DESIGNING TURBOMACHINERY BLADES USING EVOLUTIONARY METHODS

K. C. Giannakoglou
Lab. of Thermal Turbomachines,
National Technical University of Athens,
P.O. Box 64069, 15710, Athens, Greece,
e-mail: kgianna@central.ntua.gr

ABSTRACT

A method based on computational intelligence is presented for the inverse design of 2D turbomachinery blades producing desirable wall pressure or velocity distributions under certain flow conditions. Blade airfoil shapes are parameterized through the combined use of circular arcs and Bezier functions. The design variables are controlled by Genetic Algorithms, employed in order to minimize the difference between target and numerically predicted distributions for each candidate shape. All numerical evaluations are carried through a primitive variable flow solver for unstructured grids. By employing Artificial Neural Networks, trained to correlate shapes and fitness scores, a great number of costly shape evaluations is overcome. The proposed combination of *GAs* for the optimization and *ANNs* for part of the evaluation phase is attractive and dramatically reduces the design cost. This paper discusses issues such as the regular re-training of the *ANN* or the selection of the threshold for numerical flow evaluations.

NOMENCLATURE

ANN	Artificial Neural Network
AVDR	Axial-Velocity-Density-Ratio
BEP	Back-Error Propagation
$\vec{C}$	Binary String
F	Cost or Fitness Function
g	Activation Function, $g(x) = (1 + e^{-x})^{-1}$
GA	Genetic Algorithm
L	Chromosome Length
LE	Leading-Edge
N_{FP}	Number of Free-Parameters
N_{HL}	*ANN*'s Hidden Layer Units
N_{pop}	Population Size
N_P	Number of Bezier Points on the *PS*
N_S	Number of Bezier Points on the *SS*
N_{TS}	Number of Training Patterns
PS	Pressure-Side
s	Arc-length along the Airfoil
SS	Suction-Side
TE	Trailing-Edge
$\vec{u}$	Design Variables' Array
w_{ji}	*ANN*'s Connection Weights (from i to j)

Greek Letters

ζ_k	*ANN*'s Output Layer Units
η	*ANN*'s Training Rate, $\eta = 1$
ξ_j	*ANN*'s Hidden Layer Units
σ	Threshold ($0 \leq \sigma \leq 1$)

Superscripts

targ	Target
pred	Predicted

INTRODUCTION

The design of efficient turbomachinery blades can be carried out on the basis of optimized distributions of flow quantities, such as pressure or velocity, along the blade walls. In what follows, this distribution will be refered to as $f^{targ}(s)$. Airfoil shapes are sought which, tested in the wind tunnel

or analyzed numerically, yield an $f(s)$ distribution as close as possible to $f^{targ}(s)$. From a mathematical point of view, the design is related to an optimization problem involving the minimization of

$$F(\vec{u}) = \int_{contour} |f^{pred}(\vec{u}) - f^{targ}| ds \qquad (1)$$

for each candidate blade. Needless to say that the designer's experience is implicit in the selection of a suitable f^{targ} distribution (Bouras et al. 1996).

Gradient-based methods have been widely used in numerical shape optimization. Quite often, gradient information is obtained through finite-difference quotients. They are computationally demanding, since as many flow computations as the number of design variables are required. In addition, inaccuracies may result if the finite-difference step is erroneously chosen. A possible way to reduce the *CPU* cost is through sensitivity analysis (Burgreen and Baysal, 1993). A different way to compute gradients is through the adjoint system of equations, (Jameson, 1988), which includes the linearization of the relation expressing the dependence of the fitness function on the control parameters.

Rival to the above are stochastic optimizers (Aly et al., 1996), their major advantage being the capability to seek global optima. Evolutionary methods, simulating annealing techniques and especially genetic algorithms (Quagliarella and Cioppa, 1995, Lee and Hajela, 1996, Galan et al., 1996, Goel et al., 1996, Rocchetto, 1996, Poloni, 1997, Giannakoglou, 1998) are the most known optimizers.

In this paper, soft computing techniques will be incorporated in the design method presented in Giannakoglou (1998). This will render it fast and effective without resorting to multiprocessing. The blade parameterization introduced in the aforementioned paper will be used. Geometric quantities constitute the set of design variables that are controlled by *GAs*, employed in order to minimize F. Cascades formed by the candidate blade shapes, corresponding to a set of design variables that are the outcome of the genetic evolution, are meshed with triangular elements and their fitness is evaluated through a primitive variable flow solver. During the genetic operations, a multi-layered network is trained. Its role is to guess the fitness of new candidate blade shapes, outside the training set. Based on the *ANN* guesses, "promising" shapes distinguish themselves among the members of each generation. These are the only shapes that need to be re-examined through the flow solver, in the hope of providing better solutions. Thus, the costly direct solver is used only for a small percentage of the *GA* population. To maximize the gain, some *ANN* parameters should be carefully adjusted. The regular adaptation of the design variables' search space, (Giannakoglou, 1998), that improves both the exploitation capabilities of the *GA* and the convergence rate, is also used.

For the assessment of this method, existing 2D turbomachinery blades will be first analyzed through a direct flow solver. The resulting pressure or velocity distributions constitute $f^{targ}(s)$ and the present method will be used to reconstruct the blades. By modifying the fitness function, the method is ready to accommodate other requirements or design constraints. It is necessary to stress here the point that what is our major concern is the acceleration of the design process itself.

BLADE SHAPE PARAMETERIZATION

The parameterization of the blade shape is of primary importance. Related to it is the total number of free-parameters that should be kept as low as possible for the sake of maximum computational efficiency. On the other hand, by allowing a wide search space for each design parameter, the model becomes flexible enough to cope with a variety of turbomachinery blade shapes.

A brief survey of the parameterization techniques used in relevant works will be presented first. Then, the proposed geometrical model will be described.

A Brief Literature Survey

In the relevant literature, Bezier functions, B-splines and other simpler polynomial expressions have been widely used due to their smoothing properties. Indicatively, in Goel et al. (1996), the parametric representation of a turbine aifoil was carried through Bezier-Bernstein polynomials. The Bezier control points close to *LE* and the *TE* were purposely alligned to insure continuity in the shape and its slope. In addition to the Bezier control points, other geometric parameters (like the stagger angle) were defined as control variables. In Li Jun et al. (1997), 4th-order parametric splines along segments of the *PS* and *SS* have been used for the design of transonic turbine cascades. Pritchard (1985) and Trigg et al. (1997) used similar geometric models based on 3rd-order polynomials (11 basic parameters) and cubic Bezier curves (17 basic parameters), respectively.

In Poloni et al. (1996), an assembly of Bezier curves was used, giving rise to 18 design parameters; four of them were used to model the camber line through a cubic Bezier curve while the others controlled the superimposed thickness distribution. Bezier functions have been also used in Galan et al. (1996) and Rocchetto (1996). In Quagliarella and Cioppa (1995), the airfoil was represented using a linear combination of a baseline shape along with some given modification functions.

The Proposed Parameterization

The proposed airfoil shape parameterization is based on the combined use of circular arcs and Bezier-Bernstein poly-

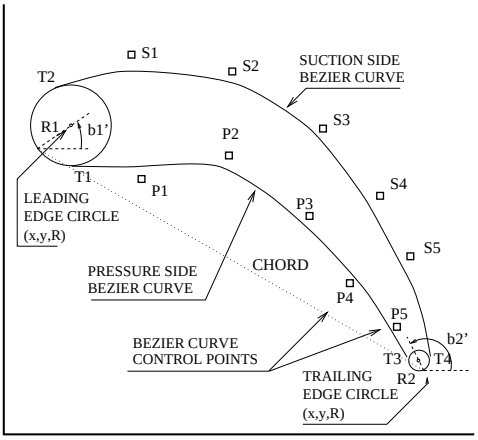

Figure 1: Airfoil parameterization.

nomials. This approach is attractive, for it may easily handle blades with rounded *LE* and/or *TE* like turbine blades. The use of circular arcs close to the front and rear part is optional and could be omitted if the targeted airfoil does not demand it. The two Bezier curves are defined by N_P+2 and N_S+2 control-points, along the *PS* and the *SS*respectively (Fig. 1) and cover their major parts.

For the purposes of this paper, major cascade characteristics (stagger angle, chord-length, pitch) will be considered known and fixed. This is not mandatory and when not observed one or more of the aforementioned quantities will be also included in the set of free parameters. According to the Bezier polynomial theory (Farin, 1988), the p^{th} derivative at each end-point is determined by the point itself and the p adjacent ones. With regard to this observation, the first derivative equals the slope of the straight line joining the end-point and the adjacent interior one. Consequently, drawing the tangent from the 2^{nd} or the $(N_P+1)^{th}$ (or $(N_S+1)^{th}$) Bezier points to the *LE* or *TE* circles respectively, the 1^{st} and the $(N_P+2)^{th}$ (or $(N_S+2)^{th}$) Bezier points can be located. On the basis of the premise above, circular arcs and Bezier curves remain continuous at the junction points.

According to the proposed parameterization, the design variables' array $\vec{u}$ consists of the radii R_1 and R_2 of the *LE* and *TE* circles, the b'_1 and b'_2 blade angles that determine the centers of the circles and the coordinates of N_P $(x_i^P, y_i^P, i=2, N_P+1)$ and N_S $(x_i^S, y_i^S, i=2, N_S+1)$ control-points. Therefore, the number of design parameters is found to equal $N_{FP} = 4 + 2N_P + 2N_S$. The set of N_{FP} real values that minimizes F is sought; this search will be undertaken by the *GA*.

Under certain conditions, the N_{FP} could be reduced. For instance, this can be done by fixing the x-coordinates of all Bezier control-points. In this case, only $4 + N_P + N_S$ design variables need to be used.

THE INVERSE DESIGN ALGORITHM

The inverse design problem is linked to the minimization of the cost function given in 1. In a previous paper by the same author (Giannakoglou, 1998), the inverse design was based on a standard *GA* combined with adaptive search domains for the design variables during the genetic search. Towards convergence, *GA* automatically switched to iterative hill-climbing optimization. A distributed memory parallel system supported the design process; processors were simultaneously associated with different cost function evaluations, in a SIMD fashion. Practically, the ellapsed time for evaluating a sufficient number of candidate solutions was equal to the duration of a single run, provided that as many processors as the examined solutions were available.

Optimization Through *GAs*

The *GA* handles a population of individuals each of which corresponds to a candidate blade shape. The components u_i, $i=1, N_{FP}$ of $\vec{u}$ are encoded as binary strings $\vec{C}_i$, each one with n_i bits. Their concatenation (head-to-tail) gives rise to the full binary string $\vec{C}$, encoding the candidate blade airfoil as a whole.

The binary substrings for the u_i, $i=1, N_{FP}$ parameters can be of variable length (i.e. different numbers n_i of bits could be used for each one of them). The chromosome length, i.e. the number of bits used to encode $\vec{u}$ is equal to

$$L = \sum_{i=1}^{N_{FP}} n_i$$

The designer should set bounds to the search space for the N_{FP} design parameters. The initially chosen lower $(u_{i,min})$ and upper $(u_{i,max},\ i=1, N_{FP})$ bounds may afterwards change in an adaptive manner.

GAs were developed to simulate processes observed in natural evolution. In the beginning, N_{pop} chromosomes are randomly selected over the search space of possible solutions. Each one of the initial generation chromosomes is given a fitness score which expresses the suitability of the corresponding individual. In a minimization problem, the genetic evolution looks for the chromosome with the minimum possible score. Maximization problems are handled in an analogous way. In each generation, the individuals compete to participate in the mating procedure that creates new offspring. The competition is based on their relative scores. In the present work, multiple reproductive trials are allocated to the fitter individuals through linear fitness ranking, whereas some of the weaker chromosomes die out. Since these trials define the number of copies of each individual in the mating

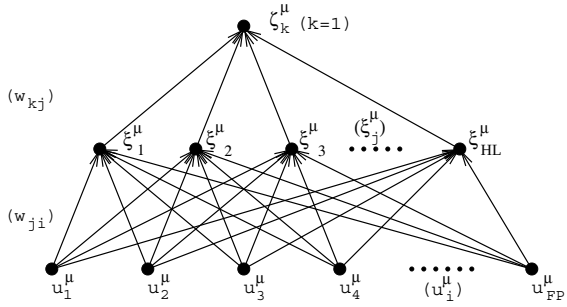

Figure 2: A three-layer *ANN*.

pool, fitter individuals are likely to receive more than one copies. Once the mating pool is filled in with copies of the previous generation chromosomes, crossover and mutation undertake to form the next generation. In crossover, pairs of individuals are selected from this pool at random and their chromosomes are cut at a point also selected at random. Parts before and after the cuts are mutually exhanged (one-point crossover). Crossover is carried out with very high (about 90%) probability. Finally, mutation is applied to the individuals formed through crossover. In mutation, bits in the chromosome string may randomly alter according to a small probability (usually less than 1%).

Fitness Evaluation Using *ANNs*

ANNs were inspired by neurophysiology and gave rise new non-algorithmic computing methodologies. A detailed discussion of *ANNs* is provided in classical textbooks (Dayhoff, 1990, Hertz et al. 1992), so it suffices to present here the most important features of the *BEP* networks, as incorporated in the shape evaluation procedure.

A *BEP* network is capable of learning complex input-output mapping rules, like the mapping between $\vec{u}$ and $F(\vec{u})$. For this purpose, the network should be presented with input patterns (airfoil shapes that correspond to already examined chromosomes) paired with the target outputs (fitness scores computed using a direct flow solver). At the completion of the training, the network can evaluate new airfoil shapes, by guessing output values for input patterns other than those used in the training process.

Fig. 2 presents a typical multilevel feed-forward network, organized in three layers. It includes only one hidden layer with N_{HL} units, fully connected to the input and output layers. The input layer consists of N_{FP} units (as many as the design variables) and the corresponding values will be referred to as $\vec{u} = u_i$, $i = 1, N_{FP}$. The output layer has only one processing unit associated with the fitness score and will be denoted by ζ_k, $k = 1$. In what follows, values at the hidden layer units will be denoted by $\vec{\xi} = \xi_j, j = 1, N_{HL}$. Each connection is associated with its own weight. According to fig. 2, weights w_{ji}, $j = 1, N_{HL}, i = 1, N_{FP}$ and w_{kj}, $k = 1, j = 1, N_{HL}$ should be adjusted during training.

In the *BEP* method, N_{TS} training patterns are repetitively presented to the network. Upon each presentation (denoted by $\mu = 1, N_{TS}$), the processing units perform weighted sums of their inputs and, through a non-linear activation function g, their outputs are computed. Propagating the signals forwards yields

$$\xi^{\mu}_{j} = g(w_{ji}u^{\mu}_{i}) \quad , \quad \zeta^{\mu}_{k} = g(w_{kj}\xi^{\mu}_{j}) \tag{2}$$

where the summation convention applies when repeated indices appear.

Through the forward propagation of each input signal $\vec{u}^{\mu}$, a single value at the output unit $\zeta^{\mu}_{k=1}$ is obtained. In general, this is different than the desired fitness score $F^{\mu} = F(\vec{u}^{\mu})$ and this deviation is a measure of the system error. The backward propagation step starts from the output and moves backward through the successive hidden layers, changing the connection weights. The correction is incremental and is carried through the computation of an additive vector $\delta\vec{w}$. According to the gradient descent algorithm, weights are modified in the direction in which the system error is decreased. For the network shown in fig. 2, a routine calculation yields

$$\delta w_{kj} = \eta\chi^{\mu}_{k}\xi^{\mu}_{j} \quad , \quad \chi^{\mu}_{k} = (F^{\mu} - \zeta^{\mu}_{k})g'(w_{kj}g(\xi^{\mu}_{j})) \tag{3}$$

$$\delta w_{ji} = \eta\psi^{\mu}_{j}u^{\mu}_{i} \quad , \quad \psi^{\mu}_{j} = \chi^{\mu}_{k}w_{kj}g'(\xi^{\mu}_{j})) \tag{4}$$

The training task entails a number of iterations and computational cost. On the contrary, the use of an already trained network for the evaluation of a candidate shape outside the training set is straightforward and with negligible cost. In the next Section, the utilization of the network will be explained in detail.

The Proposed Inverse Design Algorithm

In the proposed method, an *ANN*, which learns how to map sets of design variables to fitness scores, helps overcome a considerable amount of airfoil cascade evaluations through the costly flow solver. Compared to the algorithm analyzed in Giannakoglou (1998), the gain in *CPU* time is noticeable, at least as long as both design methods run on a sequential computer. Each time the optimizer does not resort to the flow solver for a new shape evaluation, the computing gain is equal to the cost of running the solver. The training cost of the network should be considered as equivalent to the cost of a single flow solution and does not present any threat to the efficiency of the method.

Without considering multi-processing, Fig. 3 illustrates the main loop of the proposed algorithm. At the end of the first generation of the *GA* and regularly afterwards the

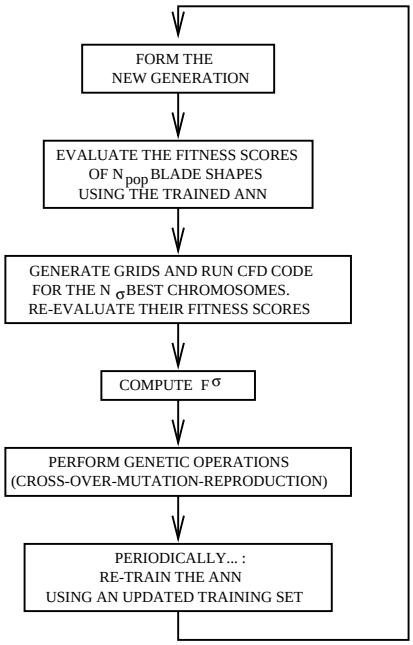

Figure 3: The iterative part of the design algorithm.

network is trained. Chromosomes examined by the direct flow solver paired with the so-obtained fitness scores are presented to the network input. The connection weights are computed and stored. Practicalities about how often the network should be (re-)trained and comments on the training patterns that should be used are given below.

During all but the first *GA* generations, each chromosome is first evaluated using the trained network. The design parameters corresponding to this chromosome are presented to the input of the network, which estimates its fitness score.

Although a properly trained network is dependable, the network cannot estimate fitness scores lower than the minimum score of the training set chromosomes. Consequently, the network is not capable to generate a fitter solution than the fittest in the training set. On the other hand, even if this were possible, guesses close to the 'extreme' training data would not be as reliable.

To overcome this problem, the N_{pop} chromosomes of any generation, are sorted according to the *ANN* estimates and the $(\sigma\ N_{pop})^{th}$ best score is found. Let us denote this score by F^{σ}. In the next generation, any chromosome for which the *ANN* provides fitness score lower than F^{σ} needs to be re-examined by the direct solver. One would therefore expect a reduced number of $N_{\sigma}(\ll N_{pop})$ flow calculations to be conducted. A fitter chromosome will be hopefully found among the N_{σ} re-examined chromosomes. The parameter σ is chosen by the designer. Low σ values reduce the number of required numerical flow solutions, but better chromosomes than the current best will be hardly captured if the network estimate is way out of the accurate. On the other hand, higher σ values tend to erode the gain in *CPU* time. As F^{σ} is computed using the scores obtained in the preceding generation the number of flow solutions per generation is not constant.

THE USE OF *ANN*: PRACTICALITIES

(a) The network is presented with u_i, $i = 1, N_{FP}$ real values rather than L binary digits, since $N_{FP} \ll L$.

(b) In the examined cases, $N_{FP} = 24$ input units and a single output unit have being used. One hidden layer with about $N_{HL} = 20$ processing units proved to be sufficient for the network to be both reliable and fast to learn. The selected network size is as close to the optimum as possible, for this particular kind of problems. By increasing the number of hidden layers and/or the number of units in these layers, the training cost increases. On the other hand, a very small number of units in the hidden layer usually leads to poor learning and the map between input and output patterns is not reliable. So, the designer should carefully select the network parameters.

(c) The connection weights are updated by re-training the network using new training data originating from the evolving populations of the *GA*. An automatic decision mechanism for determining when and how re-training should occur has been incorporated. Upon completion of the genetic operations, re-training starts if:

1. a chromosome fitter than the fittest in the training set appears, or
2. during the last m (usually about 3 to 6) generations, new chromosomes have been analyzed with the direct code, giving rise to new paired inputs-outputs that might enrich the training set.

(d) Various re-training procedures are available. They are usually based on simple linearizations (Park et al., 1991), and are efficient for slowly varying non-stationary processes. In the present problem, a much simpler approach has been adopted. Each time the network weights are to be re-adjusted due to the enrichment of the available paired shapes and fitness-scores, the network is presented with the entire training material. In the gradient descent algorithm, the connection weights w_{ji} and w_{kj} start evolving from their more recent values.

(e) Often, some candidate airfoils with unrealisic shapes are not capable to yield a converged flow field around them. In general, more delicate is the design of turbine

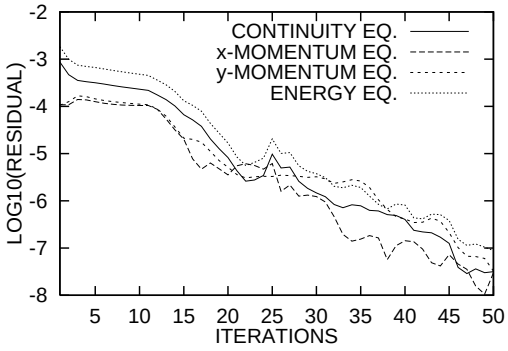

Figure 4: Typical convergence history of the direct flow solver, using restarted *GMRES*.

blades, since at choked or near-choked conditions the ratio $p_{inlet}/p_{tot,inlet}$ and the size of the cascade throat are physically related.

THE DIRECT FLOW SOLVER

The direct flow solver (Koubogiannis et al., 1998) is based on the finite-volume technique, as it applies to unstructured grids with triangular elements. It supports quasi-3D flow analyses in cascades, with variable streamtube thickness along the axial direction, to account for *AVDR* values other than unit. A vertex-centered scheme, where each control volume is formed around a grid node is used. The convective terms are discretized using the Roe (1986) scheme with MUSCL extrapolation. The numerical solution is performed using the restarting, matrix-free *GMRES* technique (Saad and Schultz, 1983), preconditioned through the Jacobi scheme. For typical meshes of about 6000 triangles - 3000 nodes, 50-60 external iterations are required for the residual to converge about 5-6 orders of magnitude. Such a typical convergence is shown in Fig. 4; this was obtained in the first examined case and corresponds to the fittest blade shape. It has been computed using a *CFL* number linearly increasing from 1 to 80 (within the first 50 iterations).

The unstructured grid is generated through an automatic triangulation software, based on the advancing front technique. The designer specifies the number of points (about $500 - 800$) along the airfoil contour. This number, though high for inviscid calculations, helps to cope with unrealistic shapes that may appear. A few intermediate layers with a gradually decreasing number of nodes are automatically created around the airfoil, in order to keep the grid size at reasonable levels. The external boundary is formed by straight lines, with a coarse nodal point distribution along them. The grid, including any node on the inner, the outer and the intermediate layers is generated at almost negligible cost.

APPLICATION OF THE METHOD

The proposed method has been used for the inverse design of a turbine and a compressor blade. For two existing blades, the aforementioned flow solver was used to compute velocity or pressure distributions along the blade contours. Using them as target, the proposed design method was then applied to reconstruct the original shapes. As stressed in the Introduction, emphasis was given to demonstrate the method's efficiency and the gain due to the incorporation of *ANNs*.

The first problem examined was the inverse design of a high turning turbine nozzle. The reference blade geometry is described in Sieverding (1990). As the available point distribution was not sufficiently smooth, we deemed that smoothing through high-order polynomials was necessary. Of course, this incurred inaccuracies and the target (reference) blade was slightly different than the original. The chord length ($40cm$), the pitch-to-chord ratio (0.78) and the stagger angle (58.8^o) remained fixed during the design process.

In direct mode, the inviscid flow in this cascade has been calculated at inlet flow angle equal to 1.9^o and isentropic exit Mach number equal to 0.974. The so-obtained velocity distribution was used as target. The comparison of target and computed distributions is always based on the arc-length along the airfoil contour, even if these were often plotted as $f(x)$ distributions. The target distribution is given below, together with the best obtained solution.

First, we will illustrate the convergence characteristics of the proposed method, in this particular case. In Fig. 5, the convergence rates of four variants of this method are compared. Referring to the legend of Fig. 5, *METHOD1* stands for the standard *GA* optimizer operating on the basis of fixed search space (adaptation was excluded) and without the acceleration through the *ANNs*. In *METHOD1*, each chromosome is evaluated using the Euler solver. Thus, the computing cost of this variant is proportional to the number of candidate shapes. In *METHOD2*, the search space automatically re-adapts its bounds by shrinking both sides of the corresponding rectangles around the current best solution. This takes place as the method evolves, provided that during the last four generations there is no improvement. *METHOD3* combines *GAs* with *ANNs* in order to decrease the number of *CFD* code runs, with fixed search spaces. Finally, *METHOD4* is the enhanced version making use of both search space adaptation and *ANNs*.

Interesting conclusions can be gleaned by comparing the behaviour of these variants. Using the same population ($N_{pop} = 50$) and the same genetic operations, the comparison is fair and meaningful. Similar conclusions have been also drawn for the design of the compressor blade. According to Fig. 5, the use of *ANNs* without the adap-

tive scheme (*METHOD3*) leads to worse solutions as only a small percentage of the candidate solutions is accurately evaluated. In this test case, *METHOD2* gives slightly better results than any other variant. Note that *METHOD1* and *METHOD4* produced results of similar quality, very close to that of *METHOD2*, whereas *METHOD3* and *METHOD4* were dramatically faster.

In order to quantify the gain in computing time, we will assume that the cost of genetic operations is negligible. Therefore, the cost of the design will be measured in terms of 'direct flow solutions'. This term includes the unstructured grid generation and the numerical solver. For the enhanced version (*METHOD4*), Fig. 6 compares the number of required direct flow solutions per generation for two σ values, namely $\sigma = 0.10$ (its convergence was illustrated in Fig. 5, for *METHOD4*) and $\sigma = 0.20$ (with slightly better convergence characteristics). Without considering any convergence criterion, all comparisons have been made on the basis of 40 generations, with $N_{pop} = 50$. Thus, 2000 flow solutions have been required in *METHOD1*. Instead, only 535 or 709 runs have been required in *METHOD4*, with $\sigma = 0.10$ or $\sigma = 0.20$ respectively.

In order to demonstrate the quality of the inverse design, the target velocity distribution along the blade walls and the best predicted one are given in Fig. 7. They are in very good agreement, showing a shock springing from the blade *TE* and meeting the *SS* of the adjacent blade. For this design, *METHOD4* was used with $\sigma = 0.10$ and $N_{PS} = N_{SS} = 5$. Linked to the previous figure is Fig. 8, comparing the reference and the best predicted blade shapes. Their comparison is also very satisfactory. The final locations of the Bezier points, are also shown in this figure. The rectangular areas define the search spaces for the Bezier control points, during the first generations. As stated above, close to convergence, the search spaces become very small and centered in relation to the current best solutions.

The second problem examined was that of a controlled-diffusion airfoil cascade (Steinert et al., 1990), representative of industrial axial flow compressors. The inlet flow angle and the isentropic exit Mach number were equal to 47^o and 0.62, respectively. The *AVDR* was equal to 0.9, so that the streamtube thickness before the *LE* or after the *TE* was constant and equal to 1.0 and 0.9 respectively, whereas in the part between the *LE* and *TE* it varied linearly.

The direct flow solver provided the target pressure coefficient distribution for the reconstruction of the blade. In Fig. 9, the target pressure coefficient distribution is shown. In the same figure, this quantity is also compared with the predicted blade shape having the best fitness score. Between the two curves, only small differences can be seen, close to the leading edge or along the rear part of the *SS*. The blade shapes which correspond to the aforesaid distri-

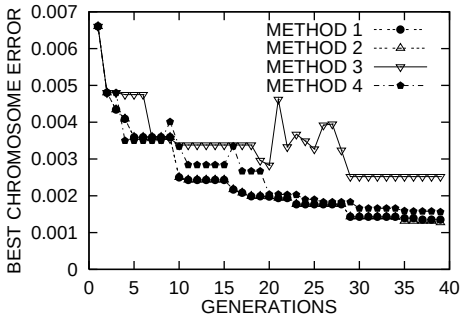

Figure 5: *GA* convergence for the turbine blade design.

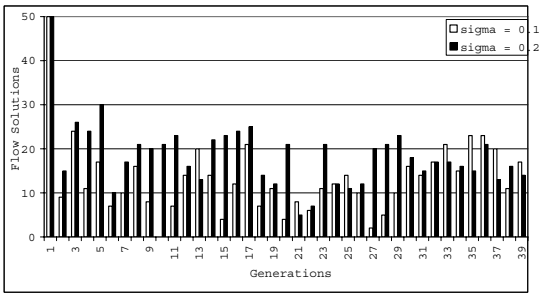

Figure 6: Number of required direct flow solutions per generation ($N_{pop} = 50$), for $\sigma = 0.10$ and $\sigma = 0.20$.

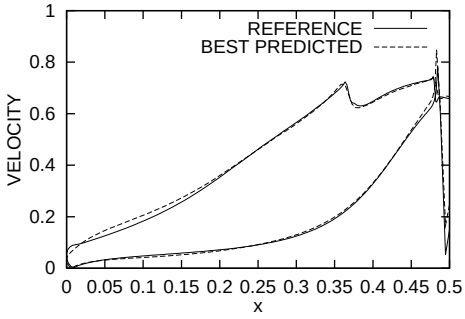

Figure 7: Target and best computed velocity distributions along the blade walls.

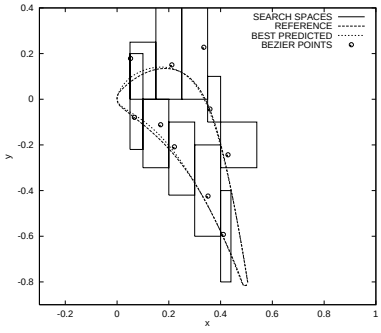

Figure 8: Reference and best predicted turbine blade, search spaces and the optimum locations of the Bezier points.

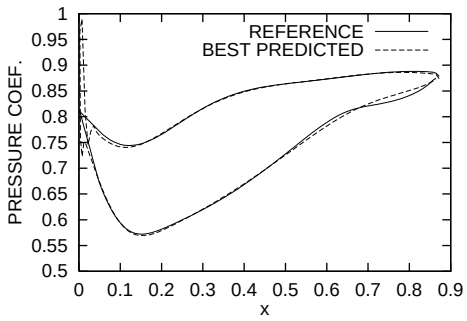

Figure 9: Target and best computed pressure coefficient distributions along the compressor blade walls.

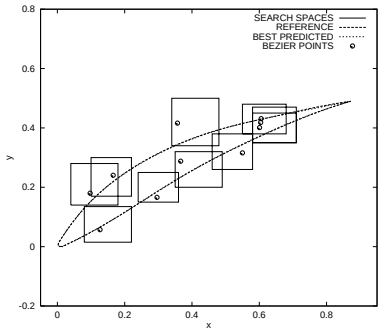

Figure 10: Reference and best predicted compressor blade, search spaces and optimum locations of the Bezier points.

butions are shown in Fig. 10. The search spaces for the Bezier points are indicated in the same figure.

We will repeat here the comparative study of the four variants of the proposed method, as we did with regard to the turbine blade. The convergence of the four versions of the method is shown in Fig. 11. The nomenclature for this figure is the one previously used in Fig. 5. *METHOD1* should be considered as the standard procedure. Compared to it, *METHOD3*, that is the use of the *ANN* accelerator without adaptivity of the search space, proved to be very slow and the convergence stagnates within the first 10

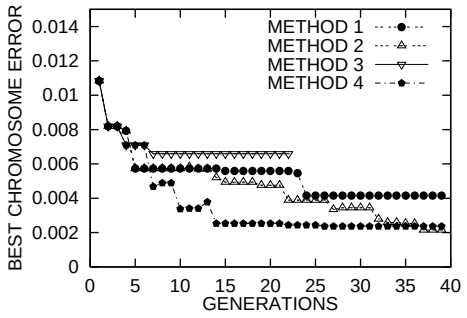

Figure 11: Comparison between different implementations of the method.

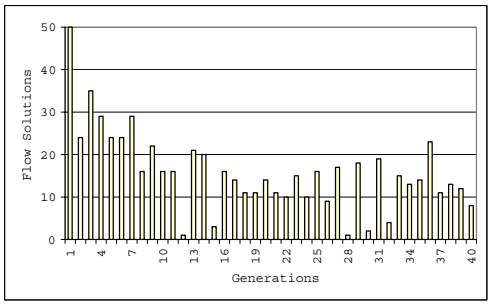

Figure 12: Number of required direct flow solutions per generation ($N_{pop} = 50$).

generations. The convergence of *METHOD2* is better than that of *METHOD1*. *METHOD4*, with a quite low σ value ($\sigma = 0.2$) gave the faster convergence, reaching almost the same final error as *METHOD1* but in a reduced number of iterations, as shown in Fig. 12. *METHOD4* demands only about 30% of the total number of direct flow solutions required by *METHOD1*.

CONCLUSIONS

Computational intelligence was succesfully incorporated to the inverse design of 2D turbomachinery blades, at high-subsonic and transonic flow conditions. The aim was to lower, as much as possible, the computing cost. Calculations were performed for the reconstruction of compressor and turbine blades, based on target wall pressure or velocity distributions. Of course, other inverse design or optimization problems can be handled; it merely suffices to define appropriately the fitness function. All of the designs have been carried out using an Euler solver for unstructured grids. A Navier-Stokes solver or a boundary layer method could be used as well.

The purpose of this paper was to illustrate the capability of *ANNs* to considerably accelerate design techniques based on *GAs*. The major conclusions are:

(a) The proposed 2D airfoil parameterization, based on the use of circular arcs and Bezier functions is adequate for typical compressor or turbine blades.

(b) *ANNs* can be trained to correlate shapes and fitness scores, replacing thus a great number of costly shape evaluations. The proposed combination of *GAs* for the optimization and *ANNs* for part of the evaluation phase reduces the design cost by a factor of about 3 to 4.

(c) *ANNs* should be necessarily combined with automatic search space adaptation. It is clear, though not discussed in this paper, that a careful timing for the repetitive adaptations is required, to avoid premature convergence of the *GA*.

(d) The efficiency of the *ANN*-based accelerator depends on the so-called parameter σ. Low σ values reduce the number of the required numerical flow solutions, but a small number of better chromosomes might be never captured. On the other hand, high σ values tend to damage the efficiency of the method.

(e) The connection weights of the *ANN* should be updated through regular re-training of the network, based on updated input-output pairs. An automatic decision mechanism to determine when and how re-training should occur, has been proposed.

REFERENCES

Aly, S., Ogot, M., Pelz, R., "Stochastic Approach to Optimal Aerodynamic Shape Design", AIAA J. Aircraft, 33(5), pp. 956-961, 1996.

Bouras, B., Karagiannis, F., Leoutsakos, G., Giannakoglou, K.C., Papailiou, K.D., "Arbitrary Blade Section Design Based on Viscous Considerations. Background Information", ASME J. of Fluids Engineering", 118, pp. 358-363, 1996.

Burgreen, G.W., Baysal, O., "Aerodynamic Shape Optimization Using Preconditioned Conjugate Gradient Methods", AIAA-93-3322-CP, 1993.

Dayhoff, J.E., "Neural Network Architectures. An Introduction", Van Nostrand Reinhold, New York, 1990.

Farin, G., "Curves and Surfaces for Computer Aided Geometric Design. A Practical Guide", Academic Press, Inc., San Diego, CA, 1988.

Galan, M., Winter, G., Montero, G., Greiner, D., Periaux, J., Mantel, B., "A Transonic Flow Shape Optimization Using Genetic Algorithms", ECCOMAS 96, John Wiley & Sons, 1996.

Giannakoglou, C.K., "A Design Method for Turbine Blades Using Genetic Algorithms on Parallel Computers", ECCOMAS 98, John Wiley & Sons, 1998.

Goel, S., Cofer, J.I., Singh, H., "Turbine Airfoil Design Optimization", ASME Paper 96-GT-158, 1996.

Hertz, J., Krogh, A., Palmer R.G., "Introduction to the Theory of Neural Computation", Addison-Wesley, 1992.

Jameson, A., "Aerodynamic Design via Control Theory", J. of Scientific Computation, 3, pp. 33-260, 1988.

Koubogiannis, D.G., Poussoulidis, L.C., Rovas, D.V., Giannakoglou, K.C.", "Solution of Flow Problems Using Unstructured Grids on Distributed Memory Platforms", Comp. Meth. Appl. Mech. Eng., 1998.

Lee, J., Hajela, P., "Parallel Genetic Algorithm Implementation in Multidisciplinary Rotor Blade Design", AIAA J. Aircraft, 33(5), 962-969, 1996.

Li Jun, Feng Zheping, Chang Jianzhong, Shen Zuda, "Aerodynamic Optimum Design of Transonic Turbine Cascades Using Genetic Algorithms", J. of Thermal Science, 6(2), pp.111-116, 1997.

Park, D.C., El-Sharkawi, M.A., Marks R.J., "An Adaptively Trained Neural Network", IEEE Transactions on Neural Networks, 2, pp. 334-344, 1991.

Poloni, C., Mosetti, G., Contessi, S, "Multi Oblective Optimisation by GAs: Application to System and Component Design", ECCOMAS 96, John Wiley & Sons, 1996.

Poloni, C., Peridora, V., "*GA* Coupled with Computationally Expensive Simulations: Tools to Improve Efficiency", Genetic Algorithms in Engineering and Computer Science, John Wiley & Sons Ltd., 1997.

Pritchard, L.J., "An Eleven Parameter Axial Turbine Aerofoil Geometry Model", ASME Paper 85-GT-219, 1985.

Quagliarella, D., Cioppa, A.D., "Genetic Algorithms Applied to the Aerodynamic Design of Transonic Airfoils", J. Aircraft, 32(4), pp. 889-891, 1995.

Rocchetto, A., "Transonic Airfoil Design via Numerical Optimization", ECCOMAS 96, John Wiley & Sons, 1996.

Roe, P.L., "Discrete Models for the Numerical Analysis of Time-Dependent Multidimensional Gas Dynamics", J. of Computational Physics, 63, pp. 458-476, 1983.

Saad, Y., Schultz, M.H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems", Research Report YALEU/DCS/RR-254, 1983.

Sieverding, C.H., "Test Case E/CA-8: Transonic Turbine Cascade", AGARD-AR-275, 1990.

Steinert, W., Eisenberg, B., Starken, H., "Design and Testing of a Controlled Diffusion Airfoil Cascade for Industrial Axial Flow Compressor Application", ASME Paper 90-GT-140, 1990.

Trigg, M.A., Tubby, G.R., Sheard, A.G., "Automatic Genetic Optimization Approach to 2D Blade Profile Design for Steam Turbines", ASME Paper 97-GT-392, 1997.