AIAA JOURNAL
Vol. 45, No. 9, September 2007

Fast Evolutionary Algorithm for Airfoil Design via Neural Network

Abdurrahman Hacioglu*
Turkish Air Force Academy, 34149 Istanbul, Turkey

DOI: 10.2514/1.24484

The main disadvantage of evolutionary algorithms for airfoil design problems is the high computational costs associated with the use of computational fluid dynamics solvers. This disadvantage was significantly reduced by using an existing technique which combines a genetic algorithm and a neural network to rapidly improve populations. In this technique, both genetic algorithms and a properly trained neural network search the design space, resulting in an interactive process between a genetic algorithm and a neural network that greatly improves the exploration power of the algorithm. Experimental results show that the implemented algorithm discovers the desired solution quickly and significantly reduces the overall costs of solving the problem with little or no reduction in robustness.

Nomenclature

AF_k	=	average fitness values at current step of genetic process
AF_0	=	average fitness values at initial step of genetic process
Cp	=	pressure coefficient
Cpc	=	calculated pressure coefficient distribution along airfoil
Cpt	=	target pressure coefficient distribution along airfoil
c	=	chord length of airfoil
gn	=	total gene number of chromosome
IP	=	vibrational mutation period during genetic process
i	=	gene (control point) index of chromosome
$J(s)$	=	objective function for inverse airfoil design problem
$J_{\mathrm{LO}}(s)$	=	objective function for lower surface of airfoil
$J_{\mathrm{UP}}(s)$	=	objective function for upper surface of airfoil
M	=	Mach number
MA	=	main amplitude of vibrational wave
m	=	order of Bezier curve
n	=	population size
P_m	=	mutation rate
s	=	arclength coordinate of airfoil
t	=	parameter in Bezier curve whose values vary uniformly between [0,1]
u	=	random real number between [1,0]
$w1$	=	user defined real number between [0, 2] to control MA
x/c	=	nondimensional x coordinate of airfoil
(x, y)	=	profile coordinates which are produced by Bezier curve representation
(x_i, y_i)	=	coordinates of control points
y/c	=	nondimensional y coordinate of airfoil
y_i^m	=	ith control points (gene) of individual m
$\phi(s)$	=	fitness function of airfoil for inverse airfoil design problem
$\phi_{\mathrm{LO}}(s)$	=	fitness function for lower surface of airfoil
$\phi_{\mathrm{UP}}(s)$	=	fitness function for upper surface of airfoil

Received 7 April 2006; accepted for publication 14 May 2007.

*Major and Associate Professor, Department of Aeronautical Engineering; hacioglu@hho.edu.tr.

I. Introduction

THE field of computational fluid dynamics (CFD) has become an integral part of the aerodynamic design process over the past three decades, and computational solution methods have come to play an increasingly dominant role in the entire engineering design process. Engineers have used CFD technology in aerodynamic design and optimization by coupling it with numerical optimization methods such as gradient-based methods [1–7], adjoint formulations based on control theory [8,9], and stochastic methods [10–23] (e.g., evolutionary algorithms, simulated annealing, artificial neural networks). Optimization methods do not rely on the computation of gradients, because the design problem can be characterized by a mix of continuous, discrete, and integer design variables, and the resulting design space can be nonconvex or even disjointed [15,24]. For this reason, stochastic methods, and, in particular, evolutionary algorithms (EAs), have received considerable interest [22]. Their robustness and easy application to a broad class of problems has made EAs useful tools for tackling complex problems in the aerospace field, regardless of such difficulties as the presence of multiple local minima in the design space and the mixing of continuous and discrete variables [10,15,16,18,19,24]. However, high computational costs associated with the use of high-fidelity simulation models pose a serious impediment to the successful application of EAs to engineering design optimization [16,17,20]. Computationally expensive problems arise in aerodynamic design problems as well as in structural design, electromagnetics, and in the design of coupled multidisciplinary systems [17]. Nevertheless, the availability of fast computing resources or the use of hybrid techniques [10,15–17,19,21,22] has made the power of EAs available even to computation-intensive problems [16].

One of the hybrid techniques which can make EAs useful for computationally expensive problems is to use neural networks (NNs) in a surrogate model [17,25]. Surrogate models are built to approximate computationally expensive simulation codes [17]. Because of being orders of magnitude cheaper to run, NNs can be used in lieu of computationally expensive CFD codes during an evolutionary search. However, since the objective function evaluations of EAs are performed using approximate solutions, the performance of the surrogate model depends on the success of NNs in approximating the exact simulation codes. Additionally, searching the design space and achieving the desired solution are limited to EA operations such as mutation and recombination (crossover).

A new hybridization technique has been proposed [26,27] to employ NNs and EAs together to solve the inverse airfoil design problem. This technique uses a NN to predict candidate solutions during the genetic algorithm (GA) process, rather than to make inexact CFD evaluations as in the surrogate model. A new GA using this technique was named "augmented genetic algorithm with neural

network (AGANN)," and its application to internal flow systems design was demonstrated in [27]. Combining the exploration power of GA with the extrapolation/interpolation ability of NN to obtain the desired solution is the main advantage of the AGANN.

The objective of this paper is to reduce the high computational costs of EAs by applying AGANN to the inverse airfoil design problem. Additionally, the performance of AGANN could be improved by using the distribution strategies (DS) [28] concept in the NN phase. The DS concept is widely known, and its implementation to the GA process was established in [28]. To demonstrate that AGANN can significantly reduce the overall costs of solving the problem, the proposed method was tested using four test cases.

This paper is organized as follows: In Sec. II, inverse airfoil design operations are explained to ensure understanding of both AGANN and application of the DS concept to GA and NN processes. In Sec. III, AGANN technique, GA and NN methods used by AGANN, and implementation of DS in the NN phase are described. Data and experimental details from the test cases are presented in Secs. IV and V, followed by a detailed analysis in Sec. VI and an interpretation of these results in Sec. VII.

II. Inverse Airfoil Design Operations

A. Airfoil Representation

In an inverse airfoil design problem, a very important issue is the representation of the geometry of the airfoil. For experiments, I used a Bezier curve representation, by means of which can be defined as a closed curve (i.e., the airfoil) with a set of $(m + 1)$ control points. Its expression is given by the following equations:

$$y(t) = \sum_{i=0}^{m} C_m^i t^i (1-t)^{m-i} y_i \tag{1}$$

$$x(t) = \sum_{i=0}^{m} C_m^i t^i (1-t)^{m-i} x_i \tag{2}$$

where $C_m^i = m!/[i!(m-i)!]$ and t is the parameter of the curve whose values vary uniformly between [0,1]. The variables (x_i, y_i) are the coordinates of the control points which define the profile coordinates $(x(t), y(t))$. The two control points (0,0) and (1,0) at the leading and trailing edges are fixed. Assuming x_i as fixed, the design parameters coded in the GA are thus limited to the y_i coordinates of the control points.

B. Objective and Fitness Function

The objective function of the GA process for an inverse airfoil design problem can be defined as follows:

$$J(s) = \sum_{i} [(Cpc_{i+1/2} - Cpt_{i+1/2})\Delta s_i]^2 \tag{3a}$$

where Cpc is the calculated Cp distribution of an individual, and Cpt is the target Cp distribution along the target airfoil shape. Index $(i + 1/2)$ indicates the midpoint between locations (i) and $(i + 1)$ at the airfoil surface. Using the profile coordinates (x, y), the arclength at the ith location of the airfoil is evaluated as follows:

$$\Delta s_i = [(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2]^{\frac{1}{2}} \tag{3b}$$

The fitness function of an individual is proportional to the inverse of the objective function, that is,

$$\phi(s) = \frac{1}{J(s)} \tag{4}$$

Thus, the problem becomes a typical maximization problem and a fitness of infinite value would mean that the two Cp shapes are identical.

C. Application of the DS Concept to the GA Process

The DS concept is widely described, proofs for it are published in [28,29], and it is alternately known as "the distribution of objective function" and "the distribution of elitism" concepts. The essence of DS is to divide the chromosome of each individual into a certain number of gene groups and to calculate the fitness value for each gene group separately. Besides carrying distinct calculations throughout the genetic processes, elitism can also be implemented distinctly for the gene groups of the individuals. To clarify the relationship to the Bezier curve representation of an airfoil explained above, the following explorations are needed. Each surface of the airfoil is represented by its control points. Based on that assumption, chromosomes which are used to represent the profile can be as follows:

$$\left.\begin{array}{ll} \text{upper half control points} & \{y_1, y_2, \ldots, y_{gn}\}_{\text{upper}} \\ \text{lower half control points} & \underbrace{\{y_1, y_2, \ldots, y_{gn}\}_{\text{lower}}}_{\text{genes}} \end{array}\right\} \text{chromosome}$$

As seen from this representation, all the chromosomes representing the whole profile are decomposed as the "upper half control points" (first gene group) and the "lower half control points" (second gene group) so that the objective function and the fitness values will be calculated separately. Specifically, the objective functions for the upper and lower surfaces can be defined as

$$J_{\text{UP}}(s) = \sum_{i(\text{UP})} [(Cpc_{i+1/2} - Cpt_{i+1/2})\Delta s_i]^2 \tag{5a}$$

$$J_{\text{LO}}(s) = \sum_{i(\text{LO})} [(Cpc_{i+1/2} - Cpt_{i+1/2})\Delta s_i]^2 \tag{5b}$$

Similarly, the fitness values for the upper and lower surfaces are as follows:

$$\phi_{\text{UP}}(s) = \frac{1}{J_{\text{UP}}(s)} \tag{6a}$$

$$\phi_{\text{LO}}(s) = \frac{1}{J_{\text{LO}}(s)} \tag{6b}$$

In a classical GA process, the chromosome (including both upper and lower halves) which has the greatest fitness value is carried into the next generation as a result of classical elitism. Additionally, when the DS concept is used, the best upper surface (first gene group) and the best lower surface (second gene group) are carried into the next generation as a result of elitism distribution. These upper and lower surfaces thus combine to form an individual (airfoil) for the next generation.

III. Augmented Genetic Algorithm with Neural Network

NNs are widely used in aerodynamic design problems because they can be used to emulate highly nonlinear relationships that exist between aerodynamic configurations and their corresponding aerodynamic performance. This feature of NNs can be used to predict the desired airfoil during a GA-based design process because NNs can extrapolate (or interpolate) the candidate solutions (airfoils) from within the population to reach the desired solution. The aim of the AGANN is to augment the population of GA by making use of the prediction ability of NNs. Adding a predicted candidate, which emulates the desired solution, to the population realizes this aim. Additionally, both GAs and NNs search the design space during the design process.

For each generation of the GA process, the NN uses candidates (airfoil geometries) in the population and their objective function solutions (i.e., pressure coefficient Cp, distributions) as a training set. For an inverse airfoil design problem, the training set includes

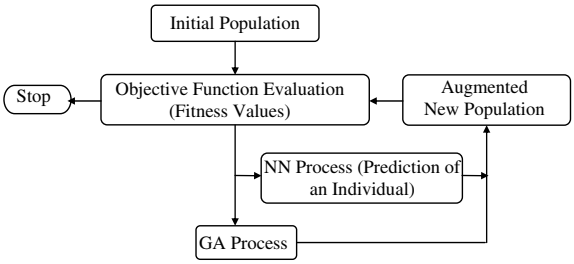

Fig. 1 Block diagram of the AGANN.

airfoil geometries as the output, and their Cp distributions as the input. By using the target Cp distribution as an input, the trained NN predicts a corresponding airfoil which is a candidate solution of the inverse design problem. Because of the interpolation/extrapolation ability of NN, this predicted individual is probably more suitable than the candidate solutions in the training set population and can augment the new population of GA. The augmented population of AGANN brings about a rapid improvement of the GA process, speeds the finding of a solution, and reduces the overall cost of the problem (Fig. 1). For an inverse airfoil design process, the main steps of AGANN can be itemized as follows:

1) First, Cp distributions of the airfoils in the current population are evaluated to determine their objective function values. The selection operation in the GA is carried out by using these objective function values, and the new population is reproduced by GA operations such as recombination (crossovers) and mutations.

2) Second, the NN is trained by using the airfoils and their Cp distributions in the current population. For this training, the Cp distributions are the input, and the control points in Eq. (1), which define the airfoils, are the output.

3) Lastly, the target Cp distributions are used as an input to the trained NN to determine the control points of the corresponding airfoil. This corresponding individual becomes part of the new population to be used as a candidate in the next GA cycle.

This procedure is repeated during the GA process.

A. Neural Network Method

Backpropagation neural network (BPNN) [30] is the generalization of the least mean square algorithm to multiple-layer networks and nonlinear differentiable transfer functions. The multilayer feedforward network is the most-used architecture of BPNN. Feedforward networks typically consist of one or more hidden layers of sigmoid neurons followed by a layer of linear neurons.

BPNN has good generalization and extrapolation abilities and is appropriate for the inverse design problem. Figure 2 shows the architecture of this BPNN, which uses a nonlinear hidden layer with sigmoid (hyperbolic tangent) transfer function and a linear output layer. The details of BPNN can be found in [31]. As mentioned previously, in a set of training data, the input parameters are the Cp distributions of airfoils, while the outputs are the parameters which represent the airfoil geometries. The training of BPNN continues for each step of the GA process using the population on hand. The weights of BPNN obtained from one set of training data are saved and are used as initial values to train the next set of data. This reduces the training errors of BPNN for a fixed amount of training epochs and enables a more comprehensive construction of the response surface for the search space.

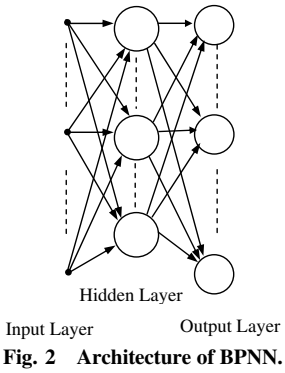

Fig. 2 Architecture of BPNN.

B. Implementation of the DS Concept to the NN Process

During the GA process, some GA operations are performed separately by using the DS concept as mentioned in Sec. II. Similarly, owing to implementation of the DS concept, the single NN process phase in Fig. 1 has to be repeated for the upper and lower parts of the airfoil. Therefore two new processes are added to the NN process (Fig. 3). It should be noted that the NN process produces two individuals at this time. One individual originates from the process performed without DS implementation for the entire airfoil as explained above. The other is produced by the two new processes as a result of DS concept implementation. For the new processes, the first of the main steps of AGANN remains unchanged, whereas in the second, NN is trained by using either upper or lower half geometries of individuals in the current population and their Cp distributions. For this training, Cp distributions of upper or lower halves of the individuals are used as the input, and the control points of the corresponding surface geometries are used as the output. Target Cp distributions of the upper or lower surfaces are then used as input to the trained NN to find the corresponding control points. The corresponding control points of upper and lower surfaces are then combined to form an individual. This individual is placed in the new population to be used as a candidate at the next GA cycle.

C. Genetic Algorithm Method

The vibrational genetic algorithm (VGA) uses the vibration concept which is well defined in [13,29,32]. By applying a vibrational mutation periodically to all individuals in the population, escaping local optimums and thus obtaining global optimum quickly becomes possible. Previous works [13,23,32,33] state that the VGA can be efficiently used for aerodynamic design. It can be concluded from these works that the most important advantages of the VGA are that it is able to use a greater mutation rate and a smaller population size for the GA process.

Vibrational mutation is used after the reproduction phase. It is applied periodically from the initial steps of the genetic process. A vibrational wave is introduced into the population just after the first period. Amplitude of this wave is random. For this operation, entire

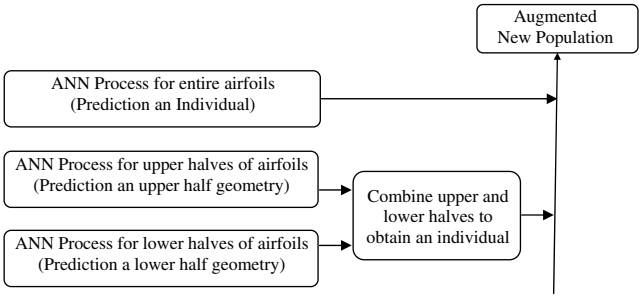

Fig. 3 Implementation of the DS concept to the ANN process.

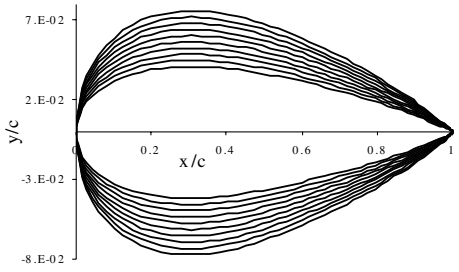

Fig. 4 Initial population.

genes in all chromosomes are mutated due to the vibrational wave as follows:

$$y_i^m = y_i^m \cdot [1 + w1 \cdot MA \cdot (0.5 - u)] \qquad m = 1, \ldots, n \quad i = 1, \ldots, gn \tag{7}$$

where y_i^m are the ith control points (gene) of the individual m; gn denotes the chromosome length (total gene number of a chromosome); n is the total number of individuals in the population (population size); MA is the main amplitude; u is a random real number between [1,0], and $w1$ is a user defined real number between [0, 2] which controls MA.

Implementation of the vibrational mutation starts from a certain gene position at the first chromosome and continues throughout the genes at the same position in the other chromosomes. This process is applied to all the individuals in the population every IP period (generation) of the genetic process. The mutation rate is

$$P_m = \frac{1}{IP} \tag{8}$$

where IP is an integer number. As the population evolves, to spread out the population in a narrower band, the main amplitude MA is changed during the genetic process as follows:

$$MA = \left[\frac{\log(1 + AF_0)}{\log(1 + AF_k)}\right]^r \tag{9}$$

where AF_0 and AF_k are average fitness values at the initial and current steps of the genetic process, respectively, and r is a real number.

IV. Experiments

A. Flow Analysis Solvers for CFD

CFD solutions of an aerodynamic problem can be performed by using different solvers. However, the accuracy of a CFD solution varies according to the flow equation that is solved. For example, full potential solvers, which solve the full potential equation [34], are less realistic due to the assumptions of irrotational and inviscid flow. They use less computational time, though, on the order of seconds (using a today's moderate PC) for the calculation of an airfoil. Because the Euler equations [34] reflect rotational flow properties, a Euler solver, which consumes time in the order of minutes, can produce a more realistic CFD solution. Even though their accuracies are different, these solvers can approximate the main character of the aerodynamic problem. For that reason, the character of the objective function in Eq. (1) does not change when different CFD solvers are used. Consequently, solvers which use less computation time can be used to test the proposed algorithm. Therefore, to demonstrate the effects of solvers, numerical experiments were performed using three different CFD solvers on a Pentium 4 1.7 GHz PC with 512 MB RAM.

1) *Solver-I* is the vortex panel code of Kuethe and Chow [35] for incompressible, inviscid flow, which uses 120 panels for CFD calculations.

2) *Solver-II* is a full potential solver [36] for transonic inviscid flow, which solves the full potential equations using Murman's difference scheme [37] and 161×31 structured O-mesh. For the following test cases (test case-II and case-III), this solver makes a CFD calculation in approximately 0.5 s.

3) *Solver-III* is a Euler solver for transonic inviscid flow, which solves the time-dependent Euler equations by using cell-centered finite volume space discretization on a structured 121×41 O-mesh, Roe flux splitting [38], and an explicit five-stage Runge–Kutta scheme. For test case-III, this solver makes a CFD calculation in approximately 75 s.

B. Test Cases

To show its robustness and effectiveness, the proposed method was tested for four inverse design cases using two different airfoils for both incompressible and transonic flow:

1) *Case-I* (subsonic flow/Solver-I): The C_p distribution of RAE2822 airfoil, evaluated at incompressible flow and an angle of attack of 2 deg, is presented. The simple panel code (Solver-I) can be used for this subsonic, inviscid flow.

2) *Case-II* (transonic flow/Solver-II): The C_p distribution of RAE2822 airfoil, at Mach number $M = 0.725$ and an angle of attack of 2 deg, is presented. Solver-II is used for this transonic, inviscid flow.

3) *Case-III* (transonic flow/Solver-II): The C_p distribution of NACA64A410 airfoil, at $M = 0.75$ and an angle of attack of 0 deg, is presented. Solver-II is used for this transonic, inviscid flow.

4) *Case-IV* (transonic flow/Solver-III): The C_p distribution of NACA64A410 airfoil, at $M = 0.75$ and an angle of attack of 2 deg, is presented. Solver-III is used for this transonic, inviscid flow.

C. GA Strategies

The following GA strategies were used for the inverse design cases above:

1) *VGA-DS:* This strategy used only VGA, in which DS are used for the GA process. Population size is 14 (previous studies [13,32,33] established that a small population size and a high mutation rate are more convenient for VGA).

2) *AGANN1:* This strategy used the AGANN technique, in which VGA-DS was employed for the GA process.

3) *AGANN2:* In addition to AGANN1, this strategy included DS implementation to the NN process as explained in Sec. III.B.

The selection method was the stochastic universal sampling (SUS) [39], and the recombination (crossover) technique was BLX-α [40] with $\alpha = 0.7$. For vibrational mutation, the IP values in Eq. (8) were chosen as 2 and 4 for incompressible and transonic cases, respectively, and the value of r in Eq. (9) was taken as 4.

The initial population includes airfoils obtained from NACA 0012 by changing its control point values $\pm 30\%$ uniformly (Fig. 4). In the Bezier curve representation of one surface of an airfoil, in Eq. (1), 9 and 13 control points ($m = 8$ and $m = 12$) were used for the incompressible and transonic test cases, respectively.

V. Results

The results of each test case are the average values of 10 replications. For all test cases, a comparison of the best objective function values (BOFVs) was obtained by using the three GA strategies (Fig. 5). AGANN1 and AGANN2 have a remarkably faster convergence than VGA-DS. Each GA strategy gave a different performance over the reduction of BOFVs by less than 10^{-5}. The data summarized in Table 1 indicate that the proposed method AGANN2 reduced CFD solver calls more than 90% compared to VGA-DS, and by more than 30% compared to AGANN1. This result clearly reflects the contribution of DS implementation during the NN phase to overall computational efficiency. Similar results from the different test cases in Fig. 5 indicate that the AGANN method is highly robust and effective.

A recent study [28] showed that inverse design by means of VGA combined with DS (VGA-DS) reduces the CFD calculation by at least 75% more than conventional GA does. However, the present study demonstrates that AGANN can decrease the CFD solver usage by more than 90%, with the result that the cumulative reduction in CFD solver calls exceeds 97.5% compared to conventional GA. Therefore, the high computational cost of EAs, its major disadvantage, can be greatly reduced using these hybrid methodologies.

The computational costs for test case-I are comparatively lower than the other test cases (Table 1). This likely results from the flow conditions, because the character of the objective function in Eq. (3a) depends on Cp distributions, which are related to flow conditions. For transonic flow conditions, the character of the objective function

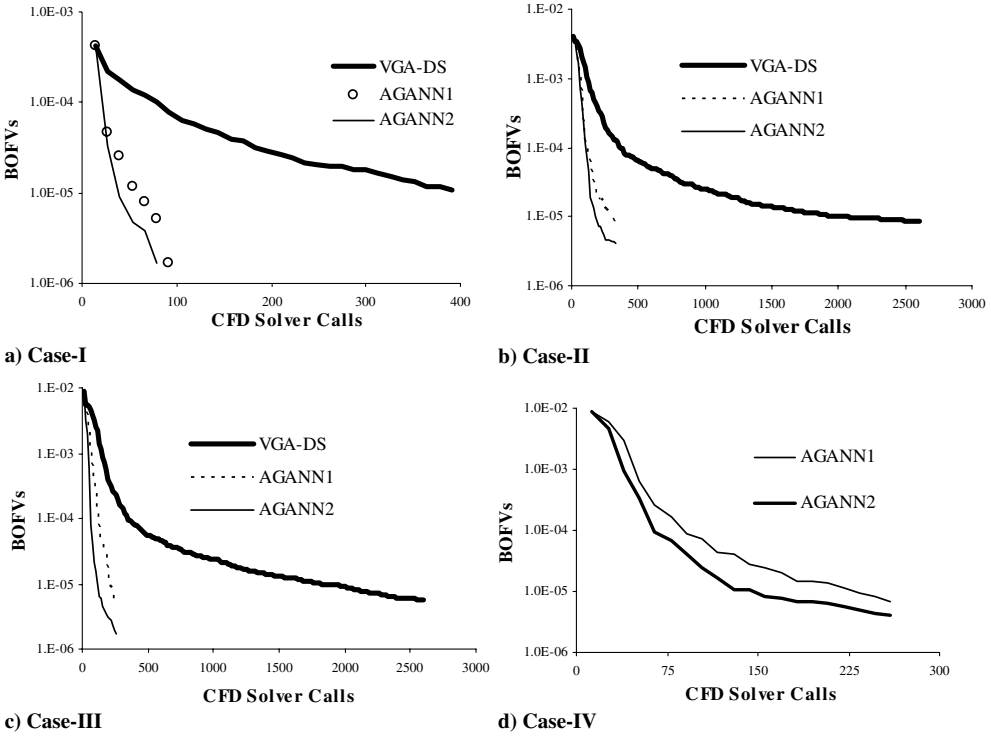

Fig. 5 Comparison of the best objective function values (BOFVs) for different algorithms.

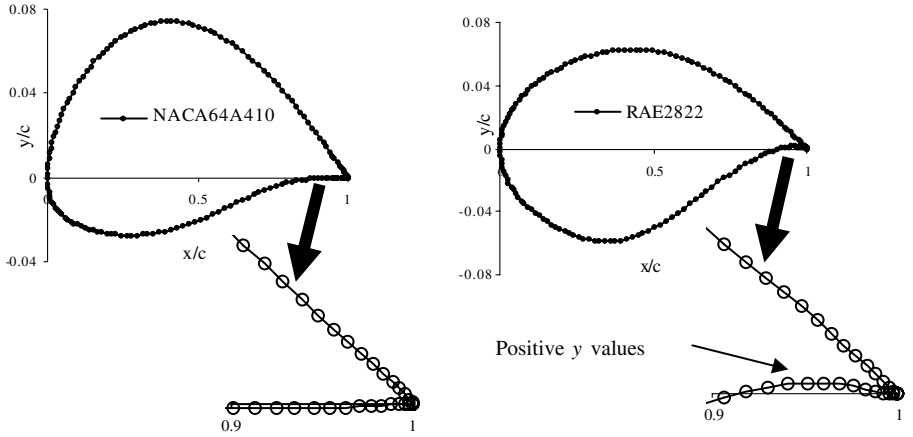

Fig. 6 Trailing edges of NACA64A410 and RAE2822.

is greatly complicated due to the compressibility effect (which increases nonlinearity) and shock wave (which causes discontinuity). However, for incompressible flow, due to the lack of shock wave and compressibility effects, the character becomes relatively simple. For that reason, the computational costs for test case-I are relatively low.

The difference between computational costs of test cases-II and -III are interesting, as both test cases are based on transonic flow, and the same CFD solver was used for both of them. Any cost difference should thus result from the airfoils themselves. By examining the geometries of RAE2822 and NACA64A410, the distinction at the trailing edge of the airfoils can be discerned (Fig. 6). The wedge angle at the trailing edge of RAE2822 is negative, thus causing positive y values around the trailing edge of the lower surface. Using the initial population in Fig. 4, to explore these positive y values requires more computational cost, but because there is no problem at the trailing edge of NACA64A410, test case-II incurs lower computational costs.

Final designs for each case are illustrated in Fig. 7. This figure shows the target profile geometries and optimized geometries produced by the inverse design with pressure coefficients for related geometries.

VI. Analysis

BPNN uses local, gradient-based minimization techniques for training multilayer perceptrons. Gradient-based minimization of the cost function is relatively fast, but for complex problems with a large number of vectors and input features, it has several drawbacks and

Table 1 CFD solver calls

		AGANN2	AGANN1	VGA-DS
Case-I	Solver calls	40	60	430
	Reduction wrt VGA-DS	90.7%	84.7%	——
	Reduction wrt AGANN1	39.4%	——	——
Case-II	Solver calls	194	299	2054
	Reduction wrt VGA-DS	90.5%	85.4%	——
	Reduction wrt AGANN1	34.8%	——	——
Case-III	Solver calls	117	208	1820
	Reduction wrt VGA-DS	93.6%	88.6%	——
	Reduction wrt AGANN1	43.8%	——	——
Case-IV	Solver calls	156	234	——
	Reduction wrt AGANN1	33.4%	——	——

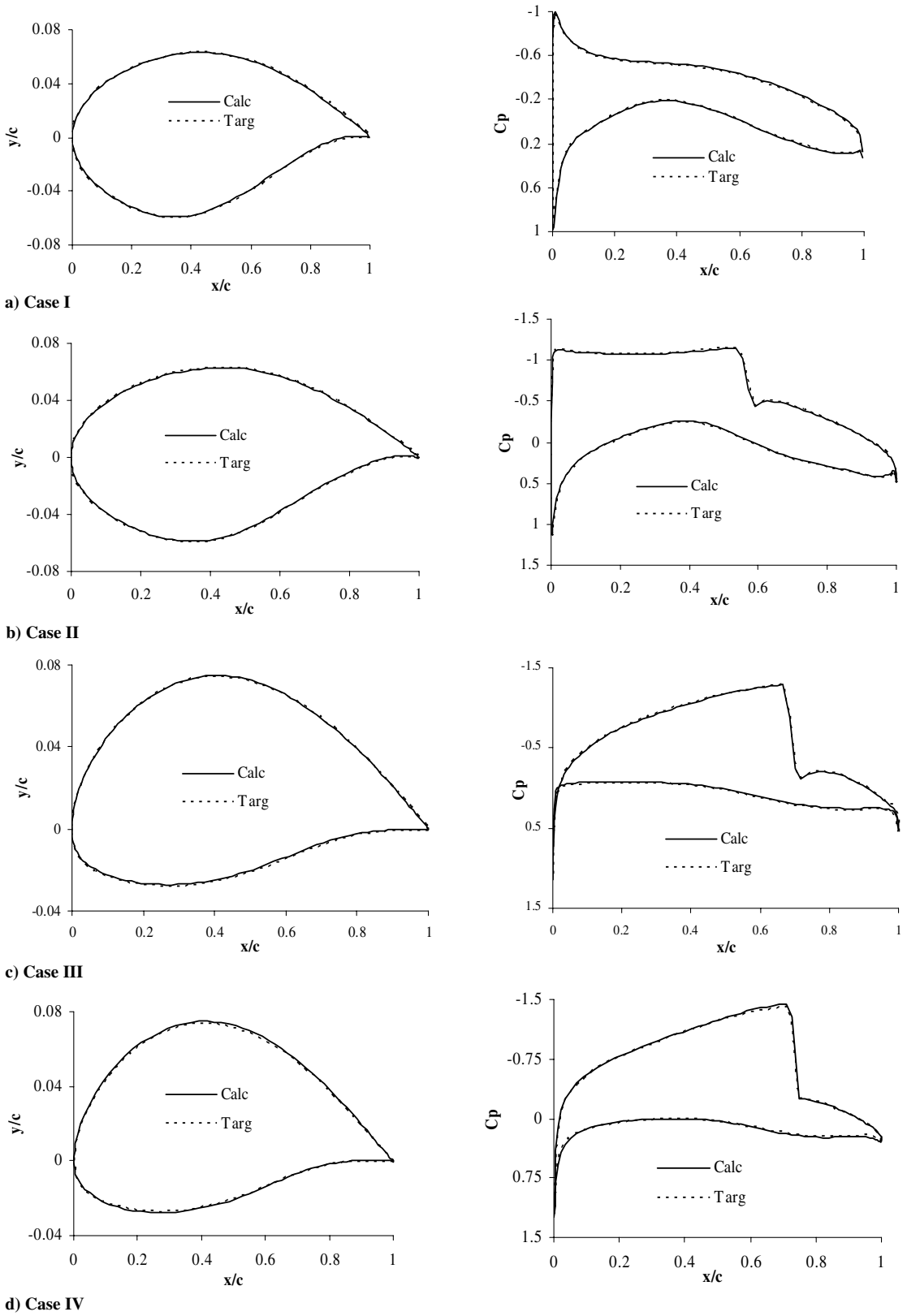

Fig. 7 Calculated/target airfoils and *Cp* distributions.

may be very time consuming. Thus, for an aerodynamic design problem, total time consumption (TTC) may increase excessively, and BPNN may not work accurately. However, these drawbacks do not prevent AGANN from working properly due to the small training set for NN and proximate predictive values for each generation.

In AGANN, the training set for NN is small (for the test cases, the training set includes 14 individuals in the population and their *Cp* distributions), so NN does not need extensive computation time for training. For example, in test case-III, in which a faster (full potential) CFD solver used, time consumption per generation (cycle) is about 6.5 s for VGA-DS without NN usage (Table 2). When AGANN2 is used, time consumption increases to 14.35 s due to double NN usage (Fig. 3). This indicates about an 8 s increase per cycle. However, TTC during the design process is more important. In this point of view,

Table 2 Comparison of time consumption (case-III)

	AGANN2	VGA-DS
Cycles	9	140
Average time consumption per cycle	14.35 s	6.5 s
Total time consumption	(9 × 14.35) 129.15 s	(140 × 6.5) 910 s
Total time reduction wrt VGA-DS	86%	——
Solver calls reduction wrt VGA-DS	93.6%	——

VGA-DS needs 140 generations to reach BOFVs of 10^{-5}, and its TTC is (6.5 × 140) 910 s; whereas AGANN2 requires 9 generations, and its TTC is 129.15 s. This demonstrates that using AGANN2 reduces TTC by 86%. On the other hand, for this test case, the reduction in CFD solver calls of AGANN2 is 93.6%. The difference between the two reduction percentages (86 and 93.6%) is 7.6%. It must be noted that this result is obtained by using a faster CFD solver, which makes a CFD calculation in less than 0.5 s. A slower but more realistic solver reduces this difference close to zero. For example, for Euler solver usage (needs about 75 s per CFD solution), the difference will be about 0.8%. It is obvious that using a more realistic Navier–Stokes solver, although it requires much more time than a Euler solver does, reduces this difference to almost zero. For that reason, at least for an inverse airfoil design problem, considering the reduction in CFD solver calls is enough, and there is no need to compare TTCs.

AGANN does not need excellent prediction from NN (BPNN) for each generation. It is sufficient that NN produces more suitable individuals than the ones already in the population. If NN produces a more suitable individual, then GA processes benefit from it to improve the population. Improved populations can then cause NN to produce individuals more closely fitted to the target. This interactive process between GA and NN provides rapid improvements in the population. On the other hand, if NN produces a less well-fitted individual, the GA process can eliminate it through the selection phase (Table 3).

Table 3 summarizes some statistical data obtained from implementation of the GA strategies to test case-III. The individual source indicates which kind of process produced the individual in the population. For example, when the GA strategy of AGANN2 was used, there were three types of individuals in the population:

1) Individuals from conventional GA operations such as recombination (crossover) or mutation (marked as GA in Table 3).

2) An individual from the distribution of elitism (Sec. II.C, marked as DSE in Table 3).

3) Two individuals from NN (Fig. 3); one from NN operations for the entire airfoil, and the other from the application of DS to the NN process (marked as NN1 and NN-DS in Table 3).

Additionally, the best individual in the previous population is always transferred to the next population as a result of classical elitism. REPEAT indicates how many times the previous best individual repeated during the design process. In other words, REPEAT indicates how many times the process failed to improve population.

For AGANN2, 19 generations (not including the initial population) were allowed for each experiment. Consequently, by using AGANN2, a total of 190 generations were produced during the 10 replications of test case-III. Within those 190 generations, the best individual was produced 42 times by NN1. That is, the number of the best individual produced (BIP) by NN1 is 42 (see NN1 column in Table 3). Its share is 22.11%. However, the average fitness value increasing ratio (AFVIR) of NN1 is 192.8%. That is, when the best individual was found by NN1, its fitness value increased 192.8% on average, with respect to the previous best individual. AFVIR represents the rate of fitness value of the best individual to the previous one and shows the improvement in fitness value.

The total number of BIP by NN (both for NN1 and NN-DS) for AGANN2 is 85 (42 from NN1 and 43 from NN-DS) within 190 generations (Table 3). That is, NN succeeds in predicting a more well-fitted individual 44.7% of the time, or NN is unsuccessful 55.3% of the time due to the disadvantages of BPNN. However, the AFVIR values of NN1 (192.8%) and NN-DS (199.3%) indicate rapid improvement in the fitness value and show that the interactive process between GA and NN works (Table 3). For that reason, BOFVs of AGANN1 and AGANN2 decreased quickly (Fig. 5). Moreover, the data of VGA-DS (Table 3) show that the AFVIR of DSE and GA are 5.4 and 9.0%, respectively, whereas they are 68.9 and 62.9% for AGANN2. That is, the exploration power of GA increased when NN was used. This also reflects the interactive process between GA and NN. Additionally, it should be noted that REPEAT's share increased from 28.42% (in AGANN2) to 49.6% (in VGA-DS) when NN was not used.

VII. Conclusions

In this paper, the AGANN method, which hybridizes genetic algorithms and neural networks, was applied to inverse airfoil design problems. Additionally, the effectiveness of AGANN was improved by applying the distribution strategies methodology to its NN phase. Using NN for augmenting the population, which causes an interactive process between GA and NN, was the main idea of the AGANN strategy. The interactive process develops the exploration ability of the algorithm beyond what is possible with conventional approaches and provides rapid improvement in the population. The numerical experiments indicate that AGANN has a remarkable impact on the number of CFD calculations required for the inverse airfoil design. Consequently, the main disadvantage of EAs for aerodynamic design problems is greatly reduced. Furthermore,

Table 3 Analysis of generation history of case-III

		Individual source of population				
		NN1	NN-DS	DSE	GA	REPEAT
AGANN2	BIP (total 190)	42	43	15	36	54
	Share, %	22.11	22.63	7.89	18.95	28.42
	AFVIR, %	192.8	199.3	68.9	62.9	0
AGANN1	BIP (Total 190)	——	57	26	64	53
	Share, %	——	30	13.68	28.42	32
	AFVIR, %	——	216	37.4	57.22	0
VGA-DS	BIP (Total 1990)	——	——	263	743	987
	Share, %	——	——	13.22	37.19	49.6
	AFVIR, %	——	——	5.4	9.0	0

because AGANN is a GA-based technique, the method is also as robust as pure GAs and may have applications in other inverse design problems.

References

[1] Baysal, O., and Eleshaky, M. E., "Aerodynamic Design Optimization Using Sensitivity Analysis and Computational Fluid Dynamics," *AIAA Journal*, Vol. 30, No. 3, 1992, pp. 718–725.

[2] Cabuk, H., and Modi, V., "Optimum Plane Diffuser in Laminar Flow," *Journal of Fluid Mechanics*, Vol. 237, April 1992, pp. 373–393.

[3] Burgreen, G. W., and Baysal, O., "Aerodynamic Shape Optimization Using Preconditioned Conjugate Gradient Methods," *AIAA Journal*, Vol. 32, No. 11, 1994, pp. 2145–2152.

[4] Mohanga, J., and Baysal, O., "Gradient-Based Aerodynamic Shape Optimization Using Alternating Direction Implicit Method," *Journal of Aircraft*, Vol. 34, No. 3, 1997, pp. 346–352.

[5] Newmann, J. C., Taylor, A. C., Barnwell, R. W., Newmann, P. A., and Hou, G. J.-W., "Overview of Sensivity Analysis and Shape Optimization for Complex Aerodynamic Configurations," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 87–96.

[6] Seokhyun, L., and Haechon, C., "Optimal Shape Design of a Two-Dimensional Asymmetric Diffuser in Turbulent Flow," *AIAA Journal*, Vol. 42, No. 6, 2004, pp. 1154–1169.

[7] Tuncer, I. H., and Kaya, M., "Optimization of Flapping Airfoils For Maximum Thrust and Propulsive Efficiency," *AIAA Journal*, Vol. 43, No. 11, 2005, pp. 2329–2336.

[8] Jameson, A., "Aerodynamic Design via Control Theory," *Journal of Scientific Computing*, Vol. 3, No. 3, 1988, pp. 233–260.

[9] Jameson, A., and Sangho, K., "Reduction of the Adjoint Gradient Formula for Aerodynamic Shape Optimization," *AIAA Journal*, Vol. 41, No. 11, 2003, pp. 2114–2129.

[10] Wang, X., Damodaran, M., and Lee, S. L., "Inverse Transonic Airfoil Design Using Parallel Simulated Annealing and Computational Fluid Dynamics," *AIAA Journal*, Vol. 40, No. 4, 2002, pp. 791–794.

[11] Obayashi, S., Yamaguchi, Y., and Nakamura, T., "Multiobjective Genetic Algorithm for Multidisciplinary Design of Transonic Wing Planform," *Journal of Aircraft*, Vol. 34, No. 5, 1997, pp. 690–693.

[12] Jones, B. R., Crossly, W. A., and Lyrintzis, A. S., "Aerodynamic and Aeroacoustic Optimization of Airfoils via a Parallel Genetic Algorithm," *Journal of Aircraft*, Vol. 37, No. 6, 2000, pp. 1088–1096.

[13] Hacioglu, A., and Özkol, I., "Transonic Airfoil Design and Optimization by Using Vibrational Genetic Algorithm," *Aircraft Engineering and Aerospace Technology*, Vol. 75, No. 4, 2003, pp. 350–357.

[14] Rai, M. M., and Madavan, N. K., "Aerodynamic Design Using Neural Networks," *AIAA Journal*, Vol. 38, No. 1, 2000, pp. 173–182.

[15] Epstein, B., and Peigin, S., "Robust Hybrid Approach to Multiobjective Constrained Optimization in Aerodynamics," *AIAA Journal*, Vol. 42, No. 8, 2004, pp. 1572–1581.

[16] De Sousa, F. L., and Ramos, F. M., "New Stochastic Algorithm for Design Optimization," *AIAA Journal*, Vol. 41, No. 9, 2003, pp. 1808–1818.

[17] Ong, Y. S., Nair, P. B., and Keane, A. J., "Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling," *AIAA Journal*, Vol. 41, No. 4, 2003, pp. 687–696.

[18] Oksuz, O., Akmandor, S., and Kavsaoglu, M. S., "Aerodynamic Optimization of Turbomachinery Cascades Using Euler/Boundary Layer Coupled Genetic Algorithms," *Journal of Propulsion and Power*, Vol. 18, No. 3, 2002, pp. 652–657.

[19] Vicini, A., and Quagliarella, D., "Airfoil and Wing Design Through Hybrid Optimization Strategies," *AIAA Journal*, Vol. 37, No. 5, 1999, pp. 634–641.

[20] Obayashi, S., and Tsukahara, T., "Comparison of Optimization Algorithms for Aerodynamic Shape Design," *AIAA Journal*, Vol. 35, No. 8, 1997, pp. 1413–1415.

[21] Jones, B. R., Crossley, W. A., and Lyrintzis, A., "Aerodynamic and Aeroacoustic Optimization of Rotorcraft Airfoils via a Parallel Genetic Algorithm," *Journal of Aircraft*, Vol. 37, No. 6, 2000, pp. 1088–1096.

[22] Foster, N. F., and Dulikravich, G. S., "Three-Dimensional Aerodynamic Shape Optimization Using Genetic Algorithm and Gradient Search Algorithm," *Journal of Spacecraft and Rockets*, Vol. 34, No. 1, 1997, pp. 36–42.

[23] Vatandas, E., and Ozkol, I., "Dynamic Mesh and Heuristic Algorithms for the Design of a Transonic Wing," *Aircraft Engineering and Aerospace Technology*, Vol. 78, No. 1, 2006, pp. 39–44.

[24] Hajela, P., "Nongradient Methods in Multidisciplinary Design Optimization-Status and Potential," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 255–265.

[25] Jin, Y., Olhofer, M., and Sendhoff, B., "A Framework for Evolutionary Optimization with Approximate Fitness Function," *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 5, 2002, pp. 481–494.

[26] Hacioglu, A., "Augmented Genetic Algorithm with Neural Network and Implementation to Airfoil Design," AIAA Paper 2004-4633, 2004.

[27] Hacioglu, A., "A Novel Usage of Neural Network in Optimization and Implementation to the Internal Flow Systems," *Aircraft Engineering and Aerospace Technology*, Vol. 77, No. 5, 2005, pp. 369–375.

[28] Hacioglu, A., and Ozkol, I., "Inverse Airfoil Design by Using an Accelerated Genetic Algorithm via Distribution Strategies," *Inverse Problems in Science and Engineering*, Vol. 13, No. 6, 2005, pp. 563–579.

[29] Hacioglu, A., "Using Genetic Algorithm in Aerodynamic Design and Optimization," Ph.D. Thesis, Department of Aeronautical Engineering, Istanbul Technical University, Istanbul, Turkey, 2003.

[30] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "Learning Representations by Back Propagating Errors," *Nature (London)*, Vol. 323, Oct. 1986, pp. 533–536.

[31] Reed, R. D., and Marks, R. J., *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks*, MIT Press, Cambridge, MA, 1999.

[32] Ermis, M., Ulengin, F., and Hacioglu, A., "Vibrational Genetic Algorithm (VGA) for Solving Continuous Covering Location Problems," *Lecture Notes in Computer Science*, Vol. 2457, 2002, pp. 293–302.

[33] Hacioglu, A., and Ozkol, I., "Vibrational Genetic Algorithm as a New Concept in Aerodynamic Design," *Aircraft Engineering and Aerospace Technology*, Vol. 74, No. 3, 2002, pp. 228–236.

[34] Chung, T. J., *Computational Fluid Dynamics*, Cambridge Univ. Press, Cambridge, England, 2002.

[35] Kuethe, A. M., and Chow, C.-Y., *Foundation of Aerodynamics*, Wiley, New York, 1998, pp. 161–163.

[36] Hacioglu, A., "Interactive Solution Procedure for Full Potential and Boundary Layer Equations," M.Sc. Thesis, Department of Aeronautical Engineering, Middle East Technical University, Turkey, 1997.

[37] Murman, E. M., and Cole, J. D., "Calculation of Plane Steady Transonic Flows," *AIAA Journal*, Vol. 9, No. 1, 1971, pp. 114–121.

[38] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 347–372.

[39] Baker, J. E., "Reducing Bias and Inefficiency in the Selection Algorithm," *Proceedings of the Second International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, 1987, pp. 14–21.

[40] Eshelman, L. J., and Schaffer, J. D., "Real Coded Genetic Algorithms and Interval Schemata," *Foundations of Genetic Algorithms 2*, Morgan Kaufmann, San Mateo, CA, 1993, pp. 187–202.

B. Balachandran
Associate Editor