Single- and Multi-Objective Airfoil Design Using Genetic Algorithms and Artificial Intelligence

A.P. Giotis * K.C. Giannakoglou †
National Technical University of Athens, Greece

Abstract

Transonic airfoil design problems are solved using a Genetic Algorithm (GA) based optimizer. At the desired operating point, the minimum drag and constant lift targets are achieved through either a scalarized objective function, involving an arbitrary weighting factor, or the Pareto technique. For the optimization of an airfoil at two operating points, similar approaches are used. The CPU cost of the optimizer is kept low through Artificial Intelligence. A multilayer perceptron is trained using already evaluated individuals and provides good, though approximate, fitness predictions. With the regularly trained network, the direct flow solver calls are noticeably reduced.

1 Introduction

The search for optimum aerodynamic shapes is a major problem in aeronautics. The airfoil shape optimization requires a geometrical model fot its contour, which defines the set of control-parameters. A fitness or objective function is then defined, which should be driven to its minimum or maximum value. The minimization of drag is the usual objective in high-speed civil transport. This should be achieved with predefined lift and, often, by satisfying constraints relevant to the airfoil cross-section. Computational Fluid Dynamics (CFD) plays an important role since it is capable to enlight flow phenomena around complex shapes that are aerodynamically evaluated without resorting to costly experiments.

Numerical optimization, usually through iterative gradient-based methods, is a widely used tool. Rival to the aforesaid methods are stochastic optimizers and soft-computing techniques. Their major advantage is the capability to locate global optima. Among them, GAs, [1], are based on the concept of natural selection and their robustness has been proved in a variety of applications, including problems that can be hardly solved using gradient information. However, the shape optimization through standard GAs is time consuming due to the great number of CFD evaluations it involves. In the past, ways to reduce the CPU cost have been proposed by the second author. These were based on (a) the concurrent treatment of the individuals on parallel computing platforms [2] and/or (b) Artificial Intelligence [3]. In the second approach, artificial neural networks (ANNs) are first trained to correlate shapes and fitness scores and then used to evaluate new candidate solutions, without always resorting to CFD computations.

The present paper extends the method previously used for inverse design turbomachinery problems, to the airfoil shape optimization problem in aeronautics. The imposed

*Postgraduate Student, agiotis@central.ntua.gr
†Assistant Professor, kgianna@central.ntua.gr

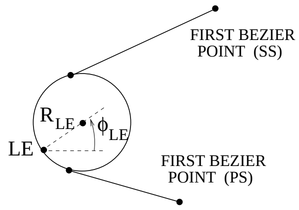

Figure 1: Parameterization near the LE

requirements (on the lift and the drag) have been used in both the single- and the multi-objective optimization fashion. Single- and multi-point designs at transonic flow conditions, have been worked out. For the multi-objective problem the Pareto front method has been programmed and its convergence speed was enhanced also through the ANNs.

2 Design through GAs

A GA starts by randomly generating a population of individuals, each of which stands for a candidate shape-solution with a binary or real number coding. Herein, the former was implied by concatenating the binary strings (bits) of the control-parameters. The population size N_{pop}, which does not alter during the genetic evolution, determines its exploration capabilities along with the computing cost per generation. About 40-60 individuals are practically used. These are all evaluated using CFD tools and their fitness scores are deduced by post-processing the numerical results. Using various selection criteria, reproductive trials are allocated to the individuals, with some bias toward the fittests. A mating pool is thus formed and this constitutes the first step for the evolution to the next generation, by applying the parent recombination operators. The recombination task gives rise to a new generation with possibly improved mean and best fitness scores. The evolutions stop if no improvement appears during the last few generations.

2.1 Profile Parameterization

The parameterization of the airfoil contour is carried through Bezier-Bernstein polynomials. The x- (optional) and y-coordinates of the Bezier points are the free-parameters controlled by the GA. Optionally, circular arcs at the leading (LE) and trailing (TE) edges can be superimposed to the polynomial curves. If the airfoil chord-length is kept fixed, Fig. 1, the use of circular arcs introduces the angular position of their centers (ϕ_{LE} and/or ϕ_{TE}) and their radii (R_{LE} and/or R_{TE}) as extra control parameters.

A geometrical model that is often used consists of two circular arcs (2+2 control parameters) and two Bezier curves with $N_{PS} = N_{SS} = 5$ points per airfoil side ($5 \cdot 2 + 5 \cdot 2 = 20$ control parameters, if both Cartesian coordinates are free). Thus, the total number of control parameters is $N_{FP} = 24$. Note that, at the junction points between Bezier curves and circular arcs, curve continuity and smoothness are automatically preserved.

2.2 Objective Function

Optimum or inverse design problems can be solved by the proposed method, provided that an appropriate objective or fitness function is defined. In the inverse design problems, the sought for airfoil should yield a desired pressure distribution over its contour at specified flow conditions. In this case, the objective is to minimize the deviation between actual and desired pressure distributions. Examples of inverse designs using different acceleration techniques can be found in [2] and [3]. In the optimum airfoil design problem, profiles giving maximum or known lift C_l and minimum drag C_d are sought for.

Herein, we will be dealing with the re-design problem; an initial airfoil is provided which generally defines the search space for the N_{FP} control-parameters. This profile will be re-designed aiming at the minimum C_d for the same $C_l = C_l^{target}$ (at the same flow conditions). Geometrical constraints will not be implied in the present studies. For a single-operating point, this can be carried out in two ways:

1. using a single objective function and the arbitrary factor b, as

$$\min F = \min\left\{(C_l - C_l^{target})^2 + b \cdot C_d\right\} \tag{1}$$

2. through a multi-objective optimization, by separately considering $\min(C_l - C_l^{target})^2$ and $\min(C_d)$ in order to provide the Pareto front, as discussed in the next section. Multi-point optimizations, at two flow conditions through the Pareto front technique will be also demonstrated.

The evaluation of the candidate shapes is undertaken by a primitive variable 2-D unstructured grid, flow solver [4]. The inviscid flow equations are solved using a vertex-centered finite-volume technique and a second order upwind scheme. The numerical solution is based on Jacobi-preconditioned GMRES techniques and requires about 40 sec on an Intel Pentium II 400MHz processor (for a typical grid of about 3000 nodes, convergence to 6 orders of magnitude; the cost increases to about 60 sec if grid adaptation is used).

Prior to the numerical solution, an unstructured grid is generated using an automated procedure. Its computing cost can be safely neglected.

2.3 Multi-objective optimization using the Pareto technique

In multi-objective optimization, a Pareto front [5] is the subset of candidate solutions obtained by eliminating any other solution for which an absolutely superior one (with respect to the ensemble of objective functions) can be found. According to the previous definition, the Pareto front consists of the so-called nondominated (or Pareto-optimal) solutions, in the sense that none of them is absolutely superior to any other constituent of the front. Thus, all of them are equally acceptable solutions to the problem and the choice of one of them requires a deep knowledge of the particular problem.

In order to obtain the Pareto front, a single-objective GA should be enriched by nondominated sorting and sharing. For each candidate solution in the current generation, scores for all of the objectives are first computed. The solutions are then ranked on the basis of nondomination among them and a number of fronts is thus formed. The first front is the Pareto-optimal front and its members are given a unit dummy cost function ($F_{dum} = 1$). The reproduction task is based on the F_{dum} values, after sharing is applied to each front separately. Sharing aims to spread the solutions all over the front by penalizing clustered ones and this is achieved by modifying the F_{dum} values. Starting from the Pareto-front, the sharing factor

$$m_i = \sum_{j=1}^{N_{pop}} Sh(d(i,j)) \tag{2}$$

is computed for its i^{th} member. The function $Sh(d)$ is defined as

$$Sh(d(i,j)) = \begin{cases} 0 & \text{if } d(i,j) \geq \sigma_{share} \\ 1 - \frac{d(i,j)}{\sigma_{share}} & \text{if } d(i,j) < \sigma_{share} \end{cases} \tag{3}$$

where $0 < \sigma_{share} < 1$, and i, j are in the same front. Their Euclidian distance $d(i,j)$ is computed either in the variables' or in the objectives' space as follows.

$$d(i,j) = \sqrt{\sum_{k=1}^{N_v} \left(\frac{v_k^i - v_k^j}{\max_{i=1,\ldots,N_f}(v_k^i) - \min_{i=1,\ldots,N_f}(v_k^i)} \right)^2} \quad (4)$$

where v stands for the variables or the objectives respectively, N_v the total number of variables (or objectives) and N_f the number of individuals belonging to the current front.

Since minimization problems are considered, the F_{dum} value of each solution is recomputed by multiplying it with m_i. On the next front, F_{dum} is first set equal to the worst score in the previous front plus a small quantity and then sharing applies, as previously.

In the genetic evolution, parent selection applies on the basis of the so-computed F_{dum} values.

3 Acceleration using Artificial Neural Networks

The overall design cost can be considerably reduced if some individuals are evaluated without resorting to the costly CFD model. For this purpose ANNs, [6], are used, provided that a training set, i.e. a sufficient number of profile shapes - fitness scores is available. Such availability is certain since the GA continuously evaluates new candidate solutions. An ANN should be trained on the input (shape) - output (fitness) mapping, through supervised learning. This entails additional CPU cost but, afterwards, the prediction of the fitness of any new shape is straightforward.

A multilayer perceptron (MLP), as the one used herein, is a feedforward ANN with its neurons arranged in layers [6]. All neurons in a layer are connected to all neurons in the adjacent layers using links associated with synaptic weights. A neuron in all but the first layer has many inputs but a single output. The synaptic weights act as signal multipliers on the corresponding links. In our case, as many neurons as the number of free parameters (N_{FP}) constitute the first layer. The output layer neurons are equal to the number of objectives and practically, 2-3 intermediate or hidden layers are used. The number of neurons in these layers should be as high as possible to increase the efficacy of the network, but as low as possible to reduce the training cost. The back error propagation (BEP) algorithm is used to compute the synaptic weights by repetitively presenting the network with the training set.

In general, the ANN guesses are not dependable. Nevertheless, they can be used to distinguish the promising solutions in the current generation which will be then accurately evaluated using the CFD tool. Our rule of thumb is to re-examine the top $\sim 40\% N_{pop}$ of individuals in each generation through the Euler solver. Considering that the network training costs as much as a direct flow solution, the overall computing cost is expected to reduce about 60%, thanks to the ANN implementation. When the ANN is used in conjunction with the Pareto method, all new individuals in the Pareto-optimal front need to be evaluated by the flow solver. If these are less than $40\% N_{pop}$ more individuals that belong to consecutive fronts are evaluated.

4 Analytical test case

A simple multi-objective problem, the simultaneous minimization of two analytical functions, is first solved and results with and without ANN acceleration are compared with the analytical solution. It should be stressed that, the computational cost of the analytical functions is negligible and the use of ANNs is only for demonstration purposes. In fact, the economy by no means outweighs the cost to train the ANN (built with one input for x and two outputs for O_1, O_2). The two objectives are $\min(O_1, O_2)$, where:

$$O_1 = \frac{1}{\sqrt{10-x}+\sqrt{x-5}}, \quad O_2 = 0.04(x-8)^2 + 0.3 \tag{5}$$

in the range of $x \in [5, 10]$. The two objectives along with the GA solutions, are plotted in fig. 2. The results with or without the use of ANN are identical. The Pareto front (fig. 3) is also the same, though the point distribution is different. Instead of using the Pareto technique, a linear combination of the two objectives is possible, $\min(O_1 + bO_2)$. Results of various single-objective optimizations are also included in the same figure for various b values. Figs. 4 and 5 present the increase of the Pareto front members and the total number of computations performed in each generation. During the evolution, dominated individuals are removed and new nondominated ones are inserted into the Pareto front. With the ANN acceleration, fewer individuals enter the optimal front in each generation. As only a percentage (40% N_{pop}) is accurately evaluated in each generation determining the maximum number of entries in the optimal front, this is reasonable side-effect. To get the same size of the Pareto front, more generations are needed. Nevertheless, the economy is still important, as one may see by summing up the accurate evaluations used.

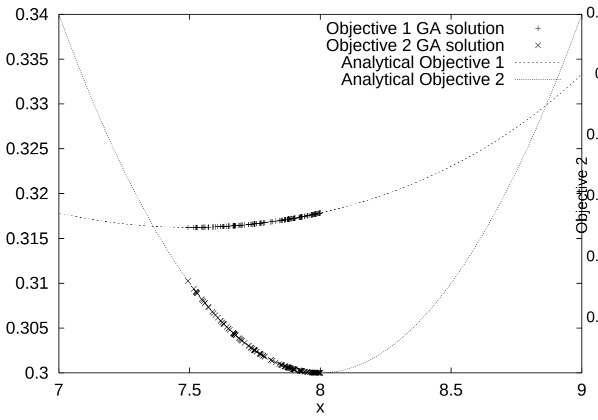

Figure 2: The O_1 and O_2 functions along with the optimum solutions.

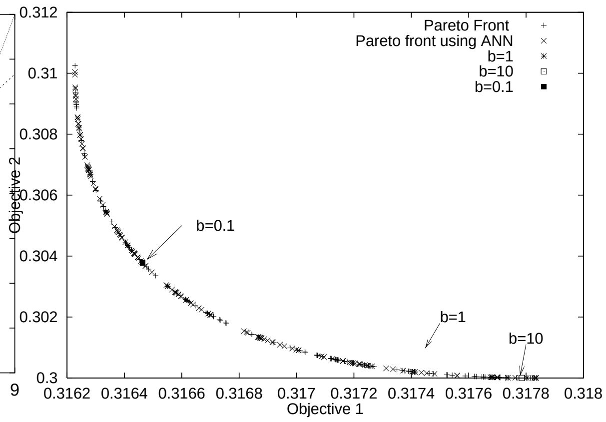

Figure 3: Pareto-optimal front.

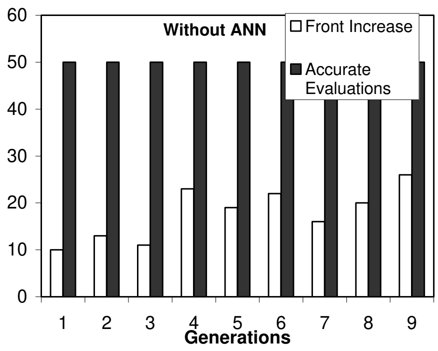

Figure 4: Stand-alone GA: Incremental entries in the optimal front and number of evaluations per generation.

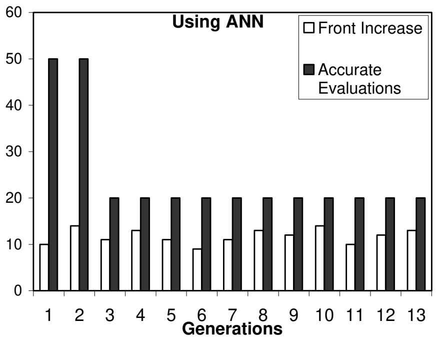

Figure 5: GA coupled with ANNs: Incremental entries in the optimal front and number of evaluations per generation.

5 Optimization of a transonic airfoil

In this section, the optimization of the RAE-2822 isolated profile, under inviscid flow conditions will be analyzed.

Single-point Designs: Two different problems have been analyzed. In the first, denoted by A, the flow conditions were $M_{inf} = 0.73$, $a = 2^o$ and the C_l^{target} was that of the standard (starting) profile $C_l^{target} = 0.780$. For point A, the following computations have been carried out: (a) six computations using the single-objective function 1 at different b values ($b = 1, 5, 10, 20$). Some of them used the stand-alone genetic optimizer and some other the ANN-based acceleration. The latter are marked by ANN, in the figures' captions. (b) two multi-objective optimizations, using the Pareto technique with and without ANNs.

The same airfoil was separately optimized at the B flow conditions ($M_{inf} = 0.75$, $a = 3.19^o$), where $C_l^{target} = 0.997$, as a single-objective optimization with $b = 1$ and as a multi-objective one with the Pareto technique. For point A, all the obtained results are shown in Fig. 6. Only a part of the Pareto front (the outer left one) has been drawn. The Pareto-optimal fronts with and without ANNs are quite close to each other, especially in the almost vertical part of this figure, where the C_l value is close to the target one. Each single-objective computation added a point in this figure, which was in general aligned with the Pareto front points. Table 1 shows the C_l and C_d values of the optimum profiles according to the single-objective optimizations.

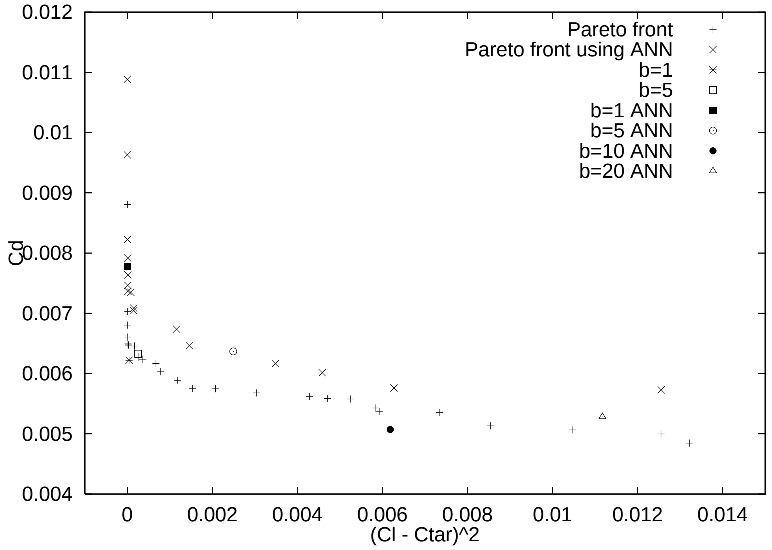

Figure 6: Operating point A, multi-objective optimization: The Pareto-optimal fronts along with optimum solutions from the single-objective studies.

Case	b=1	b=5	b=1 ANN	b=5 ANN	b=10 ANN	RAE2822
C_l	0.774	0.765	0.781	0.731	0.702	0.780
C_d	.00622	.00632	.00777	.00637	.00507	0.0180

Table 1: C_l and C_d of the re-designed airfoils (Operating point A, single-objective optimization)

Multi-point Design: For the multi-point design, two objective functions for each operating point, A and B have been devised, each of them using eq. 1. The case was studied without the ANN acceleration.

Fig. 7 illustrates the starting profile and the optimum airfoil shapes obtained using $b = 1, 5, 10$. It is worth noting that the x- and y- axes are in different scales. Fig. 8 presents the convergence of the single-objective GA in terms of the fittest individual score. A quite similar convergence with and without ANNs was obtained, though the costly stand-alone GA optimizer slightly outperforms its economic version (with the ANNs).

For the operating point B, similar plots to those shown for A can be drawn, but these will be omitted in the interest of space. The two-point design (minimization of eq.1, at both the A and B flow conditions) carried out through the Pareto technique is plotted in Fig. 9. The Pareto-optimal front contains five entries, which correspond to five different shapes. In the same figure, the computed optimum solutions for points A and B separately ($b = 1$ in all cases) are also included. For the optimum profile at point A, two solutions are shown, one with and the other without the ANN acceleration. In all these single-point optimization results the direct flow solver applied to the optimum geometry in order to calculate the flow field and the corresponding fitness score at the other operating point. As expected, the single-point optimum solutions are located at the top-left (for point A) and the bottom-right (for point B) parts of the figure. These should be conceived as the theoretical edges of the Pareto front.

6 Conclusions

The goal of this paper was to demonstrate the use of Genetic Algorithms in the airfoil optimization problem and to propose acceleration methods that increase their efficiency. The main conclusions of this work are listed below:

1. Multilayer perceptrons can reduce the CPU cost of design methods based on GAs. With a multilayer network that has been trained on the shape-fitness pairs resulting from the genetic evolution, a great part of the work volume, i.e. of the required CFD computations, is overcome. The CFD tool is only used to re-evaluate the fittest individuals on the basis of the ANN predictions.

2. The ANN can be also used with the Pareto technique, for the purpose of multi-objective optimization. In this method, a single network with as many outputs as the number of objectives has been used. An alternative way would be to train as many networks as the number of objectives, all of them with a single output neuron.

3. The Pareto technique along with an ANN yields fewer entries into the Pareto-optimal front per generation, compared to the stand-alone GA. Although more generations are required to reach a front of the same population, the total CPU cost is quite lower.

4. Through the ANNs, economy of the order of 60% can be achieved without damaging the quality of the obtained results.

References

[1] Goldberg D.E. *Genetic Algorithms in search, optimization & machine learning.* Addison-Wesley, 1989.

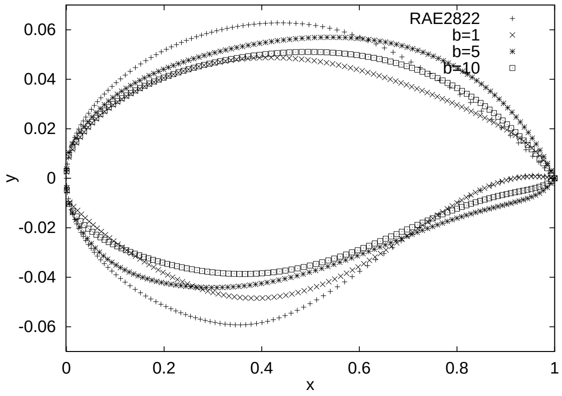

Figure 7: Optimum airfoil shapes for the operating point A (single-objective optimization).

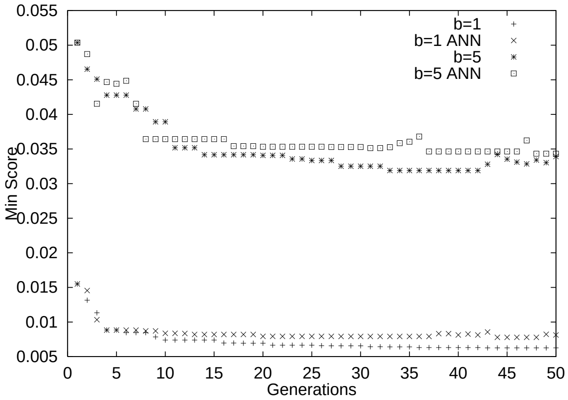

Figure 8: Convergence of the GA (single-point, single-objective). No elitism.

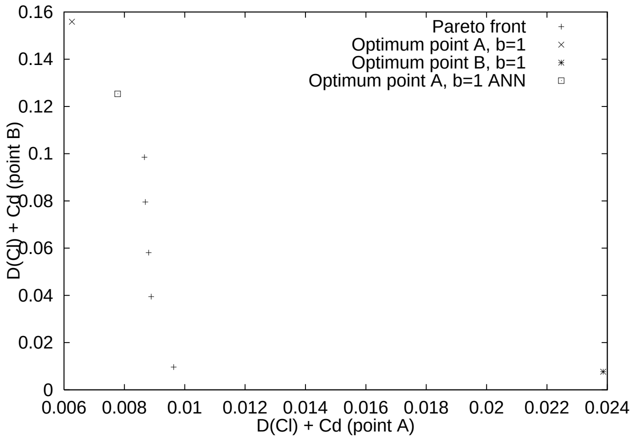

Figure 9: Two point design: the Pareto-optimal front.

[2] K.C. Giannakoglou. A design method for turbine blades using genetic algorithms on parallel computers. ECCOMAS 98, John Wiley & Sons, 1998.

[3] K.C. Giannakoglou. Designing turbomachinery blades using evolutionary methods. *To appear in ASME Turbo Expo 99, Indianapolis*, June 1999.

[4] D.G. Koubogiannis, L.C. Poussoulidis, D.V. Rovas, and K.C. Giannakoglou. Solution of flow problems using unstructured grids on distributed memory platforms. *Comp. Meth. Appl. Mech. Eng.*, 1998.

[5] J. Horn, N. Nafpliotis, and D.E. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. Proceedings of the First IEEE Conference on Evolutionary Computation, 82-87, Volume I, ICEC '94.

[6] A. Cichocki and R. Unbehauen. *Neural Networks for Optimization and Signal Processing.* John Wiley & Sons, 1993.