that by omitting the dependence hidden neuron, the network becomes brief enough and the additional tasks are omitted. Therefore, the output of the model is the best. This is the “*optimum*” concept that is put forth in the title of the paper. Despite all these reasons, by omitting the additional tasks, the computational time of the training process is also decreased.

5. Validation

In this section, the capability and efficiency of the proposed methods are investigated in comparison to the basic neural network (MLP) through the exact solution of a well-known explicit function. The selected function here is the “3-D Sinc” function as used in [25] with the formulation:

$$z(x_1,x_2)=\begin{cases}\frac{\sin(\pi x_1)\sin(\pi x_2)}{\pi^2 x_1x_2} & x_1,x_2\neq 0\\ 1 & x_1,x_2=0\end{cases} \tag{18}$$

It is noted that in the 3-D Sinc function if $x_1=0$ and $x_2\neq 0$, then $z=\frac{\sin(\pi x_2)}{\pi x_2}$ and if $x_2=0$ and $x_1\neq 0$, then $z=\frac{\sin(\pi x_1)}{\pi x_2}$. Now, we apply the proposed methods for the NN and the basic NN on this exact function. Figure 4 shows the average error of the training process for each method. The average error is defined as:

$$\text{Average error}=\frac{\sum\limits_{i=1}^{n}\left|Y_{\mathrm{output}_i}-Y_{\mathrm{exact}_i}\right|}{n}. \tag{19}$$

Figure 5 shows the results of applying both networks to this function. It is obvious in the diagrams that the proposed method has estimated the

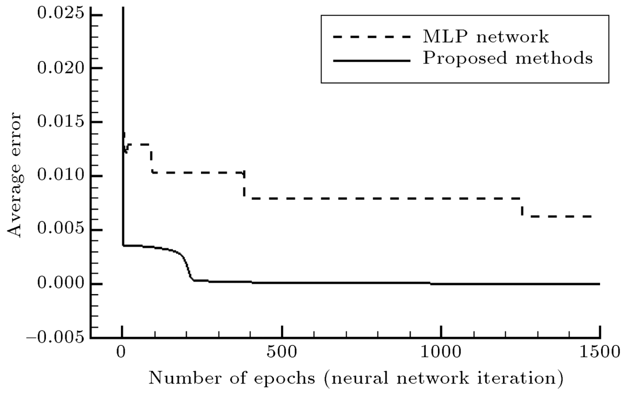


Figure 4. Neural network training error history for the 3-D Sinc function.

Table 1. Comparison of basic NN and proposed NN for the 3-D Sinc function.

	Error	**Computational time (s)**
MLP NN	0.0108	583
Proposed NN	1.351×10^{6}	207

exact function better than the MLP network. The effectiveness and accuracy of the proposed method compared with the conventional NN are also tabulated in Table 1. It is obvious in this table that the new approach improves the results for both accuracy and computational time.

6. Airfoil shape optimization results

To demonstrate the effects of the proposed methods above in the airfoil shape optimization problems and their influence on the accuracy and convergence time, a numerical implementation is carried out. RAE2822 airfoil is considered as the initial airfoil with the flow conditions of Re = 6.5 million, Mach number of 0.75, and the incidence angle of 2.79 degrees. Additionally, in all cases, the objective function is set as C_L/C_D. A widespread domain of change for the airfoil parameters is considered in this work that needs additional constraint in order to avoid the creation of thin and impractical airfoils. The geometry constraints are as follows:

$$0.006 \leq r_{\mathrm{LE}} \leq 0.0115 \qquad 0.35 \leq X_{\mathrm{UP}} \leq 0.69$$
$$0.035 \leq Y_{\mathrm{UP}} \leq 0.095 \qquad -0.9 \leq Y_{xx\mathrm{UP}} \leq -0.1$$
$$0 \leq \alpha_{\mathrm{TE}} \leq 10 \qquad 0 \leq \beta_{\mathrm{TE}} \leq 20$$
$$0.2 \leq X_{\mathrm{LO}} \leq 0.45 \qquad -0.09 \leq Y_{\mathrm{LO}} \leq -0.035$$
$$0.1 \leq Y_{xx\mathrm{LO}} \leq 0.9 \qquad 0 \leq Y_{\mathrm{TE}} \leq 0.03$$

The additional constraint is that the maximum thickness of the generated airfoil is at least equivalent to 95% of the thickness of the initial airfoil, because the initial airfoil is RAE2822 and can be seen as a reference in aerodynamic design problems. Triangular unstructured grids are used for discretizing the computational field. Figure 6(a) shows the grid generated around the initial airfoil. The final grid for the optimum airfoil is illustrated in Figure 6(b).

The airfoil shapes and surface pressure coefficient distributions are shown in Figure 7. It can be seen that the suggested new NN structure can successfully estimate the objective function in comparison with the CFD results. However, the conventional MLP NN has failed to accurately predict the fitness function. The Mach number contours for the initial RAE2822 airfoil are compared with the optimized ones in Figure 8. As illustrated, the shock wave on the upper surface of the initial airfoil is weakened in the new method, which in turn increases the aerodynamic efficiency of the airfoil. The MLP NN results are presented in Figure 8(c) that show only small improvements compared with the initial airfoil.

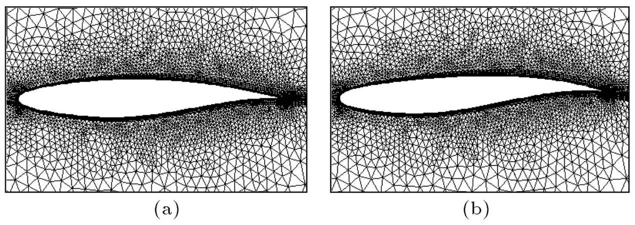


Figure 6. Unstructured grid around (a) RAE2822 airfoil and (b) the optimum airfoil.

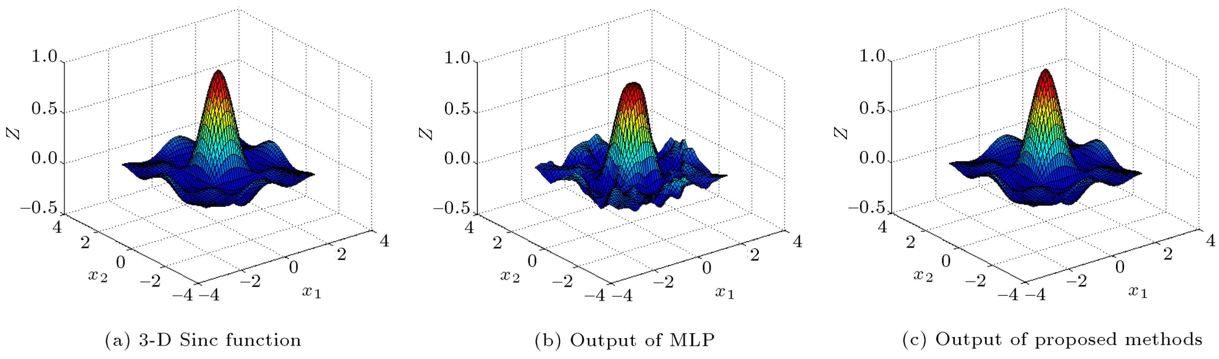


Figure 5. (a) The exact 3-D Sinc function. (b) Output of MLP network. (c) Output of the proposed methods.

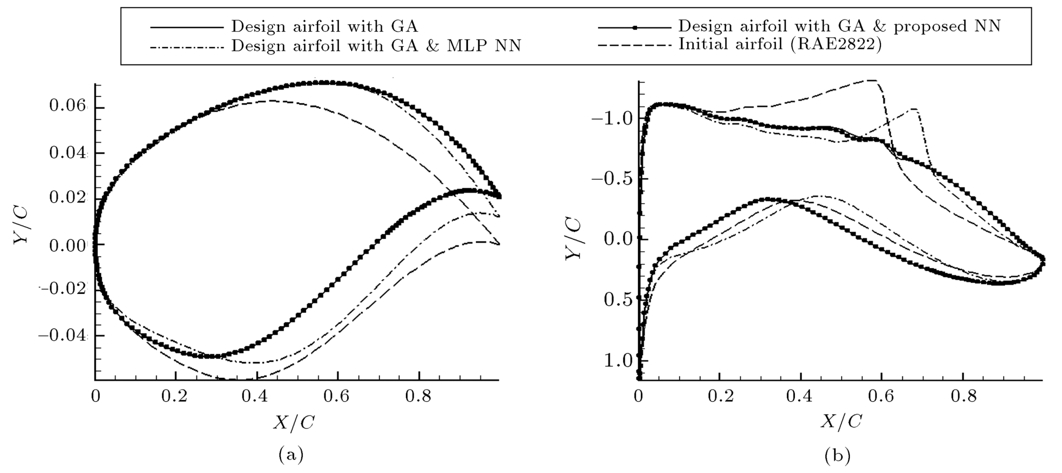


Figure 7. Comparison of (a) design airfoils and (b) surface pressure coefficient distributions.

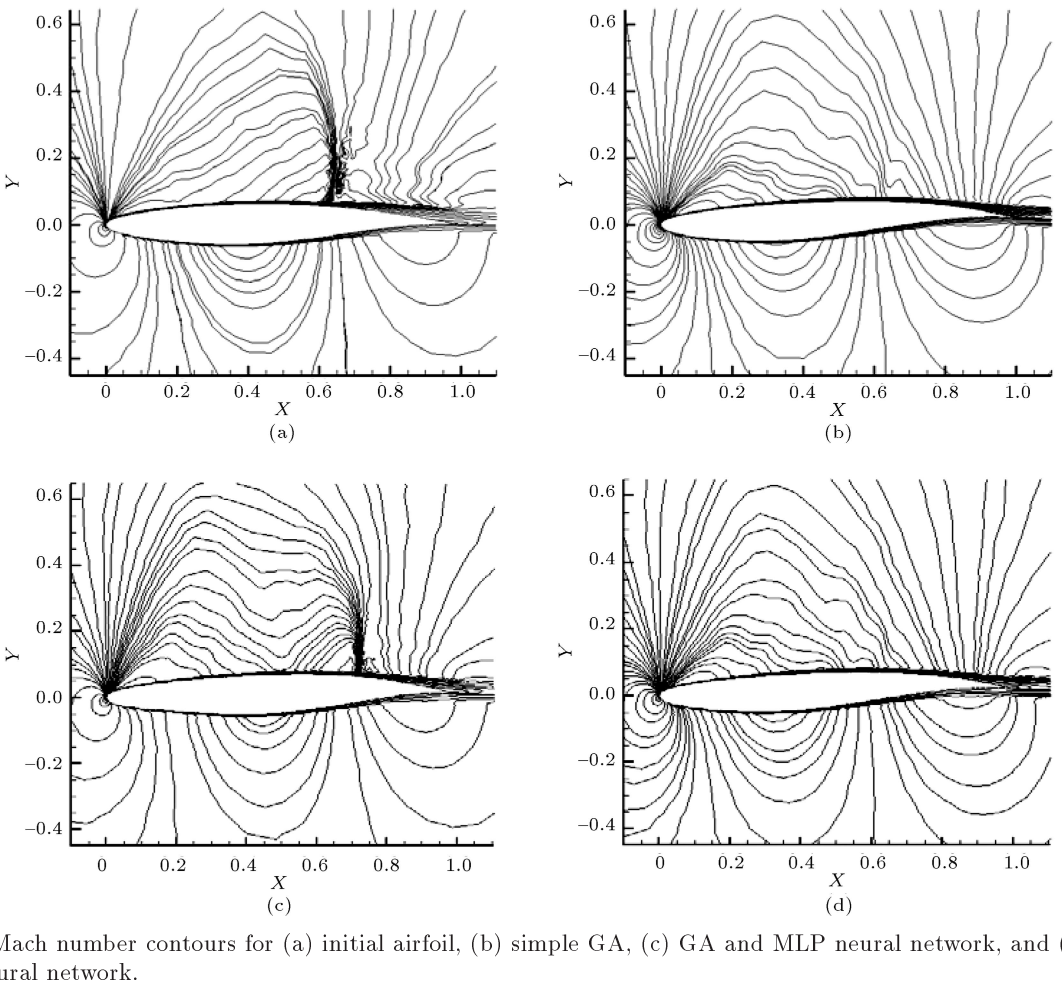


Figure 8. Mach number contours for (a) initial airfoil, (b) simple GA, (c) GA and MLP neural network, and (d) GA and proposed neural network.

Figure 9 illustrates the history of the maximum objective functions for the simple GA (with CFD fitness evaluator) and the application of GA with both conventional MLP and proposed NN methods. According to this figure, using the MLP method for all chromosomes of generations leads the network to the attractor points whose occurrence is usual and is one of the expected problems for the MLP methods. However, Figure 9 confirms that the new method is very desirable and its estimation ability is equivalent to the results obtained from the so called timely exact CFD method.

Table 2 shows, C_L, C_D, and C_L/C_D values for the initial and optimum airfoils obtained from conven-

Table 2. The aerodynamic coefficients for the initial and optimum airfoils.

	C_L	C_D	C_L/C_D	No. of generations calculated by CFD
Initial airfoil	0.822	0.0278	29.54	–
Design airfoil using GA & CFD	0.786	0.0128	61.27	87
Design airfoil using GA with MLP NN	0.75	0.016	46.98	38
Design airfoil using GA & proposed NN	0.785	0.0128	61.08	38

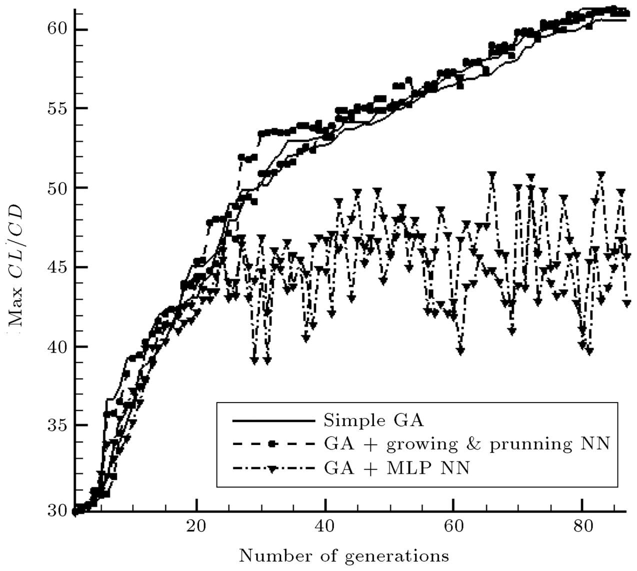


Figure 9. Convergence history of the maximum objective value.

Table 3. The optimum values of PARSEC parameters for simple GA and the new method.

	Simple GA	Proposed method
$\mathbf{r_{LE}}$	0.0110	0.0110
$\mathbf{X_{UP}}$	0.5654	0.5652
$\mathbf{Y_{UP}}$	0.0710	0.0710
$\mathbf{Y_{xxUP}}$	– 0.3625	–0.3690
$\boldsymbol{\alpha}_{\mathbf{TE}}$	10	10
$\boldsymbol{\beta}_{\mathbf{TE}}$	9.7194	9.7180
$\mathbf{X_{LO}}$	0.2827	0.2826
$\mathbf{Y_{LO}}$	–0.0490	–0.0490
$\mathbf{Y_{xxLO}}$	0.8662	0.8662
$\mathbf{Y_{TE}}$	0.0207	0.0207

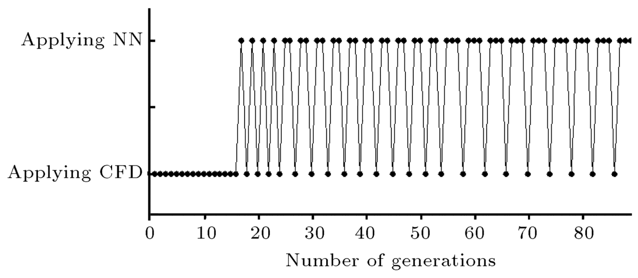
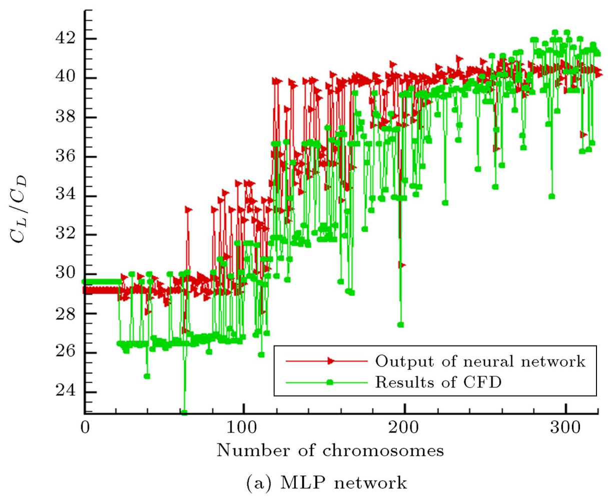
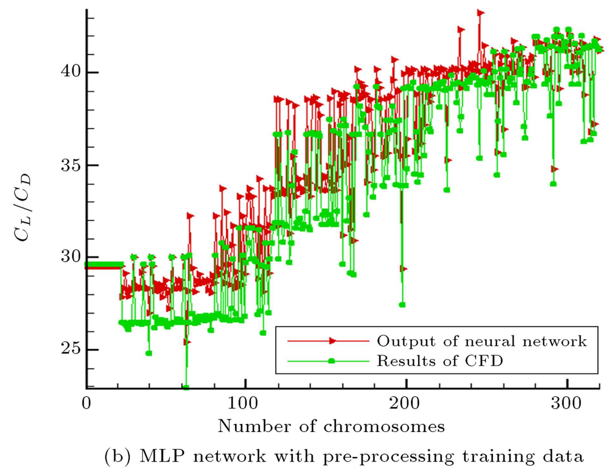


Figure 10. The pattern of applying NN methods.

tional GA and the new network offered in this paper. Table 2 shows that after applying the optimization process with simple GA and CFD flow solution, the value of C_L/C_D increases by about 107% compared with the initial airfoil. While the new optimization method by the help of the proposed network gives similar efficiency in terms of the accuracy of the results, it reduces the required number of CFD solutions by more than 56%. As expressed before, in the aerodynamic optimization process, saving the computational time is a desirable goal. The computational time required for optimization using the simple GA and the new hybrid method on a PC Core i7 with the speed of 3.7 GHz is about 51 hours, while the corresponding time for the simple GA with CFD takes more than 132 hours on the same computer. It can be seen that about 62% reduction in computational time is achieved when using the new method.

The parametric coefficients of PARSEC for the optimum results obtained from simple GA and the new hybrid method are tabulated in Table 3. It is clear that two optimum airfoils have approximately equal parameters. As mentioned before, in this paper, a high quality NN is proposed with the purpose of estimating the results of the flow solver. Actually, this network plays a replacement role for the CFD solver. The pattern of the NN and CFD solvers employment is shown in Figure 10. As illustrated in Figure 10, during the process of optimization for the first 16 generations, the CFD flow solver is used as the fitness function evaluator. The outputs of these 16 generations, which are equivalent to 320 CFD outputs, will then be given to the network for training. The rest of the optimization generations are implemented by the help of both NN and CFD flow solvers in a manner that is shown in Figure 10. In order to show the quality of the proposed NN methods, the following investigations are presented for this case. An accuracy study is initially carried out by comparing the accuracy of the trained data by NNs with the corresponding CFD evaluations. Figure 11





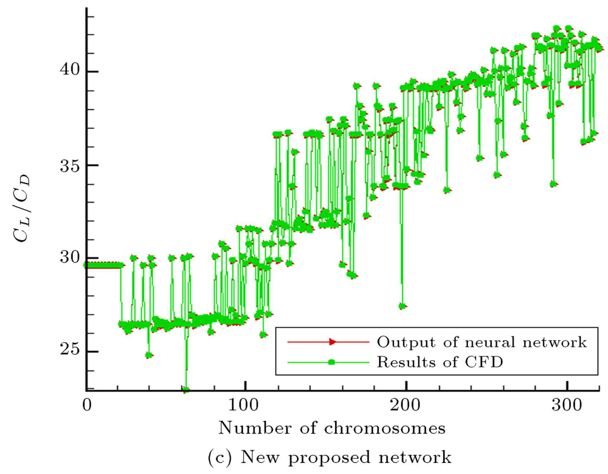


Figure 11. Network capability of the training data.

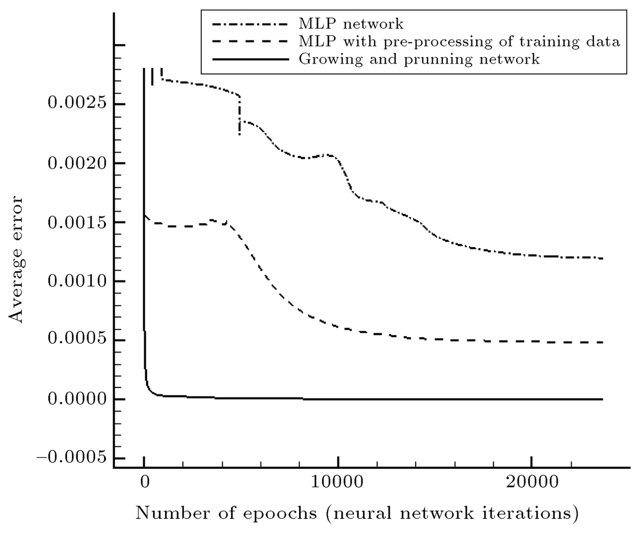


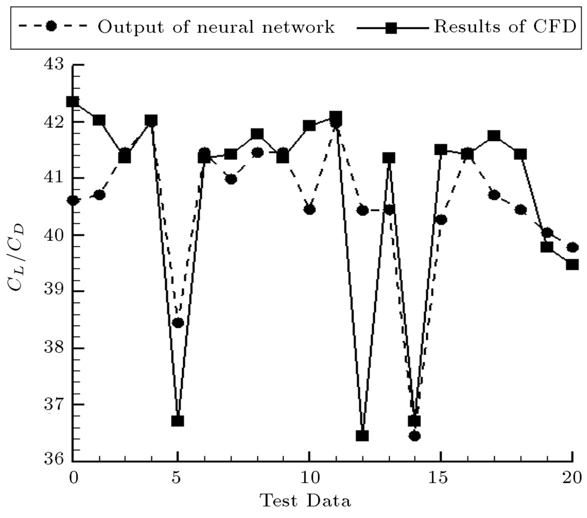
Figure 12. Neural network training error history.

compares the results of the network evaluation for the first 16 generations with the accurate results of CFD. These diagrams are plotted for all NN methods that have been mentioned in this paper. Figure 11(a) illustrates that the conventional MLP network has not correctly calculated the results and there are considerable differences between the outputs of the neural network and the results of CFD. The amount of error is reduced by applying the first proposed method, so called input data modification in Figure 11(b). Finally, as shown in Figure 11(c), using the new approach for the network structure, the error is substantially eliminated. The training efficiency of the proposed methods can be demonstrated in Figure 12 in comparison with the MLP network. It is obvious from Figure 12 that the training error for the proposed methods is less than that of the MLP network.

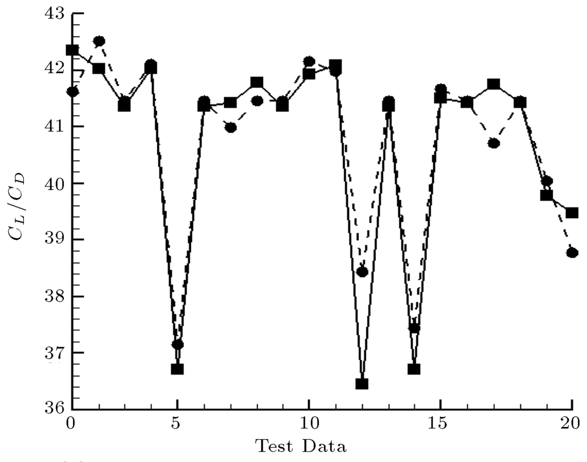
Finally, the estimation ability of the methods is evaluated by comparing the results that are not included in the training data. Figure 13(a) shows that the MLP network can only estimate the results in some chromosomes. This ability is improved by applying the modification to the input data in Figure 13(b). However, in Figure 13(c), it is obvious that the new structure is very strong in estimating the results for all the test data and the related error in this case can be neglected. Note that the test data in Figure 13 are related to the generation number 17.

7. Conclusions

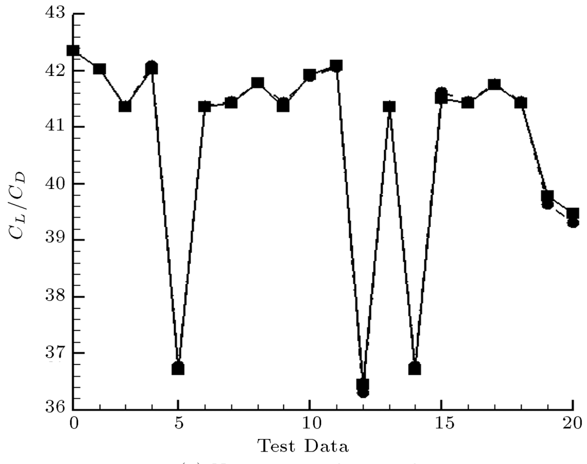
Two new methods were proposed for the NN for airfoil shape optimization to enhance the estimation abilities of the conventional Neural Network (NN) method: a normalized training input data method with the aim of improving the MLP results and a new structure for the neural network. These methods were then applied to the aerodynamic shape optimization and the results were compared with the conventional MLP method. The results indicated that the proposed method was capable of increasing the accuracy of the network and reducing the computational effort by more than 60% compared with the simple GA.



(a) MLP network



(b) MLP network with pre-processing the training data



(c) New proposed network

Figure 13. Generalizing capability of the network.

References

1. Quagliarella, D. and Cioppa, A.D. "Genetic algorithm applied to the aerodynamic design of transonic airfoils", *Journal of Aircraft*, **32**(4), pp. 889-891 (1995).

2. Mukesh, R., Pandiyarajan, R., Selvakumar, U. and Lingadurai, K. "Influence of search algorithms on aerodynamic design optimization of aircraft wings", *International Journal of Soft Computing*, **7**(2), pp. 79-84 (2012).

3. Goldberg, D.E., *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison-Wesley (1989).

4. Deb, K., *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, Chichester, UK (2001).

5. Sasaki, D., Obayashi, S. and Nakahashi, K. "NS optimization of supersonic wings with four objectives using evolutionary algorithm", *Journal of Aircraft*, **39**(4), pp. 621-629 (2002).

6. Shahrokhi, A. and Jahangirian, A. "Airfoil shape parameterization for optimum N.S. design with genetic algorithm", *Aerospace Science and Technology*, **11**(6), pp. 443-450 (2007).

7. Ebrahimi, M. and Jahangirian, A. "Aerodynamic optimization of airfoils using adaptive parameterization and genetic algorithm", *Journal of Optimization Theory and Applications*, **162**, pp. 257-271 (2014).

8. Oyama, A., Obayashi, S. and Nakahashi, K. "Real-coded adaptive range genetic algorithm applied to transonic wing optimization", *Applied Soft Computing*, **1**(3), pp. 179-187 (2001).

9. Ebrahimi, M. and Jahangirian, A. "A hierarchical parallel strategy for aerodynamic shape optimization with genetic algorithm". *Scientia Iranica, Transactions D: Computer Science & Engineering and Electrical Engineering*, **22**(6), pp. 2379-2388 (2015).

10. Tse, D.C.M. and Chan, L.Y.Y. "Application of micro genetic algorithms and neural networks for airfoil design optimization", *Aerodynamic Design and Optimization of Flight Vehicles in a Concurrent Multi-Disciplinary Environment*, **23**, pp. 1-11 (1999).

11. Duvigneau, R. and Visonneau, M. "Hybrid genetic algorithms and artificial neural networks for complex design optimization in CFD", *International Journal for Numerical Methods in Fluids*, **44**, pp. 1257-1278 (2004).

12. Vatandas, E. "Hybridizing genetic algorithm with artificial neural network in the aerodynamic optimization of the forward swept wing",*51st AIAA/ASME /ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference BR 18th*, Orlando, Florida, pp. 12-15 (2010).

13. Pandey, G. and Dukkipati, A. "Learning by stretching deep networks", *Proceedings of the 31th International Conference on MachineLearning*, Beijing, China (2014).

14. Kim, Y. "Convolutional neural networks for sentence classification", *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746-1751 (2014).

15. Karakasis, M.K. and Giannakoglou, K.C. "On the use of metamodel-assisted, multi-objective evolutionary algorithm", *Engineering Optimization,* **38**(8),pp. 941-957 (2006).

16. Shahrokhi, A. and Jahangirian, A. "A surrogate assisted evolutionary optimization method with application to the transonic airfoil design", *Engineering Optimization,* **42**(6), pp. 497-515 (2010).

17. Tes, D. and Chan, Y.Y. "Multi_ point design of airfoil by genetic algorithm", *8th Annual Conference of the CFD Society of Canada*, Montreal, Canada (2000).

18. Jahangirian, A. and Hadidoolabi, M. "Unstructured moving grids for implicit calculation of unsteady compressible viscous flows", *International Journal for Numerical Methods in Fluids,* **47** (10-11), pp. 1107-1113 (2005).

19. Jahangirian, A. and Johnston, L.J. "Automatic generation of adaptive unstructured grids for viscous flow applications",*5th International Conference on Numerical Grid Generation in CFD*, Mississippi State University, pp. 219-228 (1996).

20. Haykin, S., "*Neural Networks: A Comprehensive Foundation*", Prentice Hall (1999).

21. Minsky, M. and Papert, S. "Perceptrons", *An Introduction to Computational Geometry*, M.I.T. Press, Cambridge, Mass (1969).

22. Benardos, P.G. and Vosniakos, G.C. "Optimizing feed forward artificial neural network architecture", *Engineering Application of Artificial Intelligence,* **20**, pp. 365-382 (2007).

23. Azam, F. *Biologically Inspired Modular Neural Networks*, Blackburgs, Virgina (2000).

24. Talebzadeh, M. and Seyyedsalehi, S.A. "Layer by layer optimizer hybrid algorithm for constructing and extracting optimization components in neural network with deep structure", *6th International Conference on Cognitive Science*, Tehran, Iran. pp. 27-29 (2015).

25. Hagan, M.T. and Menhaj, M.B. "Training feed-forward networks with the marquardt algorithm", *IEEE Transactions on Neural Networks,* **5**(6), pp. 989-993 (1994).

Biographies

Narjes Timnak received an MS degree in Aerospace Engineering from Amirkabir University of Technology (AUT), Tehran, Iran, where she is currently a PhD student. Her research interests include aerodynamic optimization methods using evolutionary and surrogate assisted algorithms.

Alireza Jahangirian received BS degree in Mechanical Engineering from Amirkabir University of Technology (AUT), Tehran, Iran, in 1988; MS degree in Mechanical Engineering from Sharif University of Technology, Tehran, Iran, in 1992; and PhD degree from Manchester University, England, in 1997. In 1998, he joined the Department of Aerospace Engineering at AUT, where he is currently Associate Professor. His research interests include computational fluid dynamics, grid generation, and evolutionary aerodynamic optimization.

Seyed Ali Seyyedsalehi received his BS and MS degrees both in Electrical Engineering from Sharif University of Technology, Tehran, Iran, in 1982 and from Amirkabir University of Technology, Tehran, Iran, in 1989. Also, he received his PhD degree in Biomedical Engineering from Tarbiat Modarres University, Tehran, Iran, in 1996.Dr. Seyyedsalehi's research interests are in the areas of speech processing and recognition, biological and artificial neural networks, neural modeling, and linear and nonlinear signal processing. He is now an Associate Professor in the Faculty of Biomedical Engineering at Amirkabir University of Technology.