INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2004; **45**:1339–1362 (DOI: 10.1002/fld.747)

Robust handling of non-linear constraints for GA optimization of aerodynamic shapes

S. Peigin[1,*,†] and B. Epstein[2,‡]

[1]*Israel Aircraft Industries, Engineering Division, Ben-Gurion Airport, 70100, Israel*
[2]*The Academic College of Tel-Aviv Yaffo, Computer Science Department, Israel*

SUMMARY

A new approach to the robust handling of non-linear constraints for GAs (genetic algorithms) optimization is proposed. A specific feature of the approach consists of the change in the conventional search strategy by employing search paths which pass through both feasible and infeasible points (contrary to the traditional approach where only feasible points may be included in a path). The method (driven by full Navier–Stokes computations) was applied to the problem of multiobjective optimization of aerodynamic shapes subject to various geometrical and aerodynamic constraints. The results demonstrated that the method retains high robustness of conventional GAs while keeping CFD computational volume to an acceptable level, which allowed the algorithm to be used in a demanding engineering environment.

KEY WORDS: constrained optimization; genetic algorithms; aerodynamic design; Navier–Stokes solutions

1. INTRODUCTION

In the last decade, optimal aerodynamic shape design has aroused considerable interest. In particular, robust software for multiobjective aerodynamic optimization through drag minimization is in high demand worldwide.

The first optimization method in aerodynamics was that of Lighthill [1]. Presently, advanced optimization tools are mostly based on the fully deterministic gradient approach [2] or, alternatively, on the probabilistic evolutionary algorithms [3, 4].

Many modern gradient-based methods originate from the work of Pironneau [5] who created a framework for the formulation of elliptic design problems. The present state-of-the-art of this class of methods can be found in References [6, 7], where the gradients are computed from the solutions to the flow equations and its adjoint equations. The current status of non-gradient optimization methods is given in References [8, 9].

*Correspondence to: S. Peigin, Israel Aircraft Industries, Engineering Division, Ben-Gurion Airport, 70100, Israel.
†E-mail: speigin@iai.co.il
‡Consultant to IAI.

Published online 15 June 2004
Received 22 August 2003
Revised 5 April 2004

A vast majority of optimization problems are constrained problems. Unfortunately, the solution of the optimization problems with non-linear constraints still remains stiff and open from a theoretical as well as an application point of view.

The presence of constraints significantly decreases the performance and computational efficiency of the classical conjugate-gradients (CG) optimization methods. To explain why this happens, we should take into account that the constraints divide the search space $\mathbf{F}$ into two sub-domains: a feasible region $\mathbf{F}_f$ and an infeasible one, $\mathbf{F}_n$. Topologies of the feasible and infeasible sub-domains can be rather complex. The search strategy of the traditional CG-type optimization methods is based on the evaluation of fitness in the neighbourhood of a point where the objective function is presumably smooth, and on the determination of the local derivatives of the objective function at the above point.

If the point is located close to the constraints boundary, the radius of this neighbourhood can become very small. In such cases the calculation of the derivatives of the objective function with appropriate accuracy is an extremely difficult problem. An additional difficulty which considerably decreases the computational efficiency of CG-type methods is the fact that the determination of such a neighbourhood is not an easy task. It results in the expenditure of large amounts of computing time for unsuccessful evaluations of the cost function at the points which are located in the infeasible sub-domain, because the exact position of the constraints boundary is not known *a priori*.

The classical approach to resolve this problem for CG methods is a concept of penalty functions which penalize infeasible solutions. These methods differ in many important details of how the penalty function is designed and applied to infeasible solutions. Let us only note that this approach is not robust enough, being strongly dependent on a specific problem, and there are no general guidelines on designing penalty functions.

As it is stressed in Reference [7], the penalty parameters are difficult to choose in practice, because the theory requires that they tend to infinity while the conditioning of the problem deteriorates when they are large. In fact, the application of the penalty approach is some kind of art, because for real success the user should have good intuition about the behaviour of the objective function near the constraints boundary. In addition, this method further aggravates the computational efficiency of the CG-type algorithms.

The design problem may be characterized by a mix of continuous, discrete and integer design variables, and the resulting design space can be non-convex or even disjointed. For all these reasons, optimizations methods which do not rely on the computation of gradients, in particular evolutionary programming and GAs, became highly popular [3, 8, 9].

Unfortunately, in their basic form GAs are not capable of handling constraint functions limiting the set of feasible solutions. Therefore additional methods are needed to keep the solutions in the feasible region [33, 34].

During the last few years several categories of methods (both direct and indirect) were proposed for handling constraints by GAs for parameter optimization problems. The classification of these methods is presented in survey [10].

One of these categories contains the methods which are based on preserving feasibility of the solutions [11]. The idea is to use specialized genetic operators which transform feasible individuals into feasible individuals, that is, operators, which are closed on the feasible part $\mathbf{F}_f$ of the search space. This approach assumes linear constraints only and a feasible initial population.

Another interesting class of methods employs the idea of decoders. In these techniques, a chromosome 'gives instructions' on how to build a feasible solution [12]. However, the use of decoders for continuous domains has not been investigated.

There are a few methods which emphasize the distinction between feasible and infeasible solutions in the search space **F**. For example, the method proposed in Reference [13] (called a behavioural memory approach) considers the constraints in a sequence: a switch from one constraint to another is made where a sufficient number of individuals, feasible with respect to this constraint, are attained in the population.

Many evolutionary algorithms incorporate a constraint-handling method based on the concept of exterior penalty functions, which penalize infeasible solutions [14–20]. Similar to CG methods, there are no general guidelines on designing penalty functions in the GAs framework. Moreover, penalty function approach in general is known to often modify the actual objective function space so much that the resulting optimum point can be unacceptably far from the true optimum location.

Usually, the penalty function is based on the distance of a solution from the feasible region $\mathbf{F}_f$, or on the effort to 'repair' the solution, that is to force it into $\mathbf{F}_f$. For example, a method of static penalties was proposed in Reference [17]; it assumes that for every constraint we establish a family of intervals, which determine appropriate penalty coefficients. The method of dynamic penalties was suggested in Reference [18]. A method based on adaptive penalty functions was developed in Reference [16], where both the search length and constraint severity feedback were incorporated. It involves the estimation of a near-feasible threshold for each constraint. Such thresholds indicate 'reasonable' distances from the feasible region $\mathbf{F}_f$.

An alternative to traditional penalty methods is the segregated genetic algorithm, which was developed in Reference [20]. This is yet another way to handle the problem of robustness on the penalty level. Here two different penalized fitness functions with static penalty terms (smaller and larger, respectively) are designed.

Nevertheless, the overview of the subject shows that unfortunately, the existing approaches are mainly problem-dependent or that they are restricted to certain types of fitness functions. In particular, it is not clear what is the influence of characteristics of a constrained problem (such as the number of linear and non-linear constraints, the type of the objective function, the ratio of areas of feasible search space to the whole, etc.) on the performance of the algorithm.

We can conclude that although there has been extensive research on the constraints handling methods for GAs, this issue is still not well understood and still remains an open problem to be solved.

In this paper we suggest a new approach to robust handling of non-linear constraints in the framework of GAs. The approach is applicable to the optimization problems where the constraints boundary is not known in advance and the optimal point may be located on the boundary itself. Basically the approach can be outlined as follows:

(1) Change of the conventional search strategy by employing search paths which pass through both feasible and infeasible points (instead of the traditional approach where only feasible points may be included in a path). Since in our case a topology of feasible and infeasible sub-domains is rather complex, the realization of such a strategy should significantly improve the accuracy and efficiency of optimization. The idea is that the information from the infeasible sub-domains can be very important for the optimization,

as a path to the optimal point via infeasible points can be essentially shorter (or even the only possible, in the case of a non-simply-connected feasible domain).

(2) To implement the new strategy, it is suggested to extend the search space. This requires the evaluation (in terms of fitness) of the points, which do not satisfy the constraints imposed by the optimization problem. A needed extension of an objective function may be easily implemented by means of GAs due to their basic property: contrary to classical optimization methods, GAs are not confined to only smooth extensions.

In fact, such extension should satisfy only the condition that follows immediately from the main idea to increase diversity of the current population: in infeasible regions, the objective function should be defined in such a way that it keeps in the current population a sufficient number of infeasible individuals, which are located rather close to the constraints boundary. In such a case we can expect, with a rather high probability, that the crossover between feasible and infeasible individuals will produce high-fitness children.

It is important to emphasize that the proposed approach, contrary to penalty function methods, (1) determines the value of fitness everywhere including the infeasible region $\mathbf{F}_n$; (2) does not change the value of the objective function in the whole feasible region $\mathbf{F}_f$ and (3) does not require the smoothness of the fitness function across the constraints boundary.

An additional weakness of GAs lies in their poor computational efficiency, which prevents their practical use where the evaluation of the cost function is too computationally heavy. To overcome this obstacle, we introduce an intermediate computational tool which is based on reduced-order models (ROM) approach.

This tool approximates the objective function using a very limited number of exact CFD computations while providing a fast and reasonably accurate computational feedback in the framework of GAs search. In order to ensure the accuracy and robustness of the method, a multidomain prediction-verification principle is used. On the prediction stage, the genetic optimum search is concurrently performed on a number of search domains, and then, the whole set of corresponding optima is verified through full Navier–Stokes computations. Additionally, in order to ensure the global character of the search, the algorithm is iterated.

The above methodology was applied to multipoint constrained design of aerodynamic airfoils. Note, that the aerodynamic shapes optimization problem is an exemplary case of problems where the non-linear constraints strongly affect the solution. In such problems, the constraints boundary is not known in advance, and moreover, the optimal point is frequently located on this boundary.

The approach employs GAs as an optimization tool in combination with a ROM method based on local data bases obtained by full Navier–Stokes computations. The important features of the approach also include: (1) scanning of the optimization search space by a combination of full Navier–Stokes computations with the ROM method (2) multilevel parallelization of the whole computational framework which efficiently makes use of computational power supplied by massively parallel processors (MPP).

The results demonstrated that the method combines efficient handling of various nonlinear constraints with high accuracy of optimization. The method retains high robustness of conventional GAs while keeping CFD computational volume to an acceptable level due to a limited use of full Navier–Stokes computations. A significant computational time-saving allowed to introduce the method into engineering environment.

The paper has the following structure: The problem description is given in Section 2. In Section 3 a novel strategy for the handling of nonlinear constraints imposed on the optimization problem in the framework of GAs, is suggested. In Section 4 a new approach to improve computational efficiency of GAs is proposed. The implementation of the optimization algorithm is described in Section 5. Numerical results of optimization are given and analysed in Section 6.

2. PROBLEM STATEMENT

In this section a multipoint optimization of 2D airfoils is considered. In the case of the single point optimization problem, the objective is to minimize the cost function $Q = C_D$ (total drag coefficient) of a two-dimensional profile subject to the following classes of constraints:

(1) Aerodynamic constraints such as prescribed constant total lift coefficient C_L^*.
(2) Geometrical constraints on the shape of the airfoil surface: relative thickness of the profile t/c, radius of leading edge R_L, trailing edge angle θ, shape 'freeze' of certain portions of airfoil (such as lower or upper surfaces, trailing (leading) edge region, etc.):

$$t/c \geqslant (t/c)^*, \quad R_L \geqslant R_L^*, \quad \theta_T \geqslant \theta_T^* \tag{1}$$

where values $(t/c)^*$, θ_T^* and R_L^* are prescribed parameters of the problem.

The single-point design airfoil must be analysed over a range of Mach numbers and lift coefficients to ensure the adequacy of the off-design performance. To reach this goal multipoint optimization is needed where the objective function is a weighted combination of single point cost functions:

$$Q = \sum_{i=1}^{P} w_i \cdot C_{D_i}, \quad \sum_{i=1}^{P} w_i = 1 \tag{2}$$

where the coefficients w_i are non-negative and P is the number of the optimization points.

As a gas-dynamic model for calculating C_D and C_L values, the full Navier–Stokes equations are used. Numerical solution of the full Navier–Stokes equations was based on the code NES [21, 22].

The NES code is based on the essentially non-oscillatory (ENO) concept [23] with a flux interpolation technique [24]. The important advantage of the solver NES as a driver of the optimization process is its ability to supply reliable and sufficiently accurate results already on relatively coarse meshes, and thus to reduce dramatically the volume of CFD computations.

3. TREATMENT OF CONSTRAINTS IN THE FRAMEWORK OF GAs

GAs became highly popular as optimization methods in the last two decades [25]. They are hybrid semi-stochastic/deterministic optimization methods that are conveniently presented using the metaphor of natural evolution. The GA algorithms are based on the evaluation of a set of solutions, called population. The population is treated with genetic operators.

As a basic algorithm, a variant of the floating-point GA is used. We apply the tournament selection, which enables us to increase the diversity of the parents. Three types of crossover operator have been employed: single point, uniform and arithmetical crossover. As the mutation operator we applied the non-uniform mutation defined by Michalewicz [25]. To resolve one of the main problems that arises in GAs—a premature convergence—we used distance-dependent mutation [26]. This approach consists of using the distance between two mates in order to compute the mutation rate. This means that the mutation rate is no longer a constant parameter; it is dynamically computed for both children and depends on the parents. Every time a couple of individuals is chosen for mating, the mutation rate is computed in order to be applied to their children after crossover occurs. If the parents are quite close (that is, their mating is likely to be considered as 'incest'), that would lead to a high mutation rate for their children, whereas mutation rate will be smaller if they are different. To improve the convergence of the algorithm we also use the elitism principle [27].

The optimization method resulted in the following pseudo-code:

$t = 0$
initpopulation $P(t)$ / random or random + initial solutions /
while not converged do
$P^*(t)$:= *selectparents* $P(t)$ / tournament selection /
recombine $P^*(t)$ / single point, uniform or arithmetical crossover /
mutate $P^*(t)$ / non-uniform + distance-dependent mutation /
evaluate $P^*(t) : P(t+1) := P^*(t) + best(P)$ / elitism /
$t := t + 1$
enddo

To explain our constraint-handling approach let us assume that we have to find the minimum of the objective function $f(x)$ in the feasible region $\mathbf{F}_f$, defined by non-linear constraint $g(x) \leqslant 0$. Let us also consider the most challenging case when the optimal point x_o is located close to the constraints boundary of the feasible region $g(x) = 0$ or even lies on the boundary which is not exactly known in advance, before evaluation of the objective function.

In order to create the robust and computationally efficient GAs for solution of this type of problems we propose to change the search strategy. Instead of the traditional strategy where the optimal point is reached through steps along feasible points only, we suggest moving to the optimal point x_o via both feasible as well as infeasible points. If the topology of feasible and infeasible sub-domains is rather complex, the realization of such a strategy could significantly improve both the accuracy and the efficiency of the optimization method. Additionally, this approach allows to significantly extend the applicability range of a method. The key point is based on the following idea: the information from the infeasible sub-domain can be very important, because sometimes the path to the optimal point via infeasible points can be shorter than when utilizing feasible points only (or even the only possible, in the case of a non-simply-connected feasible sub-domain).

To utilize the new strategy we should extend our search space and by doing this, evaluate in terms of fitness the individuals that do not satisfy the constraints on the optimization problem. It is very important to note that the needed extension of fitness function may be rather rough and non-smooth, because contrary to CG-type optimization methods, the GA methods are not

restricted by the smoothness of this extension. In fact, such an extension should only satisfy the condition which immediately follows from the main idea of increasing a diversity of the current population. Namely, the objective function in infeasible region should be defined in such a manner that keeps in the current population a certain number of infeasible individuals, which are located rather close to the constraints boundary. In such a situation we can expect with rather high probability, that the crossover between feasible individuals and the infeasible ones will produce high-fitness children.

Based on these ideas, the following two-step approach to the extension of the objective function into infeasible region is proposed. A starting point is to estimate roughly the order of the objective function value over the feasible sub-domain. It can be done using the preliminary information on the behaviour of the objective function, or it can be based on testing a number of feasible points.

Let us assume that the initial estimate has the form: $f(x) \approx A \ (x \in \mathbf{F}_f)$. Based on this estimation we define the following first-step approximation for the modified objective function $f^*(x)$:

$$f^*(x) = \begin{cases} f(x) & \text{if } g(x) \leqslant 0 \\ B & \text{if } g(x) > 0 \end{cases} \tag{3}$$

where $B \gg A$.

The next step in the approach is to run the above described GA for the solution of the optimization problem (in fact a non-constrained problem) with the first-step modified objective function $f^*(x)$ and to estimate the value of the objective function for feasible individuals in the neighbourhood of the constraints boundary. Using the results of these calculations, the first-step estimation is corrected: $f(x) \approx C$. Then the modified objective function $f^{**}(x)$ for the total search space $\mathbf{F}$ is finally defined as follows:

$$f^{**}(x) = \begin{cases} f(x) & \text{if } g(x) \leqslant 0 \\ \alpha_1 C + \alpha_2 g(x) B & \text{if } g(x) > 0 \end{cases} \tag{4}$$

where $\alpha_1 \geqslant 1.0$ and $\alpha_2 > 0$ are problem-dependent real numbers. Note that the second term in the right-hand side of (4) is proportional to $g(x)$ and its influence is small in the vicinity of the constraints boundary $g(x) = 0$.

It must be emphasized that the estimations of B and C may be rather rough and in fact, only the order of magnitude of the objective function in the vicinity of the constraints boundary is required.

Specifically in this work, the proposed strategy was implemented in the following way. For approximation of the upper and lower airfoil surface, a Bezier spline representation was used. A Bezier curve of order N is defined by the Bernstein polynomials $B_{N,i}$ and it is completely determined by the Cartesian co-ordinates of the control points $\vec{P}_i^k$

$$\vec{G}^k(t) = \sum_{i=0}^{N} B_{N,i} \vec{P}_i^k, \quad B_{N,i} = C_N^i t^i (1-t)^{N-i}, \quad C_N^i = \frac{N!}{i!(N-i)!} \tag{5}$$

where t denotes the parameter of the curve taking values in $[0,1]$, superscript $k=u,l$ corresponds to upper and lower surfaces of profile, $i=0,\ldots,N$.

We also fix the position of leading and trailing edges and all the abscisses of the control points. Finally, based on the relations which ensure the smoothness of the airfoil at the leading edge, a search string $S=(a_1,a_2,\ldots,a_{N-1},a_N,\ldots,a_{2N-5})$ has the following form:

$$S=\begin{cases} a_i=y_i^u, & 1\leqslant i\leqslant (N-1) \\ a_i=y_{i-N+2}^l, & N\leqslant i\leqslant (2N-5) \end{cases}$$

Thus a string S contains $2N-5$ values (ordinates of control points). These values vary within the search domain D. The domain D is determined by Min_i and Max_i values, which are the lower and upper bounds of the variable a_i.

Based on the above approach to the handling of constraints, the modified objective function Q for the solution of drag minimization problem was defined as follows:

$$Q=\begin{cases} 0.1+[(t/c)^*-(t/c)] & \text{if } (t/c)<(t/c)^* \\ 0.2+[R_{\mathrm{L}}^*-R_{\mathrm{L}}] & \text{if } R_{\mathrm{L}}<R_{\mathrm{L}}^* \\ 0.3+[\theta_{\mathrm{T}}^*-\theta_{\mathrm{T}}] & \text{if } \theta_{\mathrm{T}}<\theta_{\mathrm{T}}^* \\ 0.5 & \text{if } y^u(t)<y^l(t) \\ C_{\mathrm{D}} & \text{otherwise} \end{cases} \tag{6}$$

The choice of constants in Equation (6) was fairly straightforward, based on the following principle. Their values should be at least several times greater than the upper bound of C_{D} (about 0.0500 for the considered class of problems), which ensures that, for any feasible point, the value of the objective function will be low in comparison with that of any infeasible point. On the other hand, these constants should not be too high, in order to ensure that a sufficient number of infeasible points will be present in the population.

In the case of multipoint optimization the value of Q represents a weighted combination of the partial modified objective functions Q_i corresponding to the flight points participating in optimization:

$$Q=\sum_{i=1}^{P} w_i\cdot Q_i \tag{7}$$

It must be stressed that, for all the considered test cases, (including the multipoint optimization) the constants appearing in relation (6) remained unchanged.

4. IMPROVEMENT OF COMPUTATIONAL EFFICIENCY OF THE ALGORITHM

Alongside the difficulties associated with handling of constraints, an additional weakness of GAs lies in their poor computational efficiency. This prevents their practical use in the case where the evaluation of the cost function is computationally expensive as it happens in the framework of the full Navier–Stokes model even in the two-dimensional case.

For example, an algorithm with the population size $M = 100$ requires (for the case of 200 generations) at least 20 000 evaluations of the cost function (CFD solutions). A fast full Navier–Stokes evaluation takes at least a couple of minutes of CPU time. That means that one step of such an algorithm takes about 650 h, which is practically unacceptable.

In order to overcome this, it is proposed to introduce an intermediate 'computational agent'—a computational tool which, on the one hand, is based on a very limited number of exact evaluations of objective function and, on the other hand provides a fast and reasonably accurate computational feedback in the framework of GAs search.

With this end in view, we propose to construct a computational agent by means of a ROM approach. Among the ROM models (in the broad sense of the word) we may mention the use of simpler gas-dynamic models (see e.g. Reference [28]), representation of the solution of gas-dynamic problem in terms of its eigenmodes [29] or representation of the aerodynamic system using the Volterra theory of non-linear systems [30].

In this work we use ROM approach in the form of local approximation method (LAM). With the ROM–LAM method, the solution functionals which determine a cost function (such as lift and drag coefficients in the case of drag minimization), are approximated by a local data base. The data base is obtained by solving the full Navier–Stokes equations in a discrete neighbourhood of a basic point positioned in the search space.

So on the one hand, the number of exact estimations of the objective function (full Navier–Stokes solutions) is proportional to the dimension of the search space. On the other hand, the computational volume required to provide approximate estimates of the objective function in the framework of GAs optimum search, is negligible.

Thus the above mentioned requirements of the computational agent, related to its computational efficiency, are fulfilled. However, due to the approximate nature of the approach, an additional effort should be made in order to ensure the accuracy and robustness of the method.

To reach this goal a multidomain prediction–verification principle is employed. That is, on the prediction stage the genetic optimum search is concurrently performed on a number of search domains. As the result each domain produces an optimal point, and the whole set of these points is verified (through full Navier–Stokes computations) on the verification stage of the method, and thus the final optimal point is determined.

Besides, in order to ensure the global character of the search, it is necessary to overcome the local nature of the above approximation. For this purpose it is proposed to perform iterations in such a way that in each iteration, the result of optimization serves as the initial point for the next iteration step (further referred to as optimization step).

The specific algorithm is described below in the case of single point drag minimization.

Denote $x = (a_1^n, a_2^n, \ldots, a_{2N-5}^n, \alpha^n)$ point in the search space, where a_i^n and α^n are the Bezier coefficients of an initial profile at nth optimization step and the angle of attack, corresponding to the prescribed C_L^*, respectively. Then each airfoil can be determined by deviations δ_i^n from the coefficients of the initial profile. At fixed values of other flow parameters, the solution functionals depend on the values of δ_i^n and δ_α^n (a deviation from the initial angle of attack). In the optimization process the following local approximation of a functional F^n is used (subscript n is omitted and $F = C_L, C_D$):

$$F(a_1 + \delta_1, \ldots, a_{2N-5} + \delta_{2N-5}, \alpha + \delta_\alpha) = F^\circ + \sum_{j=1}^{2N-5} \Delta F_j + \Delta F_\alpha$$

$$\Delta F_j = \begin{cases} \Delta F_j^+ & \text{if } (F_j^+ - F^\circ)(F_j^- - F^\circ) \leqslant 0,\ \delta_j \geqslant 0 \\ \Delta F_j^- & \text{if } (F_j^+ - F^\circ)(F_j^- - F^\circ) \leqslant 0,\ \delta_j < 0 \\ \Delta F_j^{+-} & \text{if } (F_j^+ - F^\circ)(F_j^- - F^\circ) > 0 \end{cases}$$

$$\Delta F_j^+ = \frac{\delta_j}{\Delta_j}(F_j^+ - F^\circ), \quad \Delta F_j^- = -\frac{\delta_j}{\Delta_j}(F_j^- - F^\circ), \quad \Delta F_\alpha = \frac{\delta_\alpha}{\Delta_\alpha}(F_\alpha - F^\circ) \tag{8}$$

$$\Delta F_j^{+-} = \frac{\delta_j(\delta_j + \Delta_j)}{2\Delta_j^2}F_j^+ - \frac{\delta_j^2}{\Delta_j^2}F^\circ + \frac{\delta_j(\delta_j - \Delta_j)}{2\Delta_j^2}F_j^-$$

In approximation (8) the following notation is used:

$$F^\circ = F(a_1, \ldots, a_{2N-5}, \alpha), \quad F_\alpha = F(a_1, \ldots, a_{2N-5}, \alpha + \Delta_\alpha),$$

$$F_j^+ = F(a_1, \ldots, a_{j-1}, a_j + \Delta_j, a_{j+1}, \ldots, a_{2N-5}, \alpha), \quad (j = 1, \ldots, 2N-5)$$

$$F_j^- = F(a_1, \ldots, a_{j-1}, a_j - \Delta_j, a_{j+1}, \ldots, a_{2N-5}, \alpha), \quad (j = 1, \ldots, 2N-5)$$

Here the local data base values F°, F_j^+, F_j^- and F_α are obtained by solving the full Navier–Stokes equations at the corresponding neighbouring points of the basic point in the search space. These neighbouring points are determined by positive variations $\{\Delta_j\}$ corresponding to the Bezier coefficients $\{a_j\}$ and by the variation Δ_α of the angle of attack α.

In fact, relation (8) represents a mixed linear-quadratic approximation in the neighbourhood of the basic point x^o which employs the local data-base. One-dimensionally, we use either a one-sided linear approximation (in the case of monotonic behaviour of the functional F) or a quadratic approximation (otherwise).

Such a reduced second order approximation A_r^2 neglects mixed second derivatives compared to the full second order approximation of objective function A_f^2 in the vicinity of the basic point x^o. However, the reduced approximation retains a number of properties of the full approximation which are crucial with optimization in view.

In particular, a simple algebraic analysis shows that if a full quadratic form A_f^2 possesses a local extremum, then the reduced form A_r^2 also possesses the corresponding extremum at some point x_e. Moreover, it is ensured that the value of A_f^2 at the point x_e is nearer to the extremum of the full form than the value F° of the objective function at the basic point. Note that the stronger is the extremum, the higher is the improvement.

Values of C_L and C_D calculated via relations (8) were systematically compared with results of numerical solution of the full Navier–Stokes equations for a wide range of feasible values of Δ_j. It was concluded that formula (8) possesses acceptable accuracy at least in the following range of deviations δ_j and δ_α:

$$-2\Delta_\alpha \leqslant \delta_\alpha \leqslant 2\Delta_\alpha, \quad -2\Delta_j \leqslant \delta_j \leqslant 2\Delta_j, \quad (j = 1, \ldots, 2N-5)$$

Using (8) we can obtain the following formula for cost function C_D at prescribed lift coefficient C_L^* (for current point at the search space $(a_1+\delta_1, \ldots, a_{2N-5}+\delta_{2N-5})$) which is based

on the local data base $C_{\mathrm{D}}^{\mathrm{o}}$, C_{D_α}, $C_{\mathrm{D}_j}^{+}$, $C_{\mathrm{D}_j}^{-}$, $C_{\mathrm{L}}^{\mathrm{o}}$, C_{L_α}, $C_{\mathrm{L}_j}^{+}$, $C_{\mathrm{L}_j}^{-}$:

$$C_{\mathrm{D}} = C_{\mathrm{D}}^{\mathrm{o}} + \sum_{j=1}^{2N-5} \Delta C_{\mathrm{D}_j} + \frac{\delta_\alpha}{\Delta_\alpha}(C_{\mathrm{D}_\alpha} - C_{\mathrm{D}}^{\mathrm{o}})$$
$$\delta_\alpha = \frac{\Delta_\alpha}{C_{\mathrm{L}_\alpha} - C_{\mathrm{L}}^{\mathrm{o}}} \left[C_{\mathrm{L}}^{*} - C_{\mathrm{L}}^{\mathrm{o}} - \sum_{j=1}^{2N-5} \Delta C_{\mathrm{L}_j} \right] \tag{9}$$

5. IMPLEMENTATION OF THE ALGORITHM

The general sketch of the optimization algorithm can be presented by the following pseudo-code:

```
opt_step = 0
Determine_Initial_Basic_Point   /starting basic point—initial profile/
while not converged do
Calc_Local_Data_Base            /CFD computations in a discrete neighbourhood
                                 of the basic point/
Search_Optim_Candidates         /Hybrid GA-ROM search of optimal points for various
                                 search domains/
Verification_Optim_Cand         /CFD computations for optimal points/
Choose_New_Basic_Point          /Choose a new basic point—the best one
                                 among all the points in the global CFD data base/
opt_step := opt_step + 1
enddo
```

In what follows a step by step description of the pseudo-code is given.

Step Determine_Initial_Basic_Point: The step deals with determination of Bezier coefficients of the initial profile (determination of the initial basic point in the search space).

Step Calc_Local_Data_Base: At this step the CFD local data base is obtained by solving the full Navier–Stokes equations at the neighbouring points of the basic point in the search space. For given values of variations Δ_j and Δ_α, the values of C_{L} and C_{D} are attained at the points, corresponding to these variations.

Step Search_Optim_Candidates: Here, GA is applied for various search domains D_k (corresponding to different search scales), and optimal points O_k for each domain are obtained ($k = 1, \ldots, N_{\mathrm{D}}$; N_{D} is the number of the search domains). Fast estimation of objective function in the framework of GAs optimum search, is performed using local approximation of cost function (9).

Step Verification_Optim_Cand: At this step the full Navier–Stokes solver is applied to each optimal point O_k, and the corresponding data are added to the global CFD data base.

Step Choose_New_Basic_Point: The last step of the iterative optimization loop. A new basic point is determined as the best point in the global CFD data base.

Roughly speaking, each CFD run requires a different geometry and therefore, the construction of a new computational grid. To maintain the continuity of optimization stream, the topological similarity of geometrical configurations is used, and the grids are built by means of a fast automatic transformation of the initial grid.

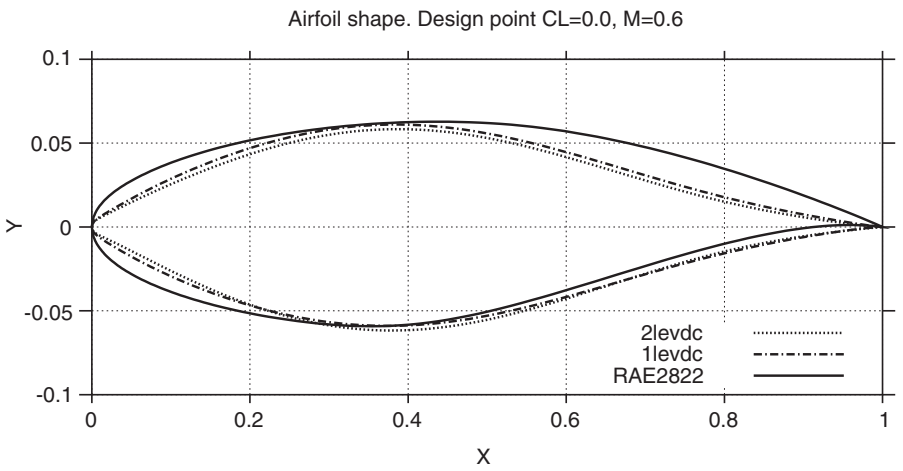

Figure 1. Airfoil shapes optimized on the coarse (1levdc) and medium (2levdc) grids vs original RAE2822 profile.

To additionally decrease the volume of computational work, computational grids coarser than those needed for exact estimations of the objective function are used. This approach may be applied where the grid coarsening preserves the hierarchy of fitness function values on the search space (that is, where the relation of order is invariant with respect to grid coarsening). Formally this condition means that the objective function Q_c defined on a coarse grid can be used for solution of the optimization problem, if for every pair of points x_1, x_2 belonging to the search space, the following relation between the values of an objective function Q_c on a coarser grid:

$$Q_c(x_1) \geqslant Q_c(x_2) \tag{10}$$

implies the same order relation for the objective function Q_f defined on a sufficiently fine grid:

$$Q_f(x_1) \geqslant Q_f(x_2) \tag{11}$$

The applicability of the above principle to the class of optimization problems considered in this paper, was validated by comparing the values of lift and drag, computed on grids of different resolution. It appeared that, for the variety of 2D aerodynamic shapes, the method preserves the hierarchy of drag values for a vast majority of points lying in the parametric space. For each shape, the CFD computations were performed, employing three sequentially refined grids (labeled coarse, medium and fine grids). In the most demanding case, where the coarse grid drag values are compared with those computed on the fine grid, the percentage of the reverse hierarchy is only 3.7% while in the case of the medium/fine comparison, only five violations of hierarchy (out of 1225) were found (0.4%). The results reflect the ability of the code to correctly predict the shock position already on relatively coarse grids. In this context, see also Figures 1–2, where the comparison between two solutions of the same optimization

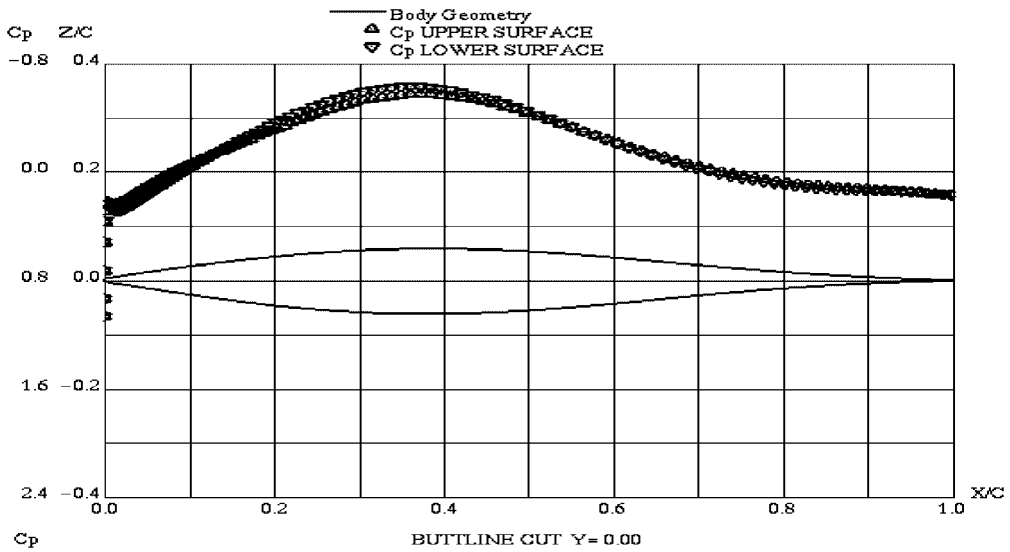

Figure 2. Airfoil optimized on the medium grid at the flight conditions $C_L = 0.0$, $M = 0.6$. Surface pressure distribution at the design point.

problem is presented: the first one employed the coarse grid for CFD computations while the second one used the medium grid.

The use of the ROM–LAM method and grid coarsening allows to reduce the computational volume by at least 2-3 orders of magnitude. However, the total number of CFD runs is still measured by hundreds. This may be acceptable on the research level but an additional major reduction of the overall computation time is vital for the success of the method in engineering environment. To reach this goal it was suggested to employ an embedded multilevel parallelization strategy which includes: (1) parallelization of full Navier–Stokes solver, (2) parallel CFD scanning of the search space, (3) parallelization of the GAs optimization process, (4) parallel optimal search on multiple search domains and (5) parallel grid generation.

The first two levels are intended to improve the computational efficiency of the CFD part of the whole algorithm, while the next two levels are needed in order to reach the same goal for the optimization part of the method.

The first level parallelization which carries out the parallel Navier–Stokes computations [22], is based on the geometrical decomposition principle.

The first level of parallelization is embedded with the second level, which performs parallel scanning of the search space and thus provides parallel CFD estimation of fitness function on multiple geometries. This is applied when executing the steps *Calc_Local_Data_Base* and *Verification_Optim_Cand* in the pseudo-code of the optimization algorithm.

The third level parallelizes the GAs optimization work unit. This level is embedded with the fourth level, which performs parallel optimal search on multiple search domains. It is applied when executing the step *Search_Optim_Candidates* in the pseudo-code of the optimization algorithm.

Finally we can conclude that the five-level parallelization approach allowed us to sustain a high level of parallel efficiency on massively parallel machines, and by this way to dramatically improve the computational efficiency of the suggested optimization algorithm.

As it is seen from the above pseudo-code, the optimization algorithm includes a number of markedly different sub-algorithms. In particular, the sub-algorithms dealing with CFD computations and with genetic optimization search may be mentioned. As it usually happens in practice, such sub-algorithms are not created from scratch, but instead are based on already existing computational core software (which is much less expensive). Moreover, the basic core codes may be written in different programming languages. For example, in our case, the CFD sub-algorithms (employing the core code NES [21, 22]) were written in the C language, while the GAs sub-algorithms employ FORTRAN-77.

In order to resolve the difficulties due to the above mentioned heterogeneity and to ensure the correct interaction between the different parts of the pseudo-code, we drive the overall optimization algorithm by means of a control code, which guides the algorithmic flow stream. The control code was written in the C language which facilitates the interconnection of computational and system software and thus increases the ability to manage different executable codes and system calls.

6. ANALYSIS OF RESULTS

In this section we present the results of optimization which was performed in the framework of the PVM software package on a cluster of MIMD multiprocessors consisting of 72 HP NetServer LP1000R nodes. Each node has two processors, 2 GB RAM memory, 512 KB Level 2 Cache memory and full duplex 100 Mbps ETHERNET interface. In total this cluster contained 144 processors with 144 GB RAM and 36 MB Level 2 Cache memory.

In the implementation of the GA the population size was equal to 200, while the number of search domains was equal to 8. In order to ensure the robustness of the GA search, 10 randomly generated initial populations (for each search domain) were employed.

The number of fitness evaluations by means of the approximated relation (9) was equal to 3 200 000 (population size $\times$ number of generations $\times$ number of initial populations $\times$ number of search domains)/optimization step. The corresponding number of 'exact' Navier–Stokes computations was equal to 63 (36 data base computations + 27 runs for the verification of optimal points). An average number of optimization steps was about 10.

It is interesting to compare the total number of exact CFD evaluations needed for one single-point optimization in the framework of the present approach with that required by the adjoint method [7]. In this book, a similar case (the transonic airfoil drag minimization) required 50 design iterations. Since each iteration includes several full CFD evaluations, the total number of such evaluations can be estimated from 100 to 300. This compares favourably to 630 CFD evaluations required by the current method though these numbers are of the same order of magnitude.

All the results presented in the paper were obtained by means of the modified objective function defined in (6). No tuning of the parameters appearing in relation (6) was needed which is indicative of the robustness of the method.

Another important property of the method is its ability to satisfy non-linear constraints with high accuracy. For example, the constraint imposed on the airfoil thickness was everywhere

satisfied with an accuracy of at least 10^{-8},

$$(t/c) - (t/c)^* \leqslant 10^{-8}$$

The optimization driver CFD solver NES was verified by running a number of 2D and 3D aerodynamic configurations over a wide range of subsonic and transonic flight conditions. The detailed results of comparison with both experiment and available Navier–Stokes solutions may be found in References [21, 22]. The comparison showed that the NES computations not only favourably compare with experiment but also indicate a good grid convergence.

In order to verify the optimization method as a whole, both consistency check as well as comparison with available results of other authors were performed. The first test case was to find an optimal 12% thickness 2D airfoil at the design point $C_L = 0.0$, $M = 0.6$ in the fully turbulent flow regime. Besides the thickness, additional constraints were imposed on the radius of leading edge and trailing edge angle.

It is aerodynamically expectable that the resulting optimal shape should be symmetric, and the verification of this property is a good test to check the consistency of results. In this connection, in order to make the problem more challenging, a highly asymmetric supercritical RAE2822 airfoil was chosen as an initial point of optimization.

The multigrid set of computational grids contained three levels. The fine, medium and coarse meshes comprised 320×96, 160×48 and 80×24 points in the streamwise and normal to the surface directions, respectively. The optimization problem was solved twice, based on CFD computations employing the coarse and medium grids, respectively. Finally, the aerodynamic characteristics of the optimized airfoils were estimated on the fine grid.

The corresponding results are presented in Figures 1–2. The optimized aerodynamic shapes vs original RAE2822 airfoil are shown in Figure 1. It is important to emphasize that the resulting optimal profiles are fairly symmetrical in both considered cases, especially taking into account that the computational meshes originating from the initial asymmetrical airfoil are far from being symmetric. An additional indication to the symmetry of the optimal solution may be found in Figure 2, where the surface pressure distribution for the airfoil optimized on the medium mesh is given.

The optimal shapes corresponding to the coarse and medium CFD computations are rather close one to another with values of the total drag coefficient C_D equal to 74.1 and 73.5 drag counts, respectively. This additionally justifies the applicability of the hierarchy principle described in Section 5.

To further verify the optimization method, the following multipoint optimization of RAE2822 airfoil was employed. The main design point was $M = 0.734$, $C_L = 0.8$, $Re = 6.5 \times 10^6$ while the secondary design points were: $M = 0.754$, $C_L = 0.74$, $Re = 6.2 \times 10^6$ and $M = 0.680$, $C_L = 0.56$, $Re = 5.7 \times 10^6$. The target was to minimize a weighted combination of total drag values at these points with the following weight coefficients: $w_1 = 0.5$, $w_2 = 0.25$, $w_3 = 0.25$. The constraints were imposed on airfoil thickness and leading edge radius which cannot decrease. This optimization problem was used as a test case within the European project AEROSHAPE [31].

The comparison of drag reduction achieved by the current optimization tool with the corresponding AEROSHAPE results is summarized in Table I.

It can be observed that the current algorithm achieves an essentially higher drag reduction, especially at the high transonic flight conditions. A detailed analysis shows that this is attributed to a successful shock destruction which allowed to eliminate the most of wave drag.

Table I. Drag reduction (counts) for multipoint transonic test case.

Design point	Current optimization	Quagliarella [31]
$M = 0.734,\ C_L = 0.80$	−59.0	−40.0
$M = 0.754,\ C_L = 0.74$	−103.0	−34.0
$M = 0.680,\ C_L = 0.56$	+2.0	+3.0

Note: Comparison between current optimization and the results by Quagliarella [31]. One aerodynamic count = 0.0001.

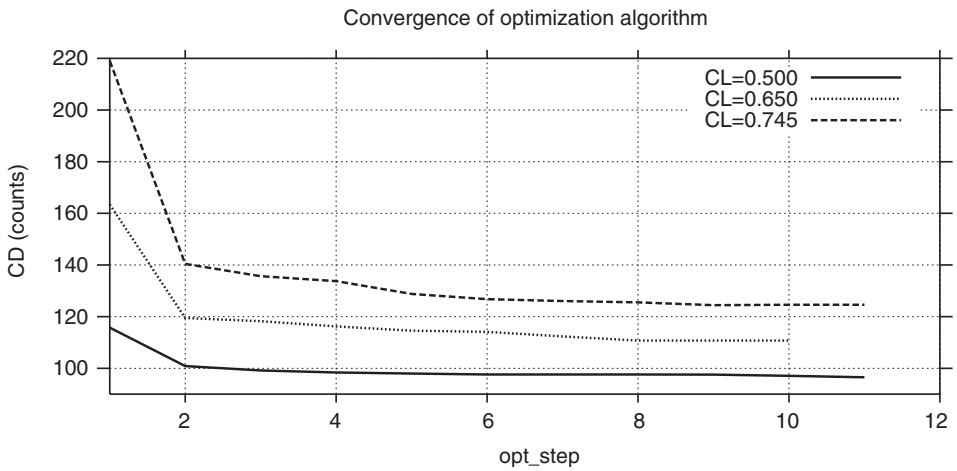

Figure 3. One-point 12% thickness optimizations starting from RAE2822 airfoil. Convergence history for design points: $M = 0.75,\ C_L = 0.5,\ C_L = 0.65$ and $C_L = 0.745$.

In order to compare the present approach with the gradient-based optimization techniques, the design point $C_L = 0.65$, $M = 0.75$ (a moderate shock case) was employed. The case served for verification studies in a number of publications, most recently in Reference [6], where the adjoint approach to viscous aerodynamic shape optimization was applied.

At this point the initial solution (corresponding to the RAE2822 airfoil) gives the total of 149 drag counts (94 counts due to pressure drag and 55 counts due to viscous forces), while the corresponding drag values in Reference [6] amount to 148, 92 and 56 counts, respectively. The two initial solutions are in reasonable agreement, thus giving a fair basis for comparison of the optimization results.

In Reference [6] a reduction of 50 drag counts was achieved when the total drag C_D was used as the objective function. In the present work, a converged (after nine optimization steps) optimal solution gave the same reduction in the total drag coefficient value.

The above mentioned convergence rate is typical of the suggested optimization algorithm. This is illustrated in Figure 3, where convergence history for different optimization cases is shown.

Since the relative thickness of aerodynamic profile is one of its major characteristics, the influence of the constraint $(t/c)^*$ upon the solution of the optimization problem was studied. The results are shown in Figures 4–5, where drag polars and shapes of airfoils, optimized at

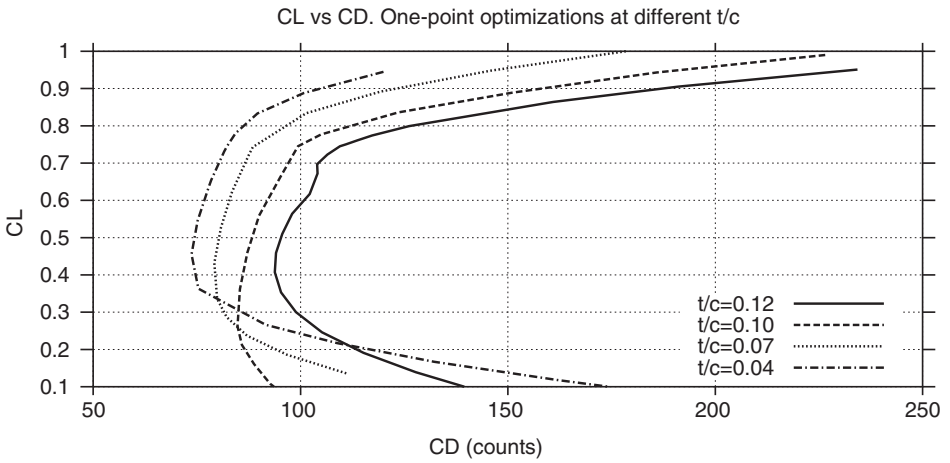

Figure 4. Drag polars of airfoils optimized for different $(t/c)^*$ values. Design point $C_L = 0.745$, $M = 0.75$.

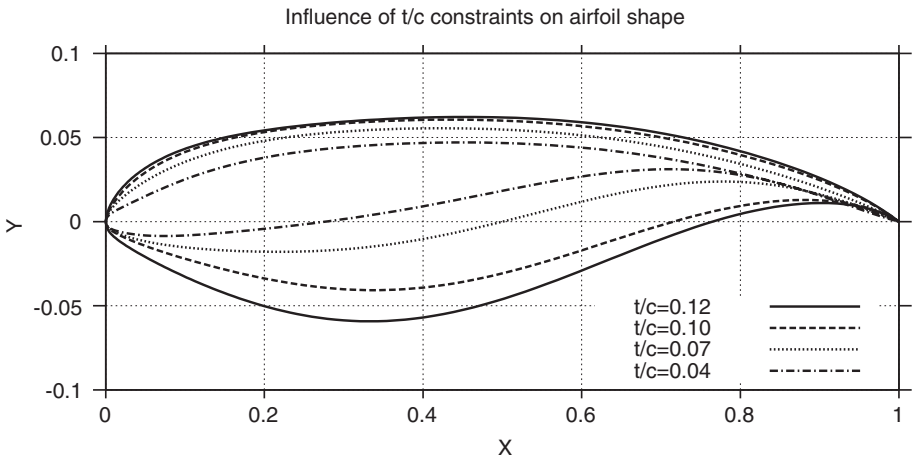

Figure 5. Shape of airfoils optimized for different $(t/c)^*$ values. Design point $C_L = 0.745$, $M = 0.75$.

a high design $C_L = 0.745$ for different $(t/c)^*$ values are, respectively, depicted. It is seen that the above constraint makes a strong impact upon the solution of the optimization problem.

Note, that in accordance with aerodynamic common sense, if no constraints on the solution are imposed, an optimal shape tends to become infinitely thin, which is practically infeasible. The obtained results confirm the above trend: profiles optimized for a lower value of $(t/c)^*$ possess a lower total drag value C_D. The above property holds well beyond the design point $C_L = 0.745$ down to $C_L = 0.35$, with a certain pay-off in total drag at lower C_L values.

In aerodynamic practice, the global rather than pointwise behaviour of airfoils is essential. For this reason, in order to estimate the off-design performance of the optimized airfoils, it is important to compare (along with the drag polars), Mach drag rise behaviour of the

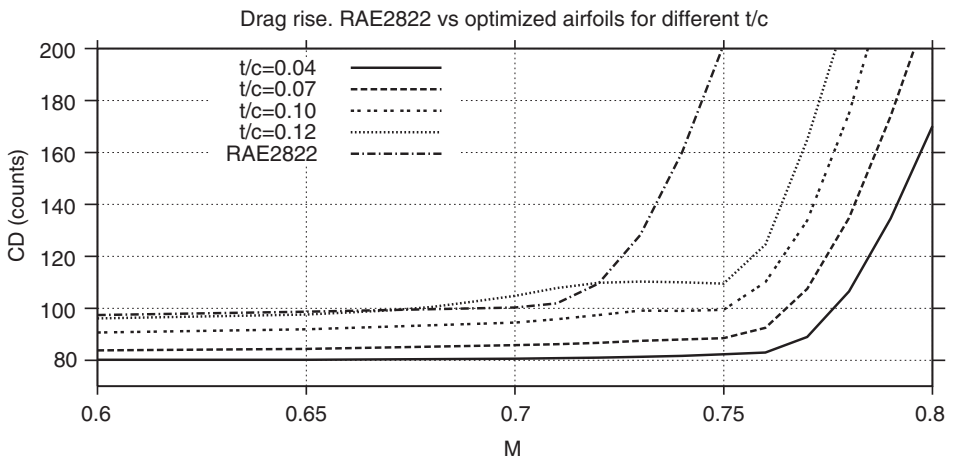

Figure 6. Mach drag rise curve at fixed $C_L = 0.745$. RAE2822 airfoil vs profiles optimized for different t/c.

optimized airfoils, with that of the original RAE2822 profile. In Figure 6, Mach drag rise curves at fixed $C_L = 0.745$ are shown, for the original RAE2822 airfoil alongside those for the optimized airfoils for different $(t/c)^*$ values.

It is seen that for the original airfoil, the drag divergence occurs immediately after $M = 0.71$. This means that for RAE2822, the design point $C_L = 0.745$, $M = 0.75$ lies far inside the domain of drag divergence. Note, that all the optimizations result in essential extension of the low drag zone up to at least $M = 0.75$, and even at lower off-design Mach numbers, the optimized profiles possess a slightly lower drag than the original airfoil.

The requirement of non-zero positive thickness of profile in internal points ensures the absence of 'fish-tails'. However this constraint does not ensure the practical feasibility of airfoil in the trailing edge region. To reach this goal an additional requirement on the trailing edge angle should be imposed ($\theta_T^* > 0$).

The influence of θ_T^* (minimum allowed value of θ_T) on the results of optimization is shown in Figures 7–8. It is seen that the optimization with the value of $\theta_T^* = 3.4^\circ$ leads to a slightly worse performance at the design lift point and at higher lift values (compared with the zero value of θ_T^*), while it highly favours the lift coefficients below $C_L = 0.45$. In terms of aerodynamic shape, a more constrained trailing edge optimization produces a more moderate curvature distribution on the lower surface, especially near the leading and trailing edges of the profile.

The presented results indicate an acceptable level of accuracy, robustness and computational efficiency of the suggested optimization method. Essential improvement of aerodynamic performance has been achieved in high transonic regime optimizing the RAE2822 airfoil. Note, that the above airfoil was chosen as the starting point in the optimization being an established test case, though RAE2822 is far from being optimal for the considered flight conditions.

For the above reason it was interesting to check the performance of the method starting from an airfoil which already possesses a fairly good aerodynamic behaviour at the point

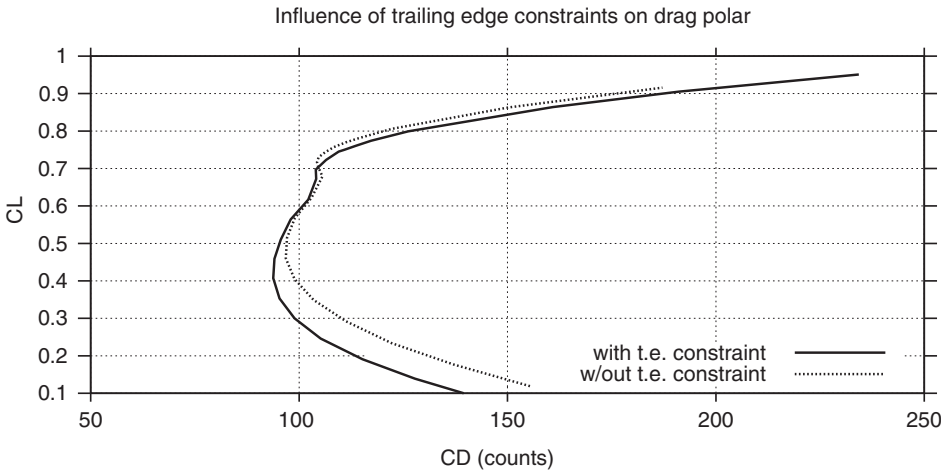

Figure 7. Influence of design value of the trailing edge angle on drag polar. Solid line $\theta_T^* = 3.4^\circ$. Dotted line $\theta_T^* = 0.0^\circ$.

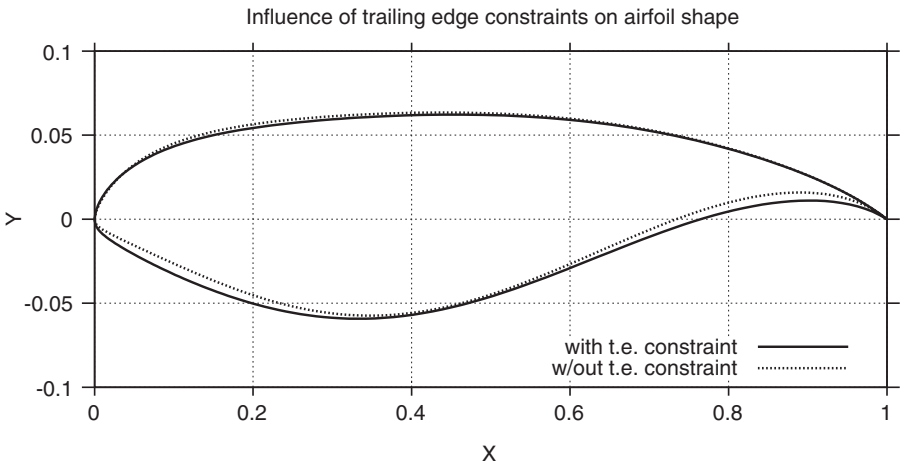

Figure 8. Influence of design value of the trailing edge angle on aerodynamic shape. Solid line $\theta_T^* = 3.4^\circ$. Dotted line $\theta_T^* = 0.0^\circ$.

of design. We took as the starting point, a 18% thickness airfoil which was the result of multipoint optimization by means of the commercial code ISES [32] at the following two design points: $M = 0.6$, $C_L = 0.4$ (point A) and $M = 0.4$, $C_L = 0.75$ (point B). The transition was fixed at 30% of the chord on both upper and lower surfaces.

The current method was applied to the following cases: two single point optimizations (at the above mentioned design points) and a multipoint optimization using a weighted combination of the total drag values at the same points: $C_D = 0.6 \cdot C_D(A) + 0.4 \cdot C_D(B)$.

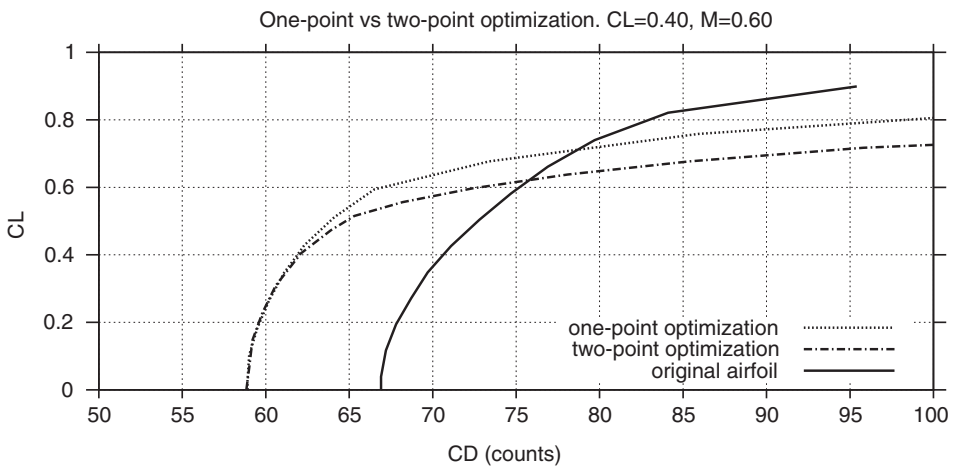

Figure 9. Drag polars at $M = 0.60$. One- and two-point optimizations vs original airfoil.

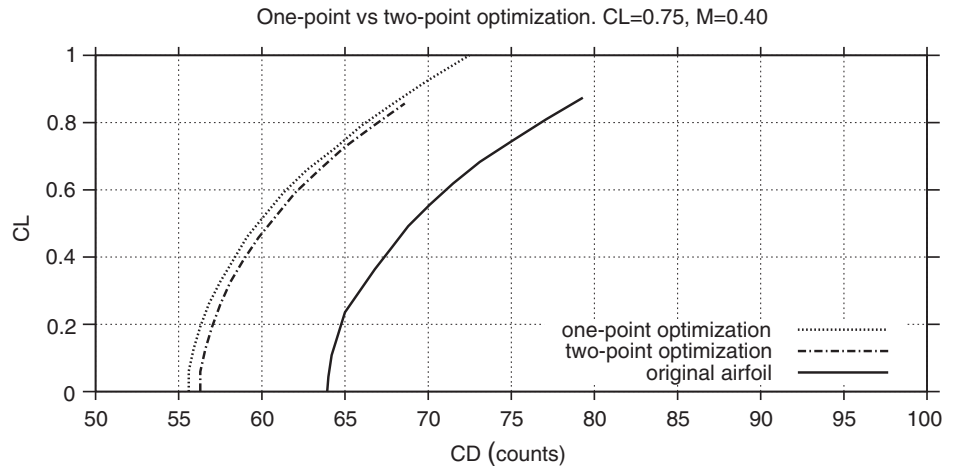

Figure 10. Drag polars at $M = 0.40$. One- and two-point optimizations vs original airfoil.

The results of the optimization are presented in Figures 9–15. In Figure 9, the drag polar at $M = 0.60$ for the original airfoil is compared with those corresponding to the above one- and two-point optimizations, while the respective results at $M = 0.40$ are given in Figure 10. The corresponding comparisons of the aerodynamic shapes are presented in Figures 11–12 while the surface pressure distributions at point B are shown in Figures 13–15.

It may be observed that also in this case the method (both in a one-point and two-point mode) allowed to obtain a significant improvement of the aerodynamic performance at the design points as well as far beyond them. It is important, that the two-point optimization, compared to the one-point optimizations, results in an airfoil shape which possesses almost

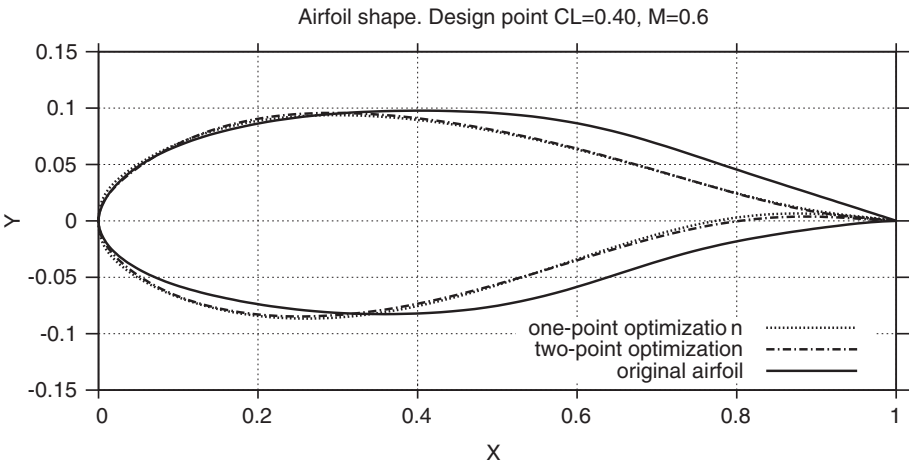

Figure 11. Profile shape. One- and two-point optimizations vs original airfoil.

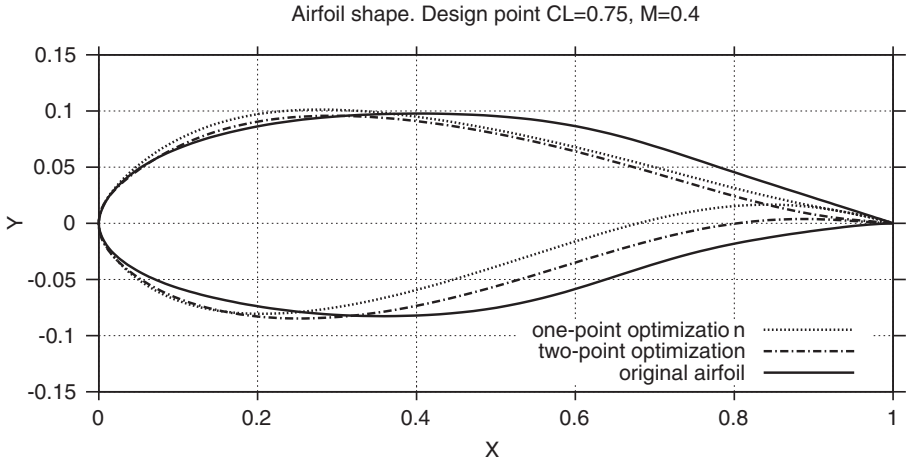

Figure 12. Profile shape. One- and two-point optimizations vs original airfoil.

identical drag value at point A while the respective value of C_D at point B is only slightly higher.

Note, that the analysis of surface pressure shows that the upper surface distribution of the original shape (see Figure 13) is not far from a triangular one. In the view of aerodynamic common sense, this indicates that the original airfoil is already reasonably good. The optimization at point B leads to a practically triangular distribution (see Figure 14). Additionally, it can be observed that the new shape is more rear-loaded than the original one. The two-point optimization retains a triangular form of the upper surface pressure distribution, while the C_p behaviour in the leading and trailing edge areas is closer to the original one.

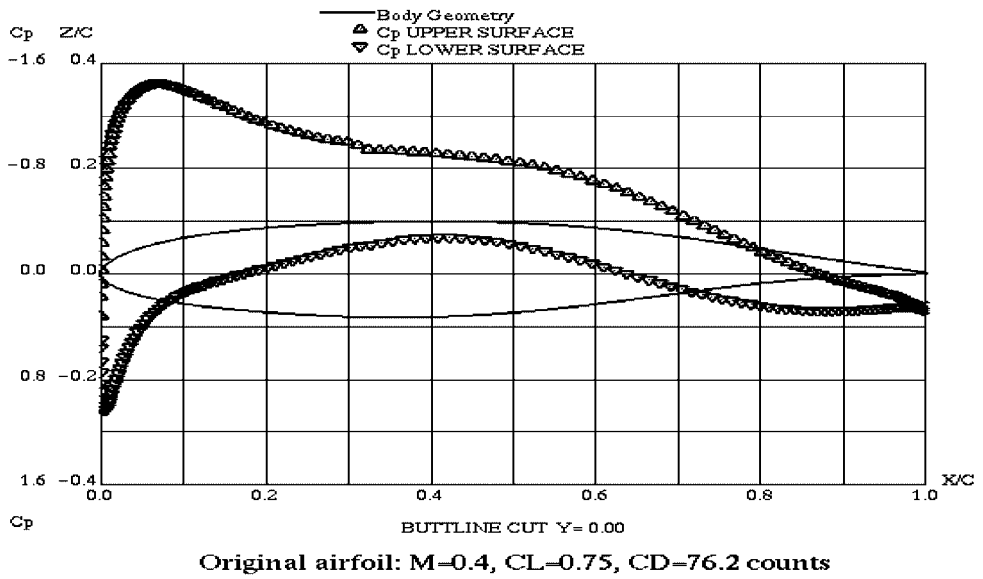

Figure 13. 18% thickness original airfoil. Surface pressure distribution.

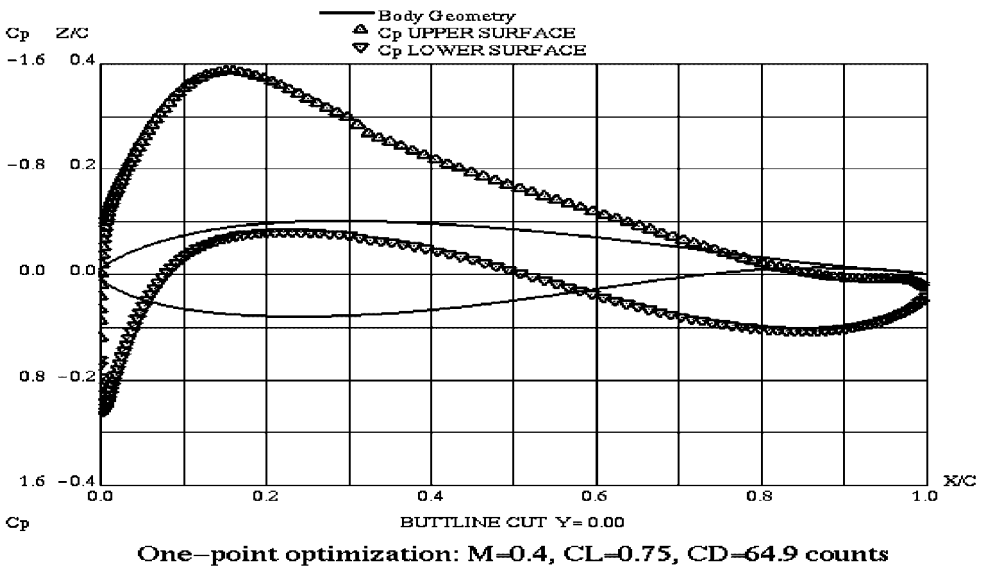

Figure 14. 18% thickness one-point optimization. Surface pressure distribution.

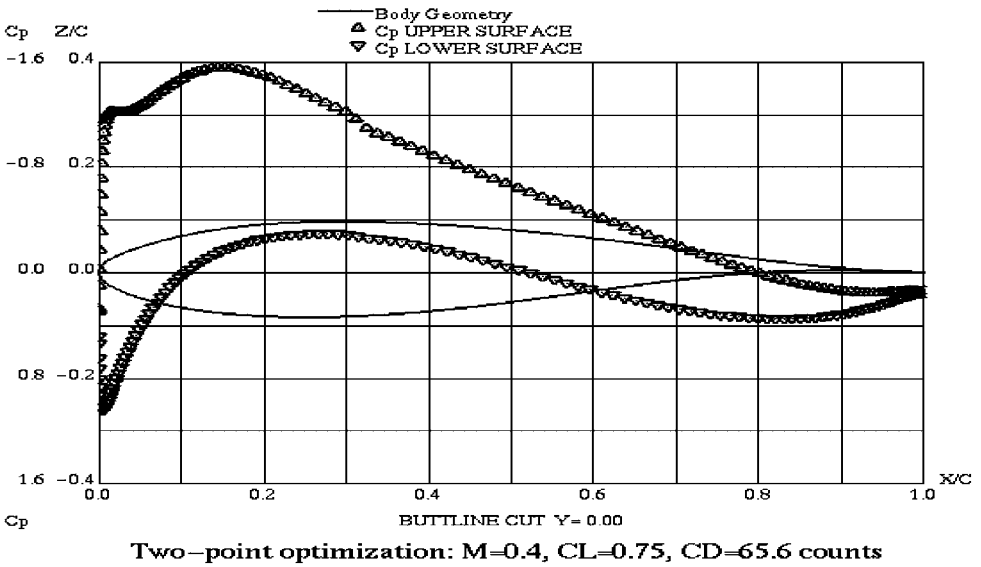

Figure 15. 18% thickness two-point optimization. Surface pressure distribution.

7. CONCLUSIONS

A robust hybrid GA/ROM approach to the multiobjective constrained optimization of aerodynamic configurations is proposed. Novel features of the algorithm include an efficient treatment of nonlinear constraints in the framework of GAs and scanning of the optimization search space by a combination of full Navier–Stokes computations with the ROM method, along with efficient multilevel parallelization strategy which makes use of computational power supplied by massively parallel processors.

The method was applied to the one-point and multipoint optimization of 2D airfoils with a variety of nonlinear constraints. The results demonstrated that the method retains high robustness of conventional GAs while keeping CFD computational volume at an acceptable level due to a limited use of full Navier–Stokes computations. This allowed the employment of the method in a demanding engineering environment.

REFERENCES

1. Lighthill MJ. *A New Method of Twodimensional Aerodynamic Design*. ARC, Rend M 2112.
2. *Optimum Design Methods for Aerodynamics*, AGARD-R-803, November, 1994.
3. Holst TL, Pulliam TH. Evaluation of genetic algorithm concepts using model problems. *Part II*: *Multi-objective Optimization. NASA/TM*, No. 2003-212813, 2003.
4. Sasaki D, Obayashi S, Nakahashi K. Navier-Stokes optimization of supersonic wings with four design objectives using Evolutionary Algorithm. *AIAA Paper No.* 2001-2531, 2001.
5. Pironneau O. *Optimal Shape Design for Elliptic Systems*. Springer: New York, 1984.
6. Nadarajah SK, Jameson A. Studies of the continuous and discrete adjoint approaches to viscous automatic aerodynamic shape optimization. *AIAA Paper*, AIAA-2001-2530, 2001.
7. Mohammadi B, Pironneau O. *Applied Shape Optimization for Fluids*. Oxford University Press: Oxford, 2001.
8. Hajela P. Nongradient methods in multidisciplinary design optimization—status and potential. *Journal of Aircraft* 1999; **36**:255–265.

9. Van Veldhuizen D, Lamont G. Multiobjective evolutionary algorithms: analyzing the state-of-the-art. *Evolutionary Computation* 2000; **8**:125–147.
10. Michalewicz Z, Deb K, Schmidt M, Stidsen Th. Evolutionary algorithms for engineering application. In *Evolutionary Algorithms in Engineering and Computer Science*. Wiley: New York, 1999; 73–94.
11. Michalewicz Z, Janikow CZ. Handling constraints in genetic algorithms. *Proceedings of the 4th International Conference on Genetic Algorithms*. Morgan Kaufmann: Los Altos, CA, 1991; 151–157.
12. Koziel S, Michalewicz Z. A decoder-based evolutionary algorithm for constrained parameter optimization problems. *Proceedings of the 5th Conference on Parallel Problems Solving from Nature*. Springer: Berlin, 1998.
13. Schoenauer M, Xanthakis S. Constrained GA optimization. *Proceedings of the 5th International Conference on Genetic Algorithms*. Morgan Kaufmann: Los Altos, CA, 1993; 573–580.
14. Powell D, Skolnick MM. Using Genetic Algorithms in engineering design optimization with non-linear constraints. *Proceedings of the 5th International Conference on Genetic Algorithms*. Morgan Kaufmann: Los Altos, CA, 1993; 424–430.
15. Richardson JT, Palmer MR, Liepins G, Hillard M. Some guidelines for genetic algorithms with penalty functions. *Proceedings of the 3rd International Conference on Genetic Algorithms*. Morgan Kaufmann: Los Altos, CA, 1989; 191–197.
16. Smith A, Tate D. Genetic optimization using a penalty functions. *Proceedings of the 5th International Conference on Genetic Algorithms*. Morgan Kaufmann: Los Altos, CA, 1993; 499–503.
17. Homaifar A, Lay HY, Qi X. Constrained optimization via genetic algorithms. *Simulation* 1994; **62**:242–254.
18. Michalewicz Z, Attia N. Evolutionary optimization of constrained problems. *Proceedings of the 3rd Annual Conference on Evolutionary Programming*. World Scientific: Singapore, 1994; 98–108.
19. Miettinen K, Makela MM, Makinen J. Handling constraints with penalty techniques in genetic algorithms—numerical comparison. *Reports of the University of Jyvaskyla*, *No.* B10/1999, 1999; 1–18.
20. Le Rich RG, Knopf-Lenor C, Haftka RT. A segregated Genetic Algorithm for constrained structural optimization. *Proceedings of the 6th International Conference on Genetic Algorithms*. Morgan Kaufmann: Los Altos, CA, 1995; 558–565.
21. Epstein B, Rubin T, Seror S. Accurate multiblock Navier–Stokes solver for complex aerodynamic configurations. *AIAA Journal* 2003; **41**:582–594.
22. Peigin S, Epstein B, Rubin T, Seror S. Parallel large scale high accuracy Navier–Stokes computations on distributed memory clusters. *The Journal of Supercomputing* 2004; **27**:49–68.
23. Harten A, Engquist B, Osher S, Chakravarthy S. Uniformly high order accurate non-oscillatory schemes I. *Journal of Computational Physics* 1987; **71**:231–303.
24. Shu CW, Osher S. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics* 1989; **83**:32–78.
25. Michalewicz Z. *Genetic Algorithms + data structures = Evolution Programs*. Springer: New York, 1996.
26. Sefrioui M, Periaux J, Ganascia JG. Fast convergence thanks to diversity. *Proceedings of the 5th Annual Conference on Evolutionary Programming*. MIT Press: Cambridge, MA, 1996; 321–335.
27. Goldberg DE. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley: Reading, MA, 1989.
28. Vicini A, Quagliarella D. Airfoil and wing design through hybrid optimization strategies. *AIAA Paper No.* 98-2729, 1998.
29. Dowell EH. Eigen-mode analysis in unsteady aerodynamics: reduced-order models. *Proceedings of the AIAA 36th Structures, Structural Dynamics and Materials Conference*, Reston, VA, 1995; 2545–2557.
30. Raveh DE. Reduced-order models for nonlinear unsteady aerodynamics. *AIAA Journal* 2001; **39**:1417–1429.
31. Quagliarella D. Airfoil design using Navier–Stokes equations and an asymmetric multi-objective Genetic Algorithm. *Proceedings of EUROGEN-2003*. CIMNE: Barcelona, 2003.
32. Giles MB, Drela M. Two-dimensional transonic aerodynamic design method. *AIAA Journal* 1987; **25**: 1199–1206.
33. Michalewicz Z, Schoenauer M. Evolutionary computation for constrained parameter optimization problems. *Evolutionary Computation* 1996; **4**:1–32.
34. Kim JH, Myung H. Evolutionary programming techniques for constrained optimization problems. *IEEE Transactions on Evolutionary Computation* 1997; **1**:129–140.

