Int. J. Simul. Multidisci. Des. Optim. **11**, 11 (2020)

https://doi.org/10.1051/smdo/2020006

IJSMDO
Available online at:
https://www.ijsmdo.org

RESEARCH ARTICLE OPEN ACCESS

# Multiobjective aerodynamic shape optimization of NACA0012 airfoil based mesh morphing

Rabii El Maani[1,*], Soufiane Elouardi[2], Bouchaib Radi[3], and Abdelkhalak El Hami[4]

[1] LSMI, ENSAM Meknès, Marjane 2, Morocco
[2] LIMII, FST Settat, Morocco
[3] LIMII, FST Settat, Morocco
[4] LMN, INSA de Rouen, France

Received: 24 March 2020 / Accepted: 15 June 2020

**Abstract.** The actual use of computational fluid dynamics (CFD) by aerospace companies is the trade-off result between the perceived costs and benefits. Computational costs are restricted to swamp the design process even if the benefits are widely recognized. The need for fast turnaround, counting the setup time, is also crucial. CFD integrates mathematical relations and algorithms to analyze and solve fluid flow problems. CFD analysis of an airfoil produces results such as the lift and drag forces that determine the performance of an airfoil. Thus, optimizing these aerodynamic performances has proved extremely valuable in practice. The aim of this paper is to model a transonic, compressible and turbulent flow over a NACA 0012 airfoil, using a density based implicit solver, for which a comparison and a validation will be made throught the published experimental data. The numerical results show that the predicted aerodynamic coefficients are in a satisfying agreement with experimental data. Then an aerodynamic shape optimization algorithm, based on a multiobjective algorithm that is an extension of the Backtracking Search Algorithm which was initially developed for single-objective optimization problems only, was used in order to obtain an improved performance control of the aerodynamic coefficients of the optimized airfoil.

**Keywords:** CFD / aerodynamic / NACA 0012 / pressure coefficient / genetic algorithm

## 1 Introduction

The current need for faster and more accurate methods for calculating flow fields around technical interest configurations directed the rapid progress of CFD. In the past, CFD was the method of choice in the design of many industrial components and processes in which fluid or gas flows play a major role with many commercial packages available for modeling flow in or around objects. Features and details that are difficult, expensive or impossible to measure or visualize experimentally are shown due to the computer simulations, such as transition from laminar to turbulent flow that plays an important role in determining the flow features and in quantifying the airfoil performance. The creation of the geometry and the meshes with a preprocessor constitute the first step in modeling a problem and a successfully generated mesh for the domain geometry takes usually the majority of time spent on a CFD project in the industry allowing a compromise between desired accuracy and solution cost. Then the governing equations of the problem are solved and the solution are computed and monitored using the solver and the physical models. Afterwards, the results are examined and saved and if necessary, revisions to the numerical or physical model parameters are considered.

In the preliminary design of any airplane, the geometrical dimensions, with selection of the principal components, are analyzed with the aerodynamics characteristics to find the suitable combination. The provided design of the airfoils shapes aims to have, for given flight conditions, high lift values at low drag. More parameters, such taper ratio or aspect ratio, affect the overall lift (L) and drag (D) of the wing. The Reynolds number is also very important for airfoil performance. This number bounds the maximum thickness-to-chord ratio, beyond which point the airfoil will have unacceptable performance. It also determines the achievable section maximum lift coefficient and lift-to-drag ratio. Airfoils provide two-dimensional lift, drag and pitch momentum, which is equivalent to the characteristics of a section of an infinite span wing. Real wings, the wing with finite span, behave quite differently. Optimizing these ratios and forces presents a very

* e-mail: elmaani.rabi3@gmail.com

usefull improvement in the aircraft wings aerodynamic performances.

By definition, optimization aims to obtain the best possible performance of a model, which is formulated in mathematically as the minimization of a function or a set of functions at the same time. This refers to single-objective optimization and multiobjective optimization, respectively. A multiobjective optimization problem (MOP) involves more than one objective function. The task of multiobjective optimization is not to find an optimal solution corresponding to each objective function but to find a set of solutions called Pareto-optimal front [1–3]. The multiplicity of principles involved in real world applications, depending on several variables of a given model, makes the use of native MOP techniques crucial and handling of these participating variables appropriately is the key for successful optimization.

A satisfactory solution under highly nonlinear and complex constraints is conditionned by optimizing simultaneously several objectives. The multiobjective nature of the problem is given by the existence of contradictory objectives meaning that the improvement of one objective implies the deterioration of another [4]. Multiobjective optimization then consists in seeking all the solutions which correspond to the best compromises between the objectives of security to be maximized and of cost to be minimized. The design problem we face therefore involves several contradictory objectives, which requires the use of an optimization method capable of managing the multiobjective nature of the problem. We are more particularly interested in the multiobjective optimization algorithm by Evolutionary Algorithm techniques (EA), which are generally used to find the global optimum when the objective function of an optimization problem is indistinguishable and not linear [5,6]. EAs have been used for problems of dynamic analysis of chemical processes [7], phase stability and equilibrium calculations for reactive systems [8], communication applications [9], dynamic optimization of biochemical processes [10], mechanical design and many other engineering problems.

It is very important, in an EA, to protect a population's diversity for the population's ability to iteratively support its development. An EA tries to develop gradually, through a "trial individual", an individual with a better fitness value using a combination of a choosen existing individuals due to various genetic operators. The trial individual with a better fitness value replaces the original one in the next-generation population. EAs radically differ from one another based on their strategies for generating trial individuals. Because these strategies have a considerable effect on their problem-solving success and speed. It is believed that recombination, crossover, mutation, selection and adaptation [11–13], constitute the genetic diversity of a population. Many EAs are based on basic genetic rules, such as the covariance matrix adaptation evolution strategy (CMAES) [14], genetic algorithms [11,15] and the differential evolution algorithm (DE) [16–18].

NSGA-II [11], SPEA-2 [19] and NNIA [20] are examples of multiobjective optimization algorithms that can process a set of solutions simultaneously in a single run and for which an EA presents an effective tool. Recently, a new evolution algorithm, called backtracking search algorithm (BSA), was proposed and applied in several numerical single-objective optimization problems by Civicioglu [21]. It has been used in the synthesis of concentric circular antenna arrays by Guney et al. [22] and the parameter identification of hyperchaotic system [23]. In this algorithm, information obtained from past generations is used to search for better fitness solutions. It has good convergence and it is very effective in solving numerical optimization problems, compared with the performances of six widely used EA algorithms: JDE, ABC, CMAES, PSO, SADE and CLPSO [21], due to its unique mechanism for generating a trial individual. To generate trial individuals, BSA uses selection, mutation and crossover. In contrast with many genetic algorithms such as DE and its derivatives, for each target individual from individuals of a randomly chosen previous generation, BSA uses only one direction individual based on a random mutation strategy. BSA uses a non-uniform crossover strategy that is more complex than the crossover strategies used in many genetic algorithms [24].

Our objective, in this paper, is to use the backtracking search algorithm for solving multiobjective optimization problems (BSAMO). BSA is extended to deal with multiobjective design problems using the fast non-dominated sorting procedure and the crowding distance. The application of the proposed algorithm to real aerodynamic problems can be considered as a multi-disciplinary design optimization (MDO). The MDO is a field of engineering that focuses on the use of numerical optimization for the design of systems that involve a number of disciplines or subsystems [25–27].

This paper is organized as follows. In Section 3 we introduce the mesh morphing tool used to achieve the motion of the mesh at any specific control points and Section 4 presents some basic concepts of the multi-objective optimization and the Backtracking Search Algorithm for Multiobjective Optimization (BSAMO). Section 5 presents the numerical simulation consisting first of computing the transonic, compressible flow over a NACA0012 airfoil compared to published experimental data, then the coupled solution procedure for the CFD optimization process was presented with the multiobjective optimization results. Finally, the main conclusions are drawn in Section 6.

## 2 Governing equations

### 2.1 Fluid mechanics

We are generally interested in the properties of the flow at certain locations in the fluid domain. Therefore, to describe the fluid flows, the Eulerian formulation is generally used. We limit ourselves to the case of Newtonian fluids which are linear viscous isotropic fluids and the most important for practical applications. The Cauchy $T$ stress tensor of these fluids is characterized

by the following material law:

$$T_{ij} = \mu\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3}\frac{\partial v_k}{\partial x_k}\delta_{ij}\right) - p\delta_{ij} \quad (1)$$

where $p$ is the pressure, $\mu$ the dynamic viscosity, $v_i$ the velocity vector with respect to Cartesian coordinate $x_i$, and the Kronecker symbol $\delta_{ij}$. On the spatial fluid mechanics domain, the Navier-Stokes equations of incompressible flows may be written as:

$$\begin{aligned} \frac{(\partial v)}{\partial t} + \nabla\cdot(\rho v \otimes v - \sigma) - \rho f = 0 \\ \nabla\cdot v = 0 \end{aligned} \quad (2)$$

where $f$, $v$, and $\rho$ are the external force, velocity, and the density, respectively, and $\sigma$ the stress tensor is defined as:

$$\sigma(v,p) = -pI + 2\mu\varepsilon(v). \quad (3)$$

Here $p$ is the pressure, $I$ is the identity tensor, $\mu$ is the dynamic viscosity, and $\varepsilon(v)$ is the strain-rate tensor given by:

$$\varepsilon(v) = \frac{1}{2}\left(\nabla v + \nabla v^T\right). \quad (4)$$

The boundary conditions associated to the fluid domain are:

$$v|_{\Gamma_v} = \overline{v} \quad (5)$$

where $\overline{v}$ can be a known velocity profile at the boundary.

### 2.2 Turbulence model

Turbulence models is a modeling of the terms related to fluctuations in statistical unstable Navier-Stokes equations by introducing mean and fluctuating quantities which form the Reynolds Averaged Navier-Stokes Equations (RANS), due to the average statistical procedure used to obtain these equations the turbulence models based on the RANS equations are called statistical turbulence models. In our study, the $k-\omega$ SST model will be used [28].

#### 2.2.1 K-Omega SST model

According to Menter [29], the model $k-\omega$ of transport by shear stress (SST) combines the two models $k-\omega$ and $k-\varepsilon$, two equations are solved, one for the specific dissipation $\omega$ and the other for the kinetic energy of turbulence $k$, with a robust and precise formulation in the region close to the wall with the freestream independence in the far field.

The SST $k-\omega$ model has a similar form to the standard $k-\omega$ model:

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_i}(\rho k u_i) = \frac{\partial}{\partial x_j}\left(\Gamma_k \frac{\partial k}{\partial x_j}\right) + \tilde{G}_k - Y_k + S_k \quad (6)$$

$$\frac{\partial}{\partial t}(\rho\omega) + \frac{\partial}{\partial x_i}(\rho\omega u_i) = \frac{\partial}{\partial x_j}\left(\Gamma_\omega \frac{\partial \omega}{\partial xj}\right) + G_\omega - Y_\omega + D_\omega + S_\omega \quad (7)$$

where $D_\omega$ represents the cross-diffusion term. $G_\omega$ is given by:

$$G_\omega = \frac{\alpha}{\nu_t} G_k \quad (8)$$

$\tilde{G}_k$ represents the production of turbulence kinetic energy, and is defined as:

$$\tilde{G}_k = \min(G_k, 10\rho\beta^* k\omega) \quad (9)$$

The SST $k-\omega$ model is based on both the standard $k-\omega$ model and the standard $k-\varepsilon$ model. To blend these two models together, the standard $k-\varepsilon$ model has been transformed into equations based on $k$ and $\omega$, which leads to the introduction of a cross-diffusion term $D_\omega$ is defined as:

$$D_\omega = 2(1-F_1)\rho\sigma_{\omega,2}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}. \quad (10)$$

Model constants are:

$$\begin{aligned} \sigma_{k,1} = 1.176, \quad \sigma_{\omega,1} = 2.0, \quad \sigma_{k,2} = 1.0, \quad \sigma_{\omega,2} = 1.168 \\ a_1 = 0.31, \quad \beta_{i,1} = 0.075, \quad \beta_{i,2} = 0.0828, \quad k = 0.41 \end{aligned}. \quad (11)$$

## 3 Mesh morpher

For typical engineering problems, the shape sensitivity field can have smoothness properties that are not adequate to define a shape modification. Mesh morphing technology is used here for two-dimensional system in a two-fold role. The first role is as a smoother for the surface sensitivity field. The second role is to provide smooth distortions not only of the boundary mesh, but also the interior mesh. This approach is very appealing since it functions for arbitrary mesh cell types [30,31].

A Cartesian or cylindrical region is defined such that it encompasses all or a part of the problem domain. Only the mesh nodes that fall within this region may move as a result of the morphing operation. A local $(u, v)$ coordinate system is defined, where $u \in [0, 1]$ and $v \in [0, 1]$. A regular array of $N_u \times N_v$ control points is then distributed in the control volume. The motion of the mesh at any point in the domain is determined by the movement of the control points.

In general, an optimization problem may include both free-form minimization or maximization of the observable (s) and constraints that are localized in space. This requires an approach that can provide a deformation field that is well-behaved and consistent with manufacturability requirements, while still allowing locally sharper deformations where required to satisfy the imposed constraints.

To achieve this, two coincident sets of control points are created. Bernstein polynomials and B-splines are used, respectively, to map the control point motions to the computational mesh nodes.

The $i$th Bernstein polynomial of degree $\ell$ is:

$$B_{i,\ell}(u) = \binom{\ell}{i} u^i (1-u)^{n-i}. \quad (12)$$

The $i$th B-spline of degree $p$ is the non-periodic B-spline that uses a knot-vector, $\{t_i\}$, where:

$$t_i = \begin{cases} 0 & i \leq p \\ (i-p)/(\ell - 2p) & p < i < \ell - p \\ 1 & i \geq \ell - p \end{cases}. \tag{13}$$

The value of the B-spline is defined recursively as:

$$S_{i,0}(t) = \begin{cases} 1 & t_i < t < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

$$S_{i,j}(t) = \frac{t - t_i}{t_{i+j} - t_i} S_{i,j-1}(t) + \frac{t_{i+j+1} - t}{t_{i+j+1} - t_{i+1}} S_{i+1,j-1}(t).$$

Depending on the degree of the B-spline, the function is zero on some portion of the unit interval. This is in contrast to the Bernstein polynomials that are strictly non-zero everywhere on the unit interval.

Let $x_i^v$ denote the $i$th coordinate of the $v$th mesh node, and let $\Delta x_i^v$ denote the change in the position of the node due to the morphing process. Let $\Delta\eta_{jk}^i$ denote the displacement of the $(j, k)$th control point associated with the Bernstein polynomials in the $i$th coordinate direction, and let $\Delta\zeta_{jk}^i$ denote the displacement of the $(j, k)$th control point associated with the B-splines. If $(u^v, v^v)$ denotes the position of the $v$th mesh node, then the displacement of the node is defined by the superposition:

$$\Delta x_i^v = \sum_{j=0}^{N_u} \sum_{k=0}^{N_v} \left( \Delta\eta_{jk}^i B_{jt}(u^v) B_{k,m}(v^v) + \Delta\zeta_{jk}^i S_{j,\ell}(u^v) S_{k,m}(v^v) \right). \tag{15}$$

The control-point movement for the Bernstein polynomials controls the large-scale smooth deformation, while the control-point movement for the B-splines controls fine-scale motions.

## 4 Multiobjective optimization problem

A multiobjective optimization problem deals with more than one objective function and these goals are independent of each other. The task of multiobjective optimization is not to find an optimal solution corresponding to each objective function but to find a set of solutions called Pareto-optimal front.

Let consider the following multiobjective optimization problem (MOP):

$$\begin{cases} \min\limits_{x \in \Omega} \quad \mathrm{f}(x) = (f_1(x), \ldots, f_m(x))^T \\ \text{subject to :} \quad g_i(x) \leq 0, \quad \text{for} \quad i = 1, \ldots, l \\ \qquad\qquad\quad h_k(x) = 0, \quad \text{for} \quad k = 1, \ldots, p, \end{cases} \tag{16}$$

where $m$ is the objective functions number, $\mathbf{f}$ is the concurrent objective functions vector to optimise, $x = (x_1, \ldots, x_n) \in \Omega$ is the $D$-dimensional decision space where each decision variable $x_i$ is bounded by lower and upper limits $x_{li} \leq x_i \leq x_{ui}$ for $i = 1, \ldots, D$, $g_i(x)$ are $l$ inequality constraints and $h_k(x)$ are $p$ equality constraints.

Let us first introduce some basic concepts of the multiobjective optimization [32]:

– *Pareto dominance:* Let $\mathbf{a} = (a_1, \ldots, a_n)$ and $\mathbf{b} = (b_1, \ldots, b_n)$ be two vectors. The vector $\mathbf{a}$ *dominates the vector* $\mathbf{b}$ if and only if $\mathbf{a}$ is partially less than $\mathbf{b}$, i.e.:

$$(\forall k \in \{1, \ldots, m\}: f_k(\mathrm{a}) \leq f_k(\mathrm{b})) \wedge (\exists k \in \{1, \ldots, m\}: f_k(\mathrm{a}) < f_k(\mathrm{b})) \tag{17}$$

($\textit{denoted by}\ \mathbf{a} \succ \mathbf{b}$).

– *Pareto-optimal solution:* A solution $\mathbf{a} \in \Omega$ is said to be Pareto-optimal if there is no solution $\mathbf{b} \in \Omega$ which dominates $\mathbf{a}$, i.e.

$$\nexists \mathrm{b} \in \Omega : \mathrm{b} \preccurlyeq \mathrm{a} \tag{18}$$

– *Pareto-optimal set:* The set $\mathbf{Ps}$ of all Pareto-optimal solutions, as defined by:

$$\mathrm{Ps} = \left\{ \mathrm{x} \middle| \nexists\, \mathrm{a} \in \Omega : f(\mathrm{a}) \preccurlyeq f(\mathrm{x}) \right\} \tag{19}$$

– *Pareto front:* The Pareto front, denoted $\mathbf{P_F}$, is the image of Pareto optimal set $\mathbf{Ps}$ in the objective space and is defined as follows:

$$\mathbf{P_F} = \{ \mathrm{f}(x) : \mathrm{x} \in \mathbf{Ps} \} \tag{20}$$

### 4.1 Multiobjective optimization based Backtracking Search Algorithm

In this paper, BSA, which was initially developed for single-objective optimization problems only, is extended to handle MOPs. This development maintain the BSA's spirit and allows a good efficiency with great benefit from new capacities of exploration of the search space and of an adapted search direction matrix due to the adaptation of BSA's main strategies and algorithm simplicity. The main development should concern the fitness values assigned to individuals in the population.

Algorithm 1 gives the proposed pseudocode of BSAMO (Backtracking Search Algorithm for Multi-objective Optimization). It integrates BSA's mutation and crossover operators presented in [24] and the fast non-dominated sorting and the crowding distance of Deb et al. [11].

Some explanations for this last item are given in the following subsections.

**Algorithm 1:** Pseudocode of BSAMO

**Function** P = BSAMO($D, N, m, \mathbf{f}(x), \mathrm{Max}_G, x_l, x_u$)
1: Generate the initial population $\mathbf{P}$ and the historical population $\mathbf{P}_h$;
2: $\mathbf{P}$ := *Find_Non_Dominated_Crowding_Distance* ($\mathbf{P}$ | $\mathbf{f}$(x));
3: **for** $t = 1 : \mathrm{Max}_G$, **do**
4: $[\mathbf{P}_c, \mathbf{P}_h]$ := BSA_Operator ($\mathbf{P}, x_l, x_u, \mathbf{P}_h$); // BSA_Operator: Mutation and Crossover;
5: $\mathbf{P}$ := *Find_Non_Dominated_Crowding_Distance* ([$\mathbf{P}$;$\mathbf{P}_c$] | $\mathbf{f}$(x));
6: Sort and find the current Pareto optimal solution;
7: **end for**

In the elaboration of the fast non-dominated sorting procedure, for every solution, and before creating the first non-dominated front and initialising it with all solutions having zero as domination count, the number of solutions which dominate the solution $p$, the domination count $n_\mathrm{p}$, and the set of solutions that the solution $p$ dominates $S_p$ are calculated. Then, for each solution $p$ with $n_\mathrm{p} = 0$, each member $q$ of its set $S_\mathrm{p}$ is visited, and its domination count is reduced by one. The second non-dominated front is then created as the union of all individuals belonging to Q, which is a separate list of any member with a domination count equal to zero. The procedure is repeated for subsequent fronts ($\mathrm{F}_3$, $\mathrm{F}_4$, etc.) until all individuals are assigned their ranks. The fitness is set to a level number, lower numbers correspond to higher fitness ($\mathrm{F}_1$ is the best). The fast non-dominated sorting procedure was developed in the framework of NSGA-II [11].

The average distance between two individuals located on either side of the given particular solution along each objective is called the crowding distance. It is an estimation of the perimeter of cuboid formed using the nearest neighbours as mentioned in Figure 1. This metric represents half of the perimeter of the cuboid encompassing the solution $i$. The sum of individual crowding distance values corresponding to each objective gives the overall crowding distance value [11].

Based on the normalised values of objectives, the computation of the crowding distance is given by Algorithm 2, where $f_m^{max}$ and $f_m^{min}$ are the maximum and minimum values of the $m$th objective function, respectively.

**Algorithm 2:** Crowding distance calculation for a set solution $\mathcal{I}$

1: $n = |\mathcal{I}|$ // number of solutions in $\mathcal{I}$;
2: **for** each $i$, **do**
3: set $\mathcal{I}[i]_{distance} = 0$;
4: **end for**
5: **for** each objective $m$, $\mathcal{I} = \mathrm{sort}(\mathcal{I}, m)$ **do**
6: $\mathcal{I}[1]_{distance} = \mathcal{I}[n]_{distance} = \infty$;
7: **for** $i = 2$ to $(n-1)$ **do**
8: $\mathcal{I}[i]_{distance} = \mathcal{I}[i]_{distance} + (\mathcal{I}[i+1].{}_m - \mathcal{I}[i-1].{}_m)/(f_m^{max} - f_m^{min})$;
9: **end for**
10: **end for**

$\mathcal{I}$ is a non-dominated set, $\mathcal{I}[i]_m$ is the $m$th objective value of the $i$th individual in $\mathcal{I}$, $n$ is the number of elements of $\mathcal{I}$, and sort ($\mathcal{I}$, $m$) is sorting of the individuals of $\mathcal{I}$ according to the $m$th objective. The theoretical aspect of this algorithm is developed in [16,24].

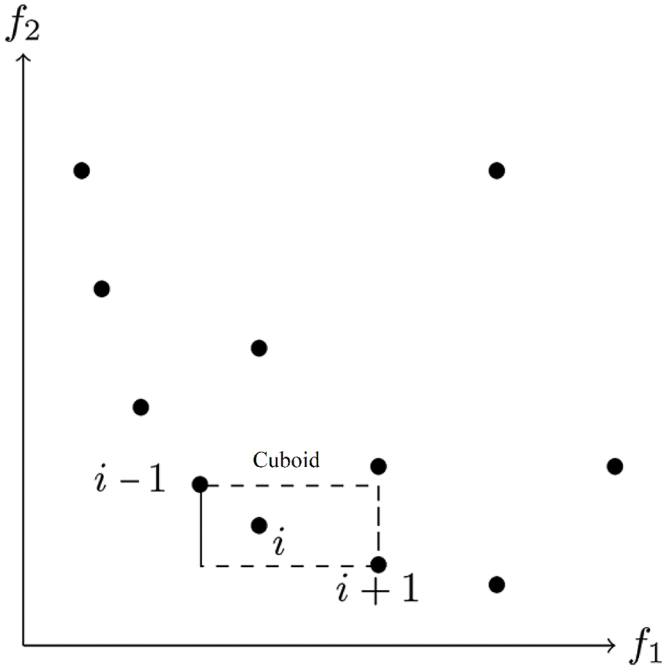


**Fig. 1.** Crowding distance of individual $i$.

**Table 1.** Parameters of the numerical simulation.

| Parameters | |
|---|---|
| Viscosity $\mu$ | $1.7894 \times 10^{-5}$ kg/m.s |
| Density $\rho$ | 1.225 kg/m$^3$ |
| Temperature | 311 K |
| Incidence $\alpha$ | 1.55° |

## 5 Numerical simulation

### 5.1 Problem statement

In this work, a numerical simulation of the NACA0012 airfoil was evaluated two different Mach numbers, $M = 0.5$ and $M = 0.7$, same with the reliable experimental data from T.J. Coakley (1987) [33], in order to validate the present simulation. The flow can be described as compressible for this Mach numbers and the numerical simulation was conducted using FLUENT with a segregated implicit solver. Table 1 illustrates the parameters used in this numerical simulation.

In the appropriate boundary conditions of the fluid domain we will be setting "pressure far-field" for the inlet and the outlet where the gauge pressure, the Mach number, the temperature, and the components of the flow direction are given, and for the airfoil lower and upper a "wall" boundary condition is set and the velocity there will be 0.

The airfoil profile meshes were created using structured meshes, which consist of a variety of quadrilateral elements. The resolution of the mesh was greater in regions where greater computational accuracy was needed, such as the region close to the airfoil (Fig. 2). Generally,

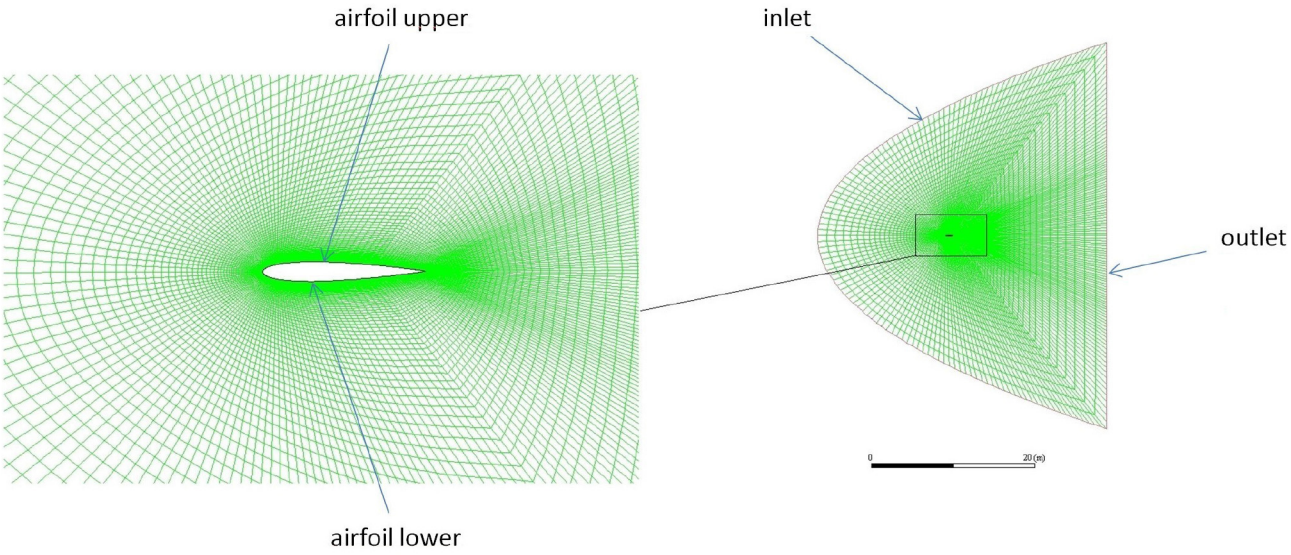


**Fig. 2.** Boundary conditions.

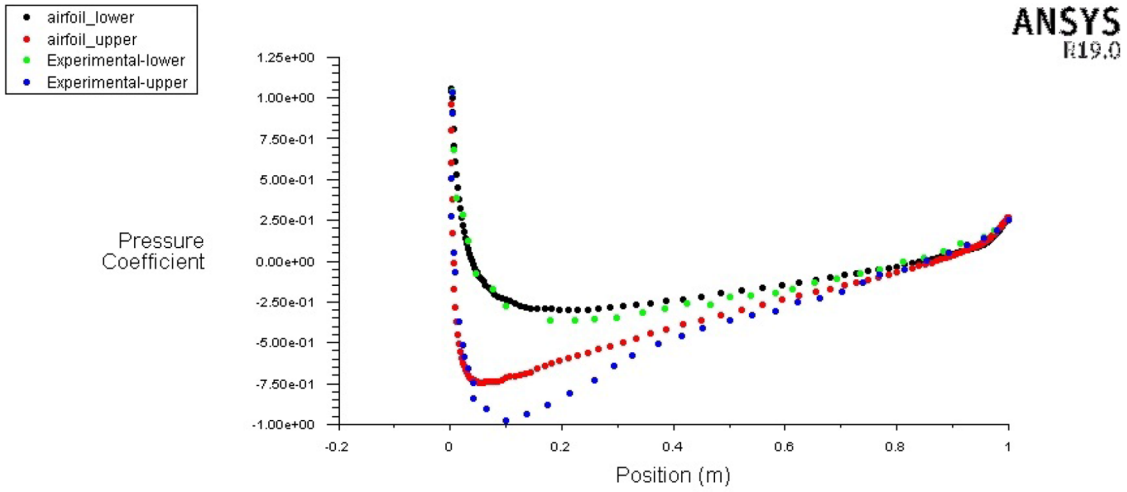


**Fig. 3.** Pressure coefficient along airfoil surfaces for Mach 0.5.

a numerical solution becomes more accurate as more nodes are used and the appropriate number of nodes can be determined by increasing the number of nodes until the mesh is sufficiently fine so that further refinement does not change the results.

## 5.2 CFD and validation results

This symmetrical airfoil is widely used for testing different computing methods and is considered as the main source of experimental data used here for comparison, and collected in the Experimental Data Base for Computer Program Assessment (AGARD-AR-138, 1979) for checking computational fluid dynamic methods. The data on the flow over the airfoil are obtained in several wind tunnels, but, according to Hoist (1987) [34], it is best to take the data from Harris's tests in the Langley 8-Foot Transonic Pressure Tunnel [35]. Computations were performed at a Reynolds number of $\mathrm{Re} = 9.10^6$. Comparison of the computed results with Harris's data for the pressure coefficient are shown in Figures 3 and 4 at $\mathrm{M} = 0.5$ and $\mathrm{M} = 0.7$, respectively, and at an angle of attack $\alpha = 1.55°$. Table 2 pressents a comparision of the lift and drag coefficients $\left(C_l = \frac{L}{\frac{1}{2}\rho S V^2}, C_d = \frac{D}{\frac{1}{2}\rho S V^2}\right)$ with the experimental data.

Figures 3 and 4 show experimental and computed pressure and forces distributions at a Mach number of 0.5 and 0.7 and with angle of attack of 1.55°. In the first case The computations are in good agreement with experimental data but a significant difference can be seen at the upper airfoil near the chordwise location of $\mathrm{x/c} = 0.15$. The computed lift and drag coefficients for those cases are compared with experimental normal force and drag coefficients in Table 2. In the second case ($\mathrm{M} = 0.7$) there is a small amount of separation, and the computed and experimental pressure distributions are in close agreement

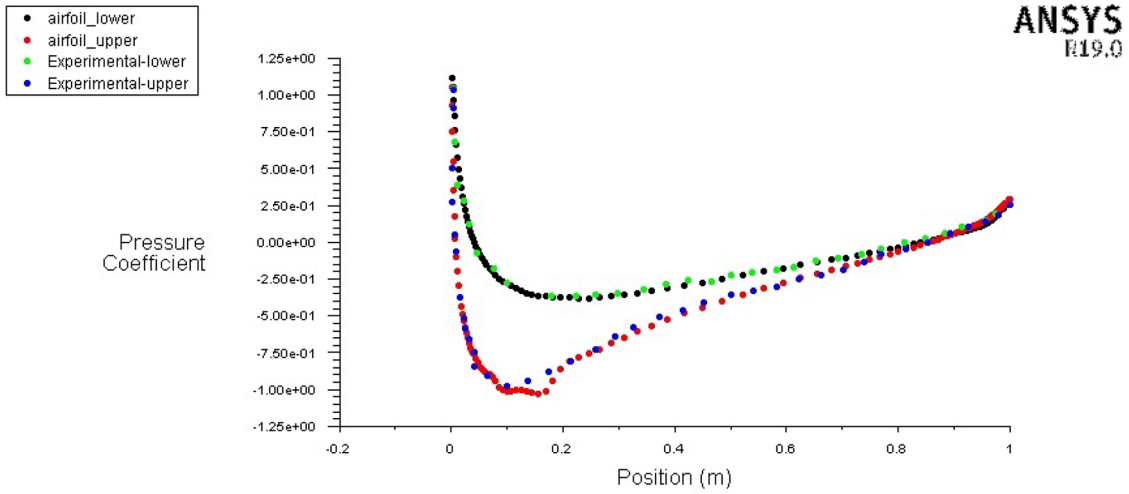


**Fig. 4.** Pressure coefficient along airfoil surfaces for Mach 0.7.

**Table 2.** Drag ($C_d$) and lift ($C_l$) coefficients comparison.

| | Experimental data | Mach 0.5 | Mach 0.7 |
|---|---|---|---|
| $C_l$ | 0.241 | 0.190 | 0.239 |
| $C_d$ | 0.0079 | 0.0079 | 0.0081 |

with one another. The computations do, however, indicate the presence of a shock wave, near the chordwise location of x/c = 0.15, which causes the separations at the upper airfoil ; it is not apparent in the experimental pressure distrlbutions. It is found that the second case gives lift and drag coefficients that are slightly the same with the experimental values. Changing the turbulence model can also give significant variations in the predicted separations [33].

### 5.3 Optimization

The objectives of this optimization are to obtain a minimal drag coefficient of the airfoil with a maximal lift coefficient which leads to a maximum aerodynamic finesse ratio, implicitly given as below:

$$\begin{cases} \min\limits_{d.v} : & f_1 = C_d \\ \max\limits_{d.v} : & f_2 = C_l \\ \text{subject to} : & \mathrm{C_d} > 0 \end{cases} \tag{21}$$

It is required in all optimization problems to identify parameters that can be modified in order to reach the optimized solution. It is the geometry, in the case of the mesh morpher, that must be parameterized. The large variety of shapes available in engineering applications involves complications of the geometric parameterization for general shapes used in CFD. In order to minimize such complications in FLUENT simulation, the problem of shape parameterization is reduced to a problem of the parameterization of changes in the geometry.

The next essential requirement for mesh morphing is a tool that can smoothly alter the shape, irrespective of the underlying mesh topology. In FLUENT, this is accomplished using a free form deformation technique. This technique manipulates designated deformation regions via displacements applied to a set of control points. The mesh region that is to be deformed is defined by a "box" that is, a rectangle for 2D cases, and the control points must be located within the box. The displacements of the control points are the result of user-defined motions (each of which involves a parameter value and other directional settings) and these displacements are then applied to the mesh as a smooth deformation by either interpolating the displacement based on radial basis functions or using the tensor product of Bernstein polynomials [30].

The shape of the airfoil is defined by a spline. The shape parameterization is done with one parameter "Param" that prescribed the relative ranges of motion of the four select control points according to Figure 5. Two other input parameters are used for the optimization process that are the Mach number and the incidence as illustrated in Table 3.

#### 5.3.1 FLUENT and MOP coupling process

The flowchart in Figure 6 shows the multiobjective optimization process of CFD problems. The optimization work was entirely automated via a main script written in MATLAB. First of all, starting the Multiobjective optimization process begins with the script generation of an initial population using a uniform random distribution function. Then multiple sets of decision variables (i.e., multiple geometries) are submitted and wait for each individual objective functions values before proceeding further. For changing the input parameters to compute the aerodynamic responses taken as objective values of the new shape, MATLAB calls FLUENT software and then it returns back those values to the optimization process for executing the multiobjective algorithm and displaying the optimal results. The coupling MATLAB/FLUENT is compiled using FLUENT as a Server session that is very

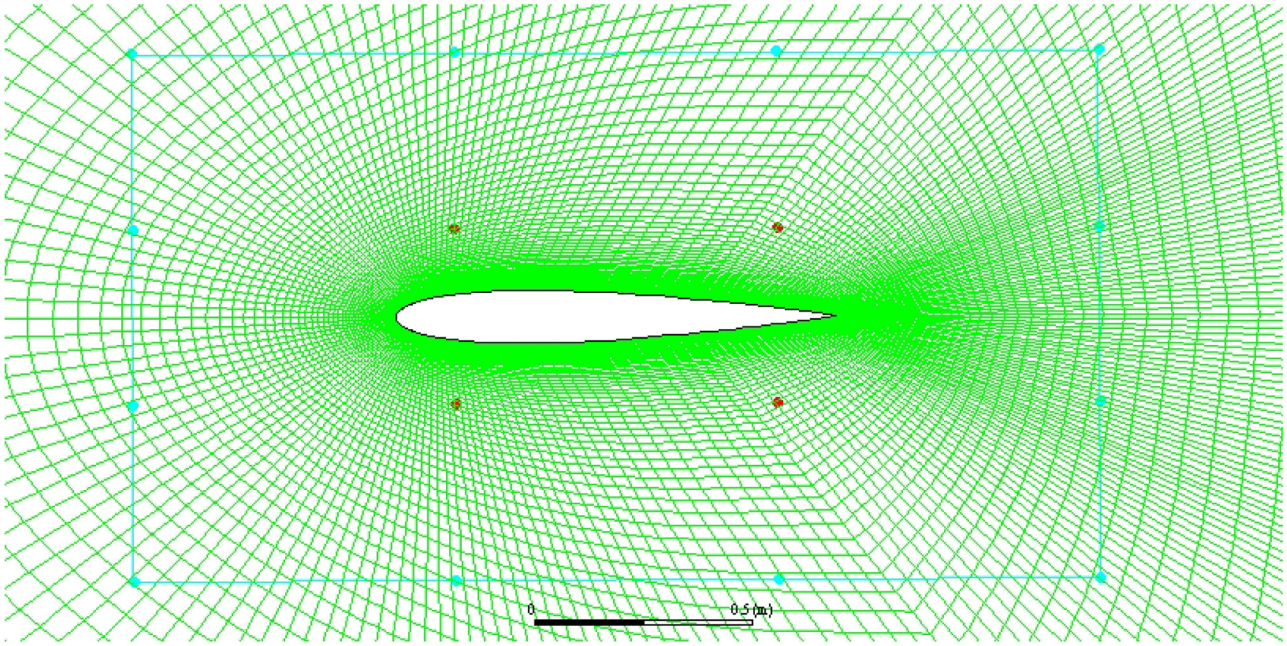


**Fig. 5.** Mesh morphing control points.

**Table 3.** Input variables of the airfoil.

| Design variable | Initial value | Lower bound | Upper bound |
|---|---|---|---|
| Mach | 0.7 | 0.3 | 0.8 |
| Incidence | 1.55 | 0 | 10 |
| Param | 0.2 | 0.1 | 0.5 |

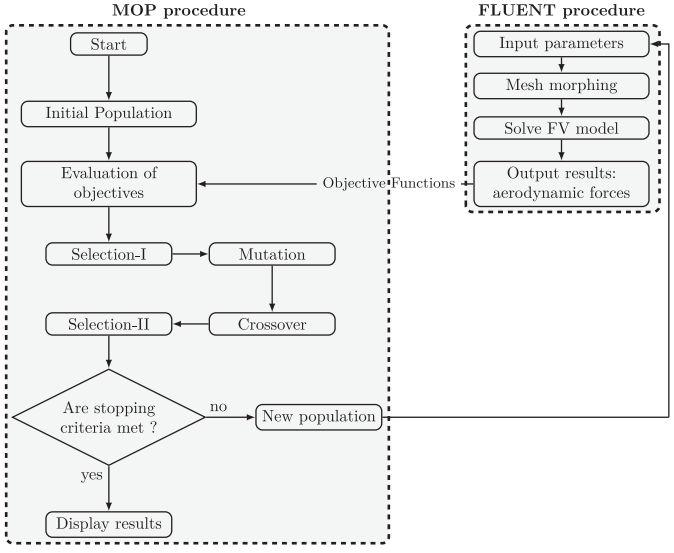


**Fig. 6.** Multiobjective optimization and FLUENT coupling process flowchart.

similar to a conventional standalone FLUENT session, but with the addition of an Internet Inter-ORB Protocol (IIOP) interface exposed that accepts connections from suitable client applications, which is MATLAB is

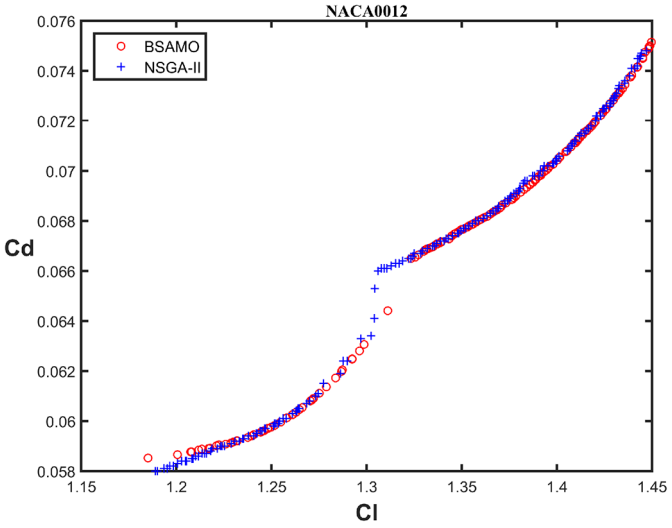


**Fig. 7.** Pareto solutions of the NACA0012 airfoil.

this paper. This capability is different from batch mode operation in that commands can be issued to the running session at any time rather than just from a predefined journal file. This allows solution steering and other manipulations without exiting from the FLUENT session.

### 5.3.2 Optimization results

The multiobjective shape optimization in this part is carried out for the airfoil NACA0012 by using BSAMO and NSGA-II algorithms coupled with a CFD solver according to the optimization process presented in Figure 6. For a population size and a maximum number of generations set respectively at 200 and 10, the Pareto fronts obtained for both algorithms (BSAMO and NSGA-II) are shown in Figure 7 and the results from Table 3 marked in bold

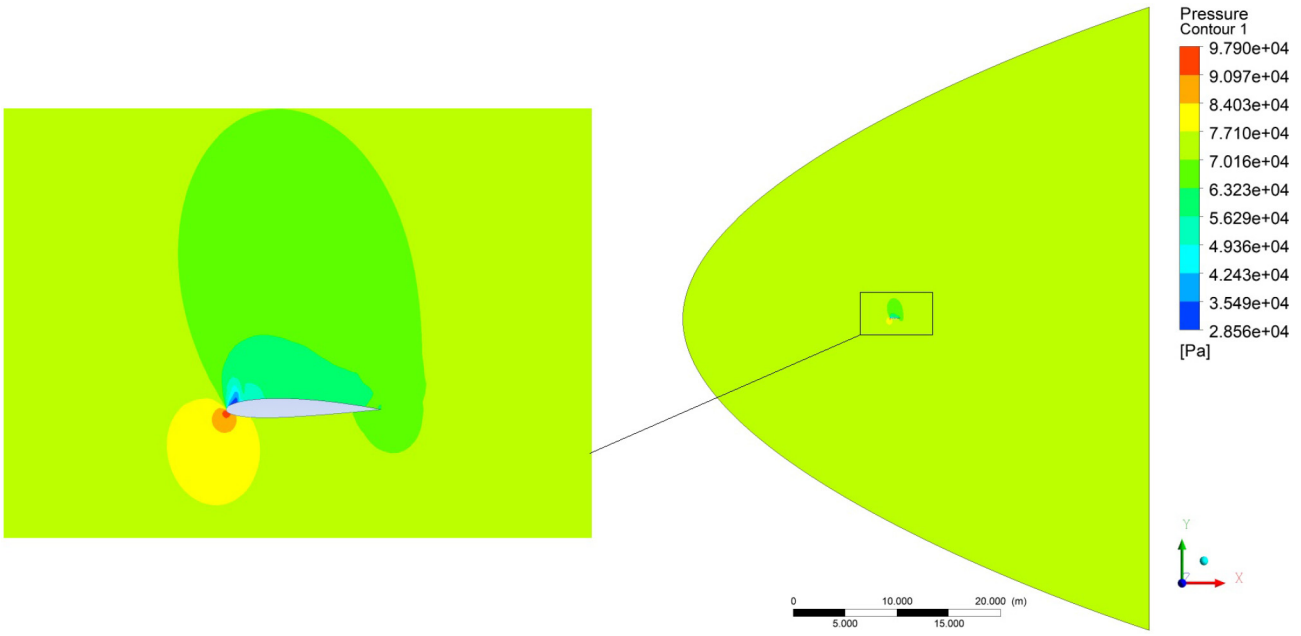


**Fig. 8.** Pressure contour of an optimized airfoil.

**Table 4.** Airfoil optimal results.

| | Design variables | | | Objective functions | |
|---|---|---|---|---|---|
| | Mach | Incidence | Param | $C_d$ | $C_l$ |
| BSAMO | 0.6728 | 10 | 0.1 | **0.0585** | 1.1851 |
| | 0.8 | 10 | 0.4113 | 0.0751 | **1.4496** |
| NSGA-II | 0.6920 | 9.5829 | 0.3217 | **0.0580** | 1.1889 |
| | 0.7978 | 9.9983 | 0.3849 | 0.0748 | **1.4470** |

present the best objective functions optimal values comparison found by the two algorithms for the airfoil.

Figure 8 displays the pressure contour of an optimized airfoil and non symmetrical compared with the original NACA0012 given in Figure 5. The freeform mesh morpher in FLUENT provides a powerful tool for arbitrary smooth changes in the geometry without being limited by a constrained parameterised geometry.

The results from Figure 7 and Table 4 indicate that the expansibility of the Pareto fronts distribution by BSAMO and NSGA-II is slightly the same when solving this optimization problem with a good quality distribution and approximation. Also for the two multi-physics simulations it obtains better optimal results for the objective functions.

## 6 Conclusion

In this paper a workbench project, including FLUENT, has been used to compute the transonic, compressible flow over a NACA0012 airfoil. The implicit density based solver with solution steering was employed and the computed results have been compared to published experimental data and good agreement was achieved. A case comparison has been carried out within CFD Post to compare the pressure fields at Mach 0.5 and Mach 0.7. Then the drag minimization and lift maximization study was carried out for the airfoil NACA0012 by using BSAMO and NSGA-II coupled with FLUENT. The more challenging Navier-Stokes computation was carried out to demonstrate the reliability and robustness of multiobjective optimization algorithms based on a main script written in MATLAB and its successful coupling with the CFD software. The free stream Mach number Ma, the angle of attack and the mesh morpher parameter were allowed to vary during the course of the optimization process. The convergence history of the computation was noted after about 2199 CFD calls, that the fitness value reaches its converged value. It should be mentioned here that the averaged fitness corresponds to the entire members in the generation and the maximum fitness corresponds to the best member in each generation.

The obtained results, compared to the NSGA-II algorithm, indicate that BSAMO is generally successful and competitive in dealing with complex and multi-physics systems. Therefore, we conclude that BSAMO offers a potential alternative solution for solving MOPs.

## References


1. K. Deb, Multiobjective optimization, in *Search Methodologies* (Springer, US, 2014), pp. 403–449
2. C.A.C. Coello, Evolutionary multiobjective optimization: a historical view of the field, IEEE Comput. Intell. Mag. **1**, 28–36 (2006)
3. A. Konak, D.W. Coit, A.E. Smith, Multi-objective optimization using genetic algorithms: a tutorial. Reliab. Eng. Syst. Safety **91**, 992–1007 (2006)
4. X. Yang, P. Deb, Multiobjective cuckoo search for design optimization, Comput. Oper. Res. **40**, 1616–1624 (2013)
5. J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archiv, IEEE Trans. Evol. Comput. **13**, 945–958 (2009)
6. D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J. Global. Optim. **39**, 459–471 (2007)
7. C.O. Ourique, E.C. Biscaia, J.C. Pinto, The use of particle swarm optimization for dynamical analysis in chemical processes, Comput. Chem. Eng. **26**, 1783–1793 (2002)
8. A. Bonilla-Petriciolet, J.G. Segovia-Hernandez, Particle swarm optimization for phase stability and equilibrium calculations in reactive systems. Comput. Aid. Ch. **26**, 635–640 (2009)
9. A.A. Salman, I. Ahmad, M.G.H. Omran, G. Mohammad, Frequency assignment problem in satellite communications using differential evolution, Comput. Oper. Res. **37**, 2152–2163 (2010)
10. J. Zhang, L. Xie, S. Wang, Particle swarm for the dynamic optimization of biochemical processes, Comput. Aid. Ch. **21**, 497–502 (2006)
11. K. Deb, A. Pratab, S. Agrawal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. **6**, 182–197 (2002)
12. F. Neri, V. Tirronen, Recent advances in differential evolution: a survey and experimental analysis, Artif. Intell. Rev. **33**, 61–106 (2010)
13. D. Karaboga, B. Akay, A comparative study of artificial bee colony algorithm, Appl. Math. Comput. **214**, 108–132 (2009)
14. C. Igel, N. Hansen, S. Roth, Covariance matrix adaptation for multiobjective optimization, Evol. Comput. **15**, 1–28 (2007)
15. H. Smaoui, A. Maqsoud, S. Kaidi, Transmissivity identification by combination of cvfem and genetic algorithm: application to the coastal aquifer, Math. Probl. Eng. 3463607 (2019)
16. P. Civicioglu, E. Besdok, A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms, Artif. Intell. Rev. **39**, 315–346 (2013)
17. R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. **11**, 341–359 (1997)
18. A.K. Qin, P.N. Suganthan, Self-adaptive differential evolution algorithm for numerical optimization, IEEE Trans. Evol. Comput. **3**, 1785–1791 (2005)
19. E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength Pareto evolutionary algorithm, Optim. Control Appl. Ind. Probl 95–100 (2002)
20. G. Maoguo, J. Licheng, D. Haifeng, B. Liefeng, Multiobjective immune algorithm with nondominated neighbor-based selection, Evol. Comput. **16**, 225–255 (2008)
21. P. Civicioglu, Backtracking search optimization algorithm for numerical optimization problems, Appl. Math. Comput. **219**, 8121–8144 (2013)
22. K. Guney, A. Durmus, S. Basbug, Backtracking search optimization algorithm for synthesis of concentric circular antenna arrays, Int. J. Antennas Propag. **2014**, 11 (2014)
23. J. Lin, Oppositional backtracking search optimization algorithm for parameter identification of hyperchaotic systems, Nonlinear Dyn. **80**, 209–219 (2015)
24. R. El Maani, B. Radi, A. El Hami, Multiobjective backtracking search algorithm: application to FSI, Struct. Multidiscip. Optim. **59**, 131–151 (2019)
25. R.R.A. Martins, A.B. Lambe, Multidisciplinary design optimization: a survey of architectures, AIAA J. **51**, 2049–2075 (2013)
26. A. El Hami, B. Radi, Fluid-Structure Interactions and Uncertainties: Ansys and Fluent Tools (Wiley-ISTE, 2017)
27. A. El Hami, B. Radi, Uncertainty and Optimization in Structural Mechanics (Wiley-ISTE, London, UK, 2013)
28. B.E. Launder, D.B. Spalding, Lectures in Mathematical Models of Turbulence (Academic Press, London, England, 1972)
29. F.R. Menter, Two-equation eddy-viscosity turbulence models for engineering applications, AIAA J. **32**, 1598–1605 (1994)
30. ANSYS, ANSYS Fluent Users Guide, 2018
31. G. Eggenspieler, Mesh Morphing and Optimizer (ANSYS, Inc, May 14, 2012)
32. P.A.N. Bosman, D. Thierens, The balance between proximity and diversity in multiobjective evolutionary algorithms, IEEE Trans. Evol. Comput. **7**, 174–188 (2003)
33. T.J. Coakley, Numerical Simulation of Viscous Transonic Airfoil Flows, NASA Ames Research Center, AIAA-87-0416, 1987
34. T.L. Hoist, Viscous transonic airfoil workshop compedium of results, AIAA Paper No. 87-1460, 1987
35. C.D. Harris, Two-Dimensional Aerodynamic Characteristics of the NACA 0012 Airfoil in the Langley 8-foot Transonic Pressure Tunnel (NASA Ames Research Center, NASA TM 81927, 1981)


**Cite this article as**: Rabii El Maani, Soufiane Elouardi, Bouchaib Radi, Abdelkhalak El Hami, Multiobjective aerodynamic shape optimization of NACA0012 airfoil based mesh morphing, Int. J. Simul. Multidisci. Des. Optim. **11**, 11 (2020)