Self-adaptive parameterisation for aerodynamic optimum-shape design

Alberto Clarich, Jean-Antoine Desideri

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Self-adaptive parameterisation for aerodynamic optimum-shape design

Alberto Clarich — Jean-Antoine Desideri

N° 4428

Mars 2002

THÈME 4

ISSN 0249-6399 ISRN INRIA/RR--4428--FR+ENG

INRIA
SOPHIA ANTIPOLIS

Self-adaptive parameterisation for aerodynamic optimum-shape design

Alberto Clarich * , Jean-Antoine Desideri †

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet Opale

Rapport de recherche n° 4428 — Mars 2002 — 23 pages

Abstract: In this report, we propose a technique to adapt the shape parameterisation of an aircraft wing in the course of an optimisation of aerodynamic performance. The M6 Onera transonic wing was chosen as a test case, in particular the objective of the optimisation was the minimisation of drag, keeping constant the lift and the total volume of the wing.
In a first step, the parameterisation of the cross sections by Bézier curves consisted on keeping constant the control points abscissa distribution and changing the ordinates, while, in further steps, the constant abscissa distribution was modified, in order to regularise progressively the Bézier control polygon.
Our numerical experiments have demonstrated that a self-adaptive parameterisation *(i)* permits to account for a larger set of feasible solutions, *(ii)* results in a quantitatively notably improved optimisation performance, and *(iii)* converges towards a non-intuitive distribution of control points, revealing the necessity and usefulness of self-adaptive algorithms.

Key-words: parameterisation, Bézier curve, optimisation, genetic algorithm, wing shape design, fluid-dynamics simulation

* PhD student, Energetics
University of Trieste, Italy
† Directeur de recherce, INRIA
Sophia Antipolis, France

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Paramétrisation auto-adaptative pour la conception optimale de forme aérodynamique

Résumé : Dans ce rapport, on propose une technique d'auto-adaptation de la paramétrisation de voilure au cours de son optimisation aérodynamique. Des conditions classiques d'écoulement transsonique autour de l'aile ONERA M6 sont utilisés comme cas-test d'optimisation de forme. Plus précisément, on réduit la traînée, en maintenant la portance et le volume de l'aile.
Dans une première étape, on a paramétré des sections de la géométrie 3D par des courbes de Bézier dont les points de contrôle ont des abscisses prescrites (uniformément distribuées) et des ordonnées optimisées. Dans les étapes suivantes, ces abscisses ont été redistribuées en fonction d'un critère de régularisation du polygone de contrôle. Les expériences numériques ont démontré que l'auto-adaptation de la paramétrisation *(i)* permet de prendre en considération un plus grand nombre de solutions compatibles, *(ii)* améliore notoirement le résultat quantitatif de l'optimisation, et *(iii)* converge vers une paramétrisation non-intuitive de la distribution des points de contrôle, revélant la nécessité et l'efficacité des algorithmes auto-adaptatifs.

Mots-clés : paramétrisation de forme, courbe de Bézier, optimisation, algorithme génétique, conception optimale de voilure, simulation numérique en mécanique des fluides

1 INTRODUCTION

Regardless the field of application, an optimisation procedure, and particularly an optimum-shape design based on genetic algorithm [1],[2],[3], is characterised by one or more objectives to be reached, and by a parameterisation technique.
The parameterisation directly affects the object of the optimisation, for example, in aerodynamic design, it may be the shape of one or more wing cross-sections, so that every individual (so is called every different configuration obtained by the parameterisation) may be confronted with the others, on the basis of how much it satisfies the objective that is pursuited.
So clearly, any optimisation procedure, even very efficient, is strongly influenced by each particular parameterisation strategy adopted. In particular, the objective improvements may be limited by a parameterisation that does not allow a complete exploration of the design space of the individuals.
The parameterisation reduces the exploration space to a finite dimension, and it is desirable that this limitation be as little restrictive as possible.
Intuitively, we would like the range of the parameterisation (i.e. the set of functions exactly represented by it) to contain elements as close as possible to the continuous optimum function, which of course is unknown. This 'functional approximation' property is difficult to achieve rigorously, but numerical optimisation algorithms should be constructed to adapt the parameterisation during the course of convergence in view of an improved satisfaction of this goal.
A common practice is to start with a limited range of variation for the variables defined by the parameterisation and then to change it during the course of optimisation, according to the direction of potentially improved results. In any case, when the genetic algorithm seems to converge to a unique solution (or to a Pareto front in the case of several objectives), the algorithm would experience difficulties to explore other eventually better solutions. So, in this case particularly if a satisfactory solution has not been achieved, it is necessary to change the parameterisation, and restart the optimisation keeping into account the best individuals of the previous step.
In this report, we provide a preliminary demonstration that self-adapted parameterisation provide significantly improved optimisation results, in the case of a (3D) wing design. We have chosen for this demonstration, a classical test-case [4].
In this case the parameterisation was obtained by changing the ordinates of Bézier curve control points, having chosen the Bézier curves to define the wing sections shape. The limit of the parameterisation is due to the fact that the abscissa positions are fixed (otherwise, the number of design parameters would twice as large, and this would make the optimisation process stiffer), and it is easy to understand how, in that way, we would not be able to explore all the infinite curves we may obtain. So, at the end of a first optimisation step in which the abscissa are fixed and uniformely distributed, we decide to change the abscissa positions, and repeat this operation for more steps, until we are not more able to improve the objective, in this case the drag reduction. The parameterisation has been here adapted to increase the regularity of the Bézier control polygon (see section 3.3).

2 TEST CASE AND MESH DESCRIPTION

2.1 Geometry of the M6 Onera wing

The M6 Onera wing [4] is a swept back wing produced in the 1972 by the ONERA AeroDynamics Department, with the purpouse to study three-dimentional flows characterised by high Reynolds number and transonic conditions. In our test, we adapted this wing geometry to prepare the numerical model, as we were interested in reducing the drag resistance in transonic flow conditions. In figure 1, the wing geometry is depicted.

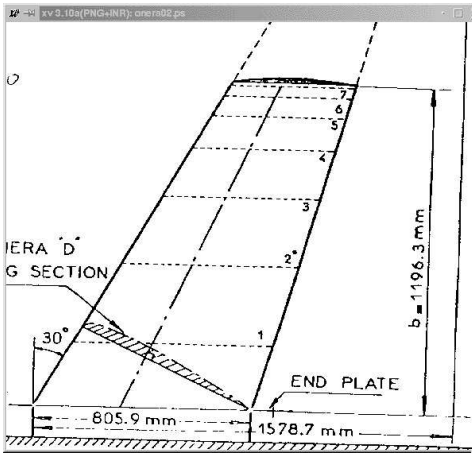

Figure 1: Geometry of the M6 Onera wing

The main chord at the root is 805.9 mm, the swept angle is 30° in the leading edge and 15.9° in the tailing edge, while the semi-span length is 1196.3 mm. The profile sections are defined ortogonally to the axis placed at 40.19% of the root section, and the geometry of these sections is expressed in terms of x and y co-ordinates relatively to the chord length of each of them. The orthogonal sections are symmetric with respect to the chord.
To construct the geometry, we have calculated the co-ordinates of the root and tip cross sections only, other sections being obtained by linear interpolation.

Flow conditions were defined as follows:

Reynolds number	$1.1 \cdot 10^7$
Mach number	0.84
angle of flow incidence	3.06°

2.2 Computational mesh

To simulate the flowfield around the wing, we have realised a multi-block structured mesh, using the fluid-dynamic code StarCd of the Computational Dynamics [5].
The geometry of the root and tip sections are read by StarCd in a text file, in which we define the co-ordinates of each section points, then some splines (polynomial curves) are created interpolating these points. Between these splines and the domain boundary lines (placed to at least 4 airfoil profile chords away from the wing in order to represent the undisturbed flow conditions), we create a total of 27 contiguous blocks that fill this space. These blocks are defined in a parametric way in terms of the number of airfoil section points. If the sections geometry is changed, the blocks shapes are automatically updated. Using this large number of blocks it is easy to define in each direction the cell distribution that is most appropriate to refine the mesh in the proximity of the boundary layer of the wing and to relax it in the direction of the domain boundaries. A final operation of smoothing is made to give the refinement differences between two contiguous blocks less evident.
In figure 2, the principal region of the multi-block structure is shown, with a detail of the wing surface and a transversal section mesh. The total number of cells is more than 120,000, as previous tests on some more refined meshes have not revealed significantly different flowfields.

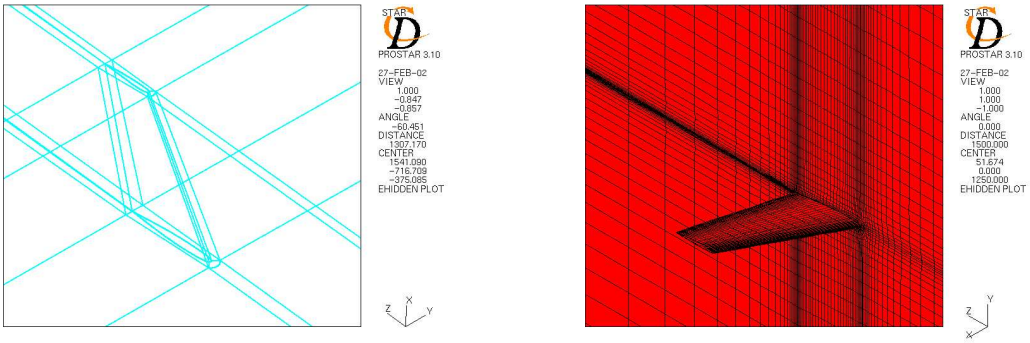

Figure 2: Mesh blocks and mesh surface details

At the inlet boundary we have defined a flow velocity V of 288 m/s, a static temperature T of 257 K and a density ρ of 0.866 kg/m^3 all constant, while on the other domain boundaries we have defined a constant pressure P equal to 63830 Pa, accordingly to the flow Mach number set to 0.84 and to the isoentropic one-dimensional theory [6], of which we report below the most important formulas:

$$V = M\sqrt{kRT} \tag{1}$$

$$T = T_0[1 + \frac{k-1}{2}M^2]^{-1} \tag{2}$$

$$P = P_0[1 + \frac{k-1}{2}M^2]^{\frac{1-k}{k}} \tag{3}$$

$$\rho = \rho_0[1 + \frac{k-1}{2}M^2]^{1-k} \tag{4}$$

In this case the subscript index $_0$ indicates the standard condition of temperature, pressure and density, while R is the constant of perfect gases and k is the air isoentropic coefficient (ratio of specific heats C_p/C_v) set to 1.4. The Mach number M is set to 0.84 as defined by the test-case conditions.
The flow solver is based on the Navier-Stokes equations, the fluid is considered compressible and no turbulence model is applied, in order to avoid an excessive calculation effort.
The simulations were run on a bi-processor Linux machine, Red Hat release 6.1, and to simulate each different wing configuration we give, as initial input, the original M6 Onera wing flowfield calculated once.
To simulate the original wing flowfield it was necessary to start the simulation with a low Mach number, and then to increase it progressively taking as initial input for each new simulation the previous flowfield, otherwise, no convergence may be obtained at high Mach numbers. Then, for any new wing configuration, each simulation takes less than 10 minutes. In this way, running two processes simultanously on the bi-processor machine, we can simulate about 300-400 individuals in a day time.
StarCd provides a tool to calculate directly the total lift and drag coefficients, which are expressed in the following formulas.

$$C_l = \frac{F_l}{1/2\rho SV^2} \tag{5}$$

$$C_d = \frac{F_d}{1/2\rho SV^2} \tag{6}$$

In these formulas, V and ρ are the velocity and density of the undisturbed flow, it means the values calculated by formulas (1) and (4) and assigned to the boundary domains. S is the reference surface of the wing, calculated by the semi-span length of the wing times the root main chord, while F_l and F_d are the total force components normal to the flow direction and parallel to it respectively, obtained by the integration of pressure and viscous forces on the wing surface.

3 PARAMETERISATION AND OPTIMISATION STRATEGY

3.1 Programs for the parameterisation

A Bézier curve [7],[8] is a continuous and smooth curve defined by the abscissas and ordinates of its control points, which may be connected as a polygon. [1]

$$x(t) = \sum_{i=0}^{n} [x_i \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}] \tag{7}$$

$$y(t) = \sum_{i=0}^{n} [y_i \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}] \tag{8}$$

Generally, to reproduce an airfoil profile through a Bézier curve [9] (one for the upper side and one for the lower), the first and second control points are placed on the y-axis, because in that way we force the curve to have a vertical tangent in the origin, as this round nose shape is better represented in subsonic and transonic conditions [6].
The first and the last points are then also placed and fixed on x-axis, while the positioning of the other points is free.

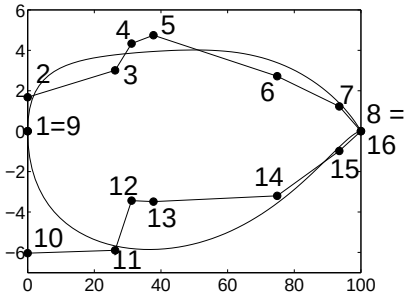

Figure 3: Example of two 7-degree Bézier curves and of their control polygons applied to define an airfoil profile

Note that a given Bézier curve admits an infinite number of associated control polygons. In fact, by degree elevation, the curve may be sought as the limit of control polygons of increasing number of edges [10]. As a remark, if the control polygon is convex, also is the curve itself; however the converse results not true.
To define in a parametric form the sections of the Onera wing, we have used three different programs, that we describe in the following paragraphs.

[1] In the formulas 7 and 8, n is the degree of the Bézier curve ($n+1$ is number of the control points), while x_i and y_i are the co-ordinates of the control points.

3.1.1 INGEOM program

INPUT FILES:	onera.txt
OUTPUT FILES:	shape_1/2.dat

This program is used to reproduce the original geometry of the M6 Onera wing. It reads from the *onera.txt* file the x and y dimensionless co-ordinates of the profiles orthogonal to the main axis of the wing, placed at 40.19% of the root chord as described in section (2.1). Then, for the root and tip cross sections, it calculates the corresponding co-ordinates of the points, and writes them in the two files, *shape_ 1.dat* and *shape_ 2.dat*, corresponding to the two sections. These co-ordiantes are still expressed relatively to the chord length, thus, the abscissas vary from 0 to 100, and the ordinate is expressed as a percentage of the section chord length.
In this way, we produce the section points necessary to define completely the geometry of the original M6 Onera wing, as the sections from root to tip are interpolated linearly.

3.1.2 BEZIER program

INPUT FILES:	xcurve.dat, xbez.dat, ybez.dat
OUTPUT FILES:	ycurve.dat

This program is used to calculate the points ordinates of any Bézier curve, whose control points are defined in the files *xbez.dat* and *ybez.dat*. Reading the x_i and y_i co-ordinates of these points (see formulas 7 and 8), it is in fact possible to define the $y(t)$ ordinate corresponding to the abscissa $x(t)$ read in *xcurve.dat* (in particular, for each value of x we find the corresponding t, and then we obtain the corresponding y), and then to write the ordinate in the output file.

3.1.3 CREATECURVE program

INPUT FILES:	xbez_1/2.dat, var
OUTPUT FILES:	section_new.1/2, beziernew.1/2

This is the main program used for our parameterisation. At first, it reads from the file *var*, produced in the optimisation cycle for every new configuration, the 24 variables that define each new wing configuration.
The first 12 are relative to the root section, while the remaining to the tip section. Each of the two groups is then again divided into two subgroups, so that we have six variables to define each of the two half-sides of the section (upper and lower surfaces). These six variables represent the ordinates of the six control points free to move, the abscissa being fixed. In fact, to represent each side of the sections, we have chosen an 8-control points Bézier curve (fig.3), but as the first and the last point are fixed on the x-axis, we have effectively six design free parameters for each side.
Then the program runs a loop two times, one for each section. It puts the six variables that define the upper side of the section to the file *ybez.dat*, it copies the file *xbez_ 1.dat* or *xbez_ 2.dat* to *xbez.dat* (depending on if we are considering the control points abscissa

distribution of the root or tip section, defined respectively in those input files), and then it calls the *BEZIER* program, in which results a unique *xcurve.dat* file. Once the *ycurve.dat* file containing the Bézier curve ordinates is produced, the program reads it and repeats the same operations for the lower side of the profile, changing the variables in *ybez.dat*. Once all the new points of the section are calculated and read, they are written in an output file, *beziernew.1* or *beziernew.2* depending on which section we consider, and the same operation is repeated for the other section.
At this point, we calculate the real 3D co-ordinates of the sections, multiplying the Bézier co-ordinates for each section chord length, and considering the translation due to the swept angle of the wing and the rotation due to the angle of incidence of the wing with respect to the flow direction. The x-y-z co-ordinates are written in the two files *section_new.1* and *section_new.2*, that become the input files for the StarCd mesh writer, as described in section (2.2).

3.2 Primary optimisation

In each primary optimisation step, we represent the 3D wing sections with 24 variables, as described in section 3.1.3, and these variables represent the ordinates of Bézier control points, the abscissa distribution being fixed for each of the two sections. To set up and run the optimisation step we have implemented the Frontier environment [11]. Frontier is a platform-independent optimisation tool, which allows to define any particular optimisation case and to run the optimisation in batch mode, using different type of algorithms. In figure 4 we depict the environment set up for our optimisation.

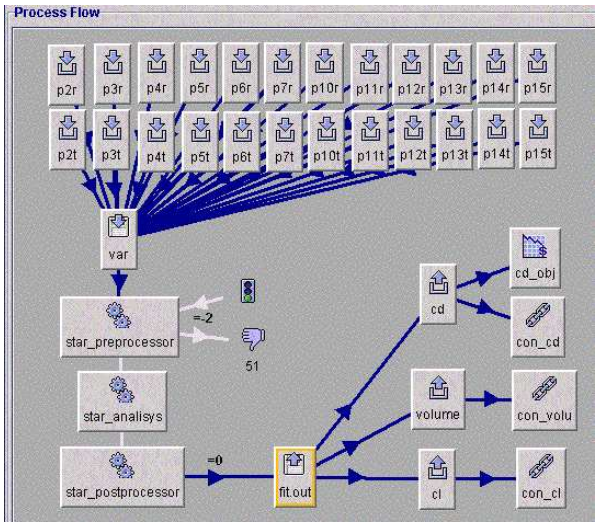

Figure 4: Frontier optimisation environment

The 24 variables are represented at the top of the figure, each of them requires setting a range of variation. For each configuration proposed during the optimisation, its variables are written in the file *var*, represented by the first icone with a disk. Then a first application is run, *star_preprocessor*, which reads the input files, runs the *CREATECURVE* program (section 3.1.3), and then runs the StarCD pre-processor program in batch mode. StarCD reads a macro in which all the commands needed to define the wing geometry and mesh are defined, and just the section points co-ordinates change accordingly to the *section_new.1* and *section_new.2* files produced by the *CREATECURVE* program.
At this point a second application, *star_analysis*, is run, and its function is to execute in batch mode the StarCD numerical analysis. If the calculations are converged and no error file is produced (as would occur, for example, when the wing geometry is so unsatisfactory that the mesh is distorted), a last application, *star_postprocessor*, is run. The last one is used to calculate the drag and lift coefficient on the wing surface, and to calculate the total volume of the wing. All these data are written in the text file *fit.out*, which is represented in fig.4 by the last icone with a disk.
We can see that from this icone there are three arrows directed to three other symbols, that represent the output variables, c_d, c_l and *volume*. For each of them we define a constraint, represented by a chain, and only for c_d we define also an objective. In table 1 below we summarize the constraints and the objective of the optimisation, the constraints being derived from the original Onera wing output variables.

Table 1: Objectives and constraints of the optimisation

1^{st} constraint	$c_d \leq 0.0485$
2^{nd} constraint	$c_l \geq 0.0244$
3^{rd} constraint	$\frac{\lvert volume - 3.715 \cdot 10^7 mm^3 \rvert}{3.715 \cdot 10^7 mm^3} < \epsilon, \quad \epsilon = 0.005$
objective	minimise c_d

Of course we impose that the drag should be not greater than the original one and the lift must not be less than the original one, but this is not sufficient. In fact, it may happen that the wing sections become very small during the optimisation and although this may result in acceptable aerodynamic coefficients, we want to avoid this situation that could cause structural problems due to the pressure stresses on the wing. For this reason we force the relative difference of the total wing volume (*volume*) with respect to the original one ($3.715 \cdot 10^7 mm^3$) to be less than a fixed tolerance ϵ.
During the optimisation, a classic genetic algorithm is used. The probability of directional cross-over is set to 0.85, while the selection and mutation one is set to 0.05. Due to the large number of variables, we have set the number of individuals for each generation to 40, while the constraints have been treated with some penalty functions.

3.3 Secondary optimisation

As discussed in the introduction, at the end of a first step in which a classic genetic algorithm is used, considering as variables of the optimisation the Bézier control points ordinates and the abscissas being fixed, we have to perform a secondary (static) optimisation, whose objective is to regularise the Bézier control polygon that define the best configuration obtained in the first step.
As mentioned in section 3.1, a given Bézier curve may be sought as the limit of a sequence of control polygons of larger and larger number of edges ('degree elevation'). Starting with a very irregular polygon (made of a small number of edges), this convergence process has the effect of a regularisation, since in the limit, the curve itself is achieved.
Our intuition is that very irregular polygons correspond to the early stage of this convergence. Thus, if we can replace a polygon by one of equal number of edges (equal degree), but 'more regular', we anticipate a convergence rate improvement, or conversely, a better approximation of the smooth optimum shape, for a given degree. Our experiment aims at confirming this intuitive claim.

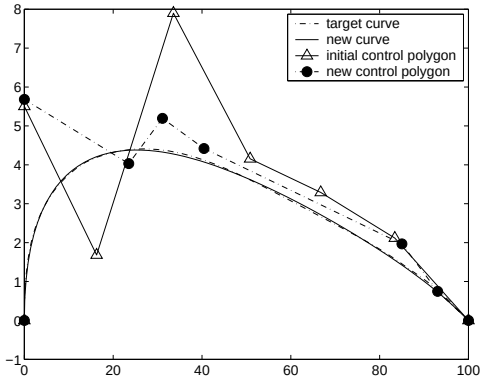

Figure 5: Example of regularisation of the control polygon of a Bézier curve

In figure 5 we see an example of a target Bézier curve obtained by a former abscissa distribution (uniform) of the control points, in particular we can notice how the Bézier control polygon is very irregular. The new curve obtained by a new abscissa distribution is characterised by a slight difference of its points ordinates (that we can express by the L_2 norm (eq.10) forced to be less than a fixed tolerance ϵ [13]) and by a higher regularity of its control polygon, measured by the *Total Variation* TV (eq.9).
Thus, in this secondary optimisation, the variables considered are both the abscissas and the ordinates of the Bézier control points, while the objective of the optimisation is the

minimisation of the total variation TV with a constraint on the L_2 norm [2] , as we want to reproduce the original curve with the lowest error possible.

$$TV = \sum_{i=1}^{n} |y_i - y_{i-1}| \tag{9}$$

$$L_2 = \sqrt{\sum_{i=1}^{N} (y_{Ti} - y_i)^2 \frac{x_{Ti} - x_{Ti-1}}{2}} \tag{10}$$

In the following section we describe the programs we have used for this optimisation.

3.3.1 EXAM1.FOR and OPT4.FOR programs

The main program we use to perform this optimisation is called *EXAM1.FOR*[13]. As input files it considers the *beziernew_1.dat* and *beziernew_2.dat* files that describe the sections geometry of the best configuration obtained during the primary optimisation last step and considered as target one, and the file *polg.dat* that defines the abscissa distribution considered in that step, and used here as initial point for the optimisation. A gradient-based method [12] is used to obtain the convergence of the fitness function TV. Beside the non-linear constraint on the L_2 norm, there are other n linear constraints on the abscissa variables, n being the degree of Bézier curve, as we have to force the abscissa of each control point to be greater or equal to the previous one. All the information about the constraints are given by the *oput.dat* file. The optimisation is repeated independently for both the two sections, whose new abscissa distribution may then be different.
Once the new abscissa distributions are obtained, they are written to primary optimisation new step files *xbez_1.dat* and *xbez_2.dat* and then the *TRANSLATE* program is run. The function of this program is to define the initial generation for the primary optimisation next step. In fact we need to conserve the information obtained in the last primary step, but the variables that describe the best individuals of that step now are changed, as the Bézier control points abscissa are different.
This program has then to read the design variables of the last primary step best individuals (from the file *best.des*), reproduce their sections geometry accordingly to the old abscissa distributions, and run for each individual another program, *OPT4.FOR*, whose function is to provide the design variables for the new abscissa distribution. In other words, it performs a third-level optimisation, in which the variables are just the Bézier control points ordinates and the objective is to minimise the L_2 norm function, each last step individual being a new target curve. Also for this third-level optimisation a gradient-based method is used, and due to the few number of variables and to the low cpu-demanding fitness function, the results are obtained immediately for each individual.

[2] In eq.9, n is the curve degree and y_i is the control point ordinate, while in eq.10 N is the number of points of the discretized Bézier curve and with $_{Ti}$ we denote the abscissas and ordinates of the target curve, that is the best configuration of the previous primary optimisation step.

In particular, we set the *TRANSLATE* program in order to reproduce, for the 70-80 % of the next primary step first generation, the best individuals of the last primary step, and to generate randomly the other indiduals, to avoid a premature convergence. The new design variables are written to the file *star.des*, that will be read by *FRONTIER* as initial generation for the primary optimisation next step. However, it is important to note that the individuals that are reproduced are not exactly the same ones of the previous step, as the L_2 norm is not necessary equal to zero, so it will not be surprising if the fitness of these individuals are slightly different, perhaps corresponding to less efficient designs. But what is really important to obtain, after an old primary step and before a new one, is just the modification in the abscissa distribution, because if the individuals are better represented by the Bézier curve, i.e. with a more regular polygon, they will have more chances to improve their fitness function in the course of the next step in the optimisation.
In the next chapter we show how the primary and secondary optimisation have been alternated in subsequent steps, and the results obtained prove the usefulness of this approach.

4 RESULTS OF THE OPTIMISATION

4.1 The optimisation process

In figure 6 we show a logic flow-chart that summarizes how the primary and the secondary optimisation procedures have interacted in alternating steps.
As explained in the previous chapter, in the primary optimisation step, *FRONTIER* reads

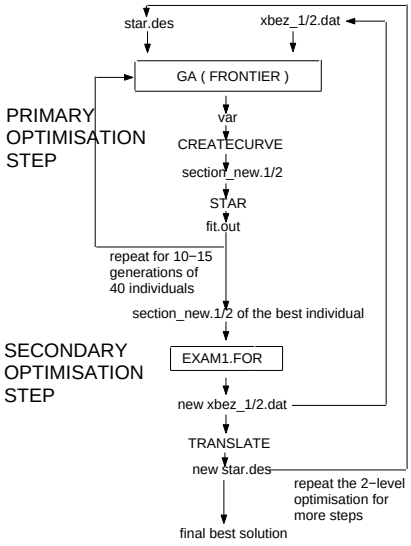

Figure 6: Flow-chart of the optimisation procedure

in *star.des* the first generation designs and in *xbez_ 1/2.dat* the initial Bézier abscissa distribution, it produces for every individual the file *var* containing the Bézier ordinates, and by the *CREATECURVE* program it writes the sections geometry in the files *section_ new.1/2.* StarCD reads these files, creates the mesh, runs the simulations and produces the post-processing file *fit.out* that defines the fitness of the individual.
In the first optimisation step, we have considered a number of generations equal to 15, as we wanted to be sure that by this classic approach there would not be any further improvement, while in the following steps the number of generations was fixed to 10. When the primary optimisation step is completed, the best individual sections geometry is given to *EXAM1.FOR* program, that performs the secondary optimisation step giving as output the new abscissa distributions that minimize the total variation of the Bézier control polygon respecting the target curve under a certain tolerance. A new primary optimisation step is then performed, after obtaining a new initial generation of designs by the program *TRANSLATE*, that reproduces for 70-80% the last step best configurations using the new abscissas. A total of four primary optimisation steps were performed, and in figure 7 we can see the drag evolution history. In the first image we consider, for each of the 45 generations, the

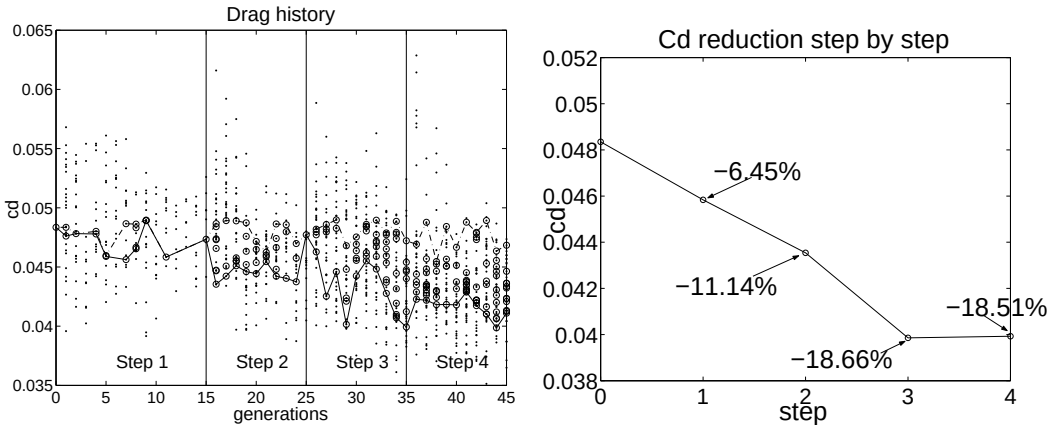

Figure 7: Drag evolution history

fitness value of all the configurations simulated. In particular, the unfeasible designs, that is those which violate one or more constraints, are represented by points. When the geometry corresponding to a given design is so distorted that no fitness value is produced by the analysis code, a case considered as 'error design', no symbol appears on the plot. Instead, the circles denote feasible values. The set of best (resp. worst) feasible design circles are connectd by a solid (resp. dashed) line.
We can immediately note two important facts. One is that, step by step, the best feasible values of the fitness are improving, and it is more clear in the second image, in which we report for each step the percentage of fitness improvement. After the first classical optimi-

sation step of 15 generations, the reduction of the c_d drag coefficient was limited to 6.45 %, while after the third or fourth step the reduction is of more than 18 %.
The other fact is that, while in the first step there is a large number of unfeasible or even error designs, in the following steps the number of feasible designs is largely growing.

This confirms our hypothesis: by a parameterisation realising a better fitting of the sections geometry, i.e. producing more regular Bézier control polygons, it is possible to explore the search space more effectively, and this has the effect of increasing the percentage of feasible individuals in each generation, that is a fundamental condition for the algorithm efficiency.

In fact, if we had continued the first optimisation step without changing the parameterisation, it is clear that probably we wouldn't have been able to improve very much the fitness, because of the very high number of unfeasible and error designs.
In conclusion, this experiment confirms that a self-adaptive parameterisation effectively enlarges the search space, and this permits improved optimisations.

4.2 Comparison of the best solution with the original one

In figure 8 below, we compare the original M6 Onera root and tip sections geometries with the best configuration obtained in the optimisation, while in table 2 we compare the performances of the two configurations, relatively to the drag and lift coefficients and to the total wing volume.

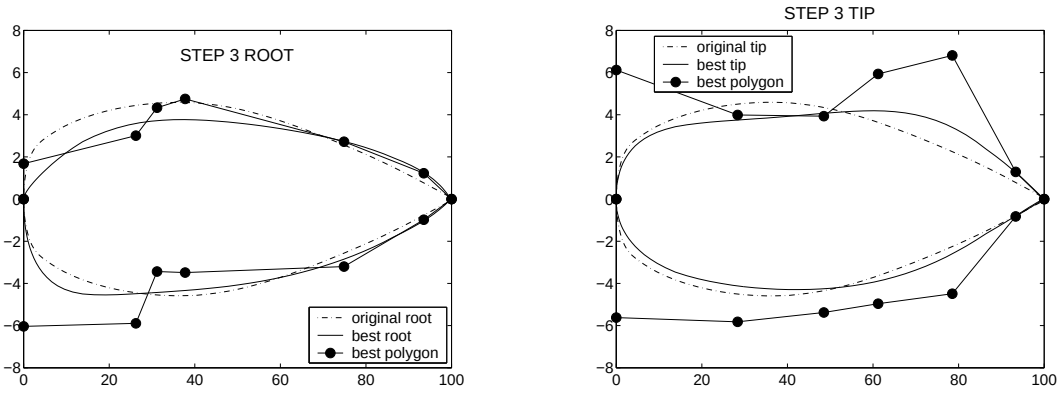

Figure 8: Comparison of the root and tip sections in the original and best configurations

As we can see, the improvement on the drag reduction is about 19 %, while the other constraints have respected the fixed tolerances. In figure 8 the Bézier control polygons that define the best root and tip sections are also indicated. We can see how the root section is characterised by an edge part which is bended downward and by a trailing part which is slightly thicker, while the tip section is importantly thicker in the trailing part and thinner

Table 2: Comparison of the performances of the original and best configuration

	original M6 wing	best configuration
c_d	$4.84 \cdot 10^{-2}$	$3.99 \cdot 10^{-2}$
c_l	$2.44 \cdot 10^{-1}$	$2.46 \cdot 10^{-1}$
volume	$3.71 \cdot 10^7 mm^3$	$3.71 \cdot 10^7 mm^3$

in the edge and middle part. In figure 9 below we reproduce the pressure distribution on the root and tip sections of the two configurations. The high peak of pressure at the tailing point, especially in the tip sections, may be due to the sudden reduction of section thickness in the last part of the tailing edge, as in the subsonic regions this is always followed by a speed reduction and a pressure increment [6]. On the other hand, the reduction of the drag force may be justified by the fact that, in the best configuration, the mean values of pressure are higher than in the original one, so the Mach number is generally lower (eq. 3), and in fact the viscous resistance is a direct function of Mach number, even though most of the drag resistance is however due to the pressure force.

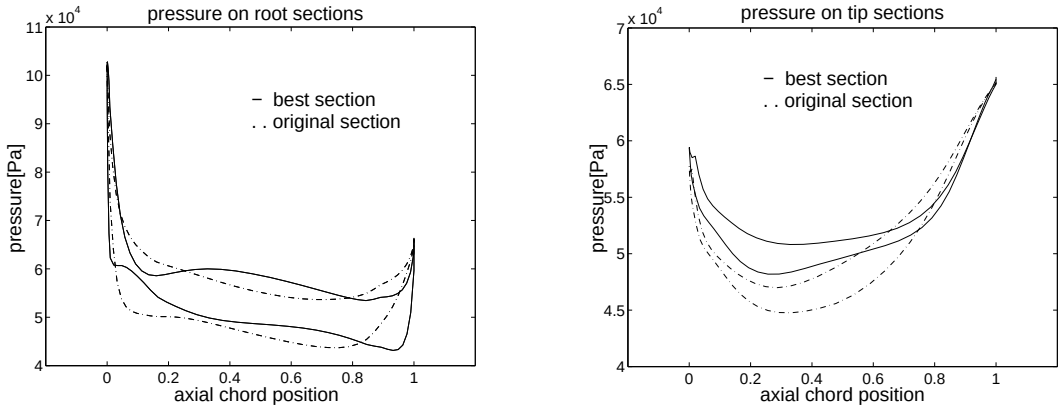

Figure 9: Comparison of the root and tip pressure distributions in the original and best configurations

In figure 10 we compare the pressure fields of the two configurations relatively to different sections. On the left column we have the original geometry and on the right column the best one. In the first row we have an isometric view and a root section normal to the span-wise x-axis, while in the second row there is a tip section. In the third row there is a section normal to the flow-wise z-axis and in the last row a section normal to the trasversal y-axis. From all the comparisons, we see clearly that the regions of low pressure, especially in the upper part of the profiles but also all around the tip, are very much reduced in the best configuration.

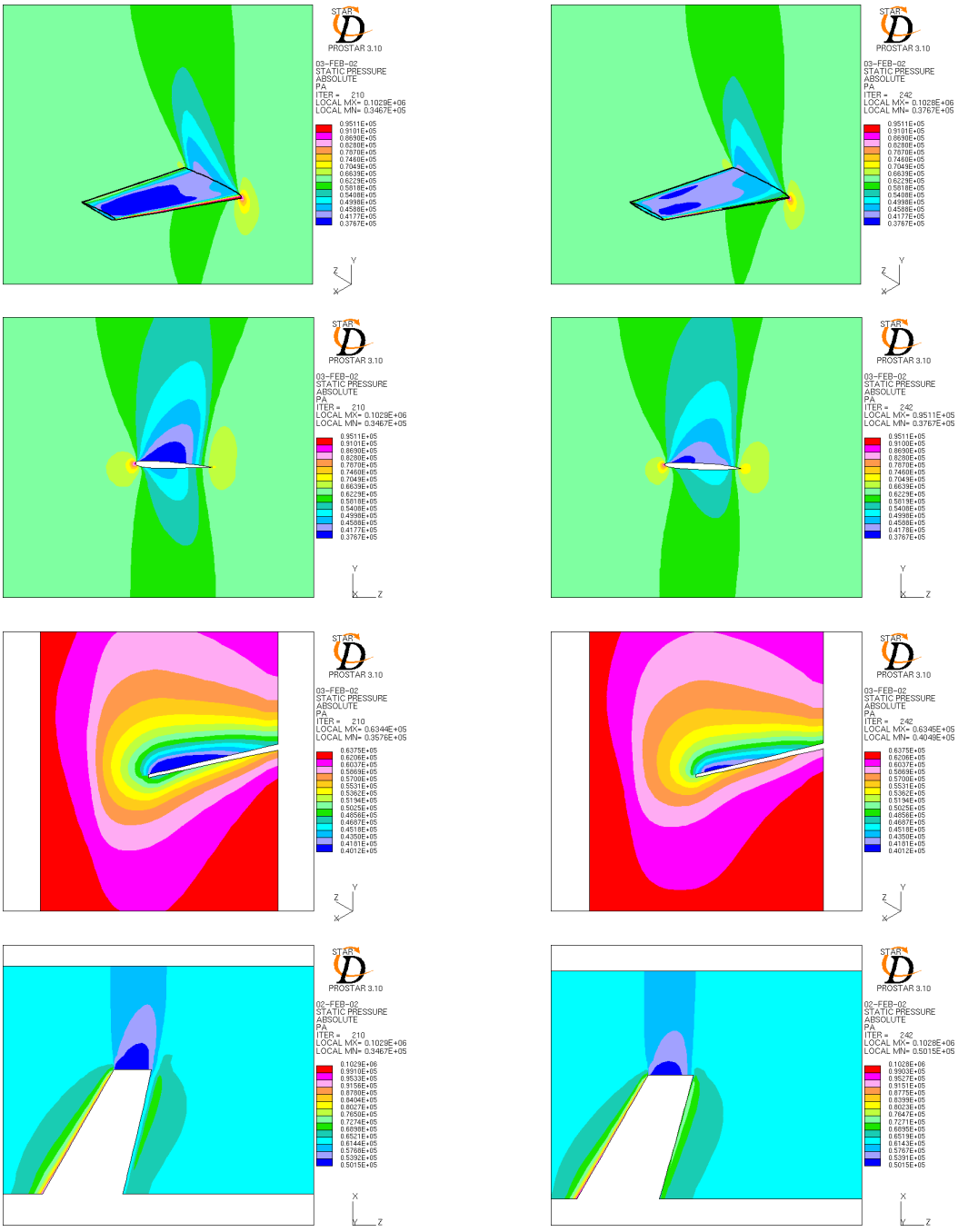

Figure 10: Pressure fields on original (left) and best (right) Onera wing configurations

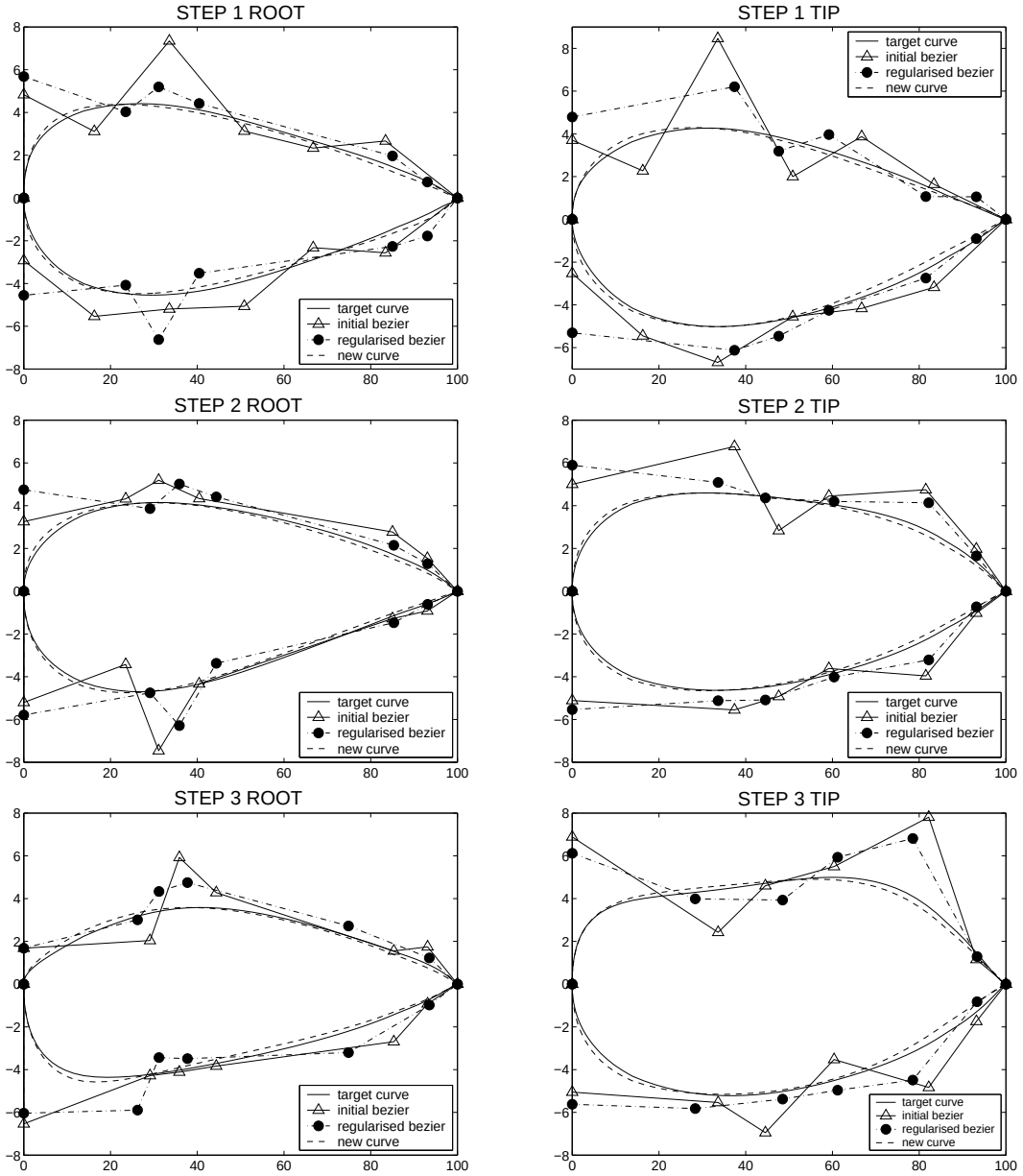

Figure 11: Bézier control polygon before and after every secondary optimisation step

In figure 11 we show the results of the secondary optimisation procedure at the end of each primary step. We compare the control polygon which defines each step best configuration before and after the regularisation. The better regularity of the new polygons is very evident, especially after the first step, while in the last one it does not seem to change very much. For this reason, and also because the fitness did not improve much in the last step, we have not continued the optimisation with further steps.

4.3 Further optimisation of control points abscissas

At the conclusion, we have thought to perform a last different optimisation step. This time, we reversed the roles played by abscissa and ordinata of control points: holding the ordinata fixed, we optimise (with respect to drag reduction) the abscissas.
This approach is done just as a local refinement of the best solution, as the variation in geometry allowed by this method is not very large, and of course we need not explore solutions that are completely different. However, also this time it was possible to improve the fitness: in figure 12 we can see the final geometry obtained confronted with the step 3 best one. Table 3 summarizes the performances relative to the drag coefficient of these two configurations and of the M6 Onera original one. In this last step we have performed 2 series of 10 generations of 30 individuals each one, we have just modified the range of variations for the variables after the first series according to the direction of the best designs variables. After the second series the variables and the fitness function seem however to converge, so we didn't change any more the parameterisation.

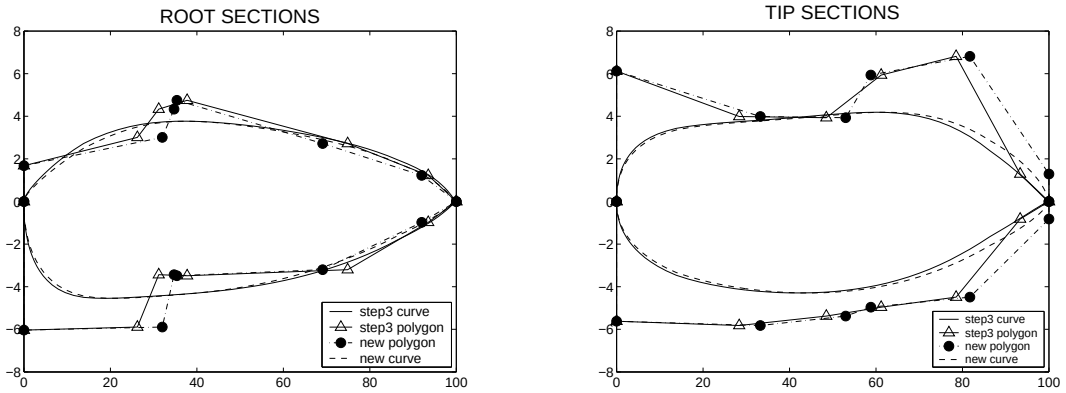

Figure 12: Comparison of the best root and tip sections after step3 and after ascissa adjustments

Table 3: drag coefficient for step 3 and abscissa adjusted best configurations

configuration	c_d	% improvement
original M6	$4.84 \cdot 10^{-2}$	0
best step3	$3.99 \cdot 10^{-2}$	18.7%
best abscissa adjusted	$3.81 \cdot 10^{-2}$	22.2%

4.4 A last remarck

Observing further Figures 8-11-12, it appears that the best parameterisation, or should we say 'improved' parameterisation, corresponds to non-intuitive location of abscissas.
In particular, we observe 3 control points very close (near x=0.3) perhaps to counterbalance the effect of the first two points at x=0 necessary for a vertical tangent (infinite rope at leading edge). In the rear portion of the airfoil, the control polygon is smoothly closer to the curve, perhaps because the geometry is not complex at the tailing edge (x=1).
This observation confirms that further efforts should be made to better understand the parameterisation, since non-intuitive adapted parameterisations results in significant impovement of performance.

5 CONCLUSIONS

The purpose of this work was to demonstrate that the choice of an improved parameterisation is very effective to improve an optimisation process, especially in the case of a 3D airfoil shape optimisation.
A classic approach, in which the airfoil sections are represented by Bézier curves and the abscissas are fixed and uniformly distributed while the ordinates are taken as design variables, has produced in our test-case a drag reduction of 7 % only, and it seemed impossible to obtain further improvements. In fact, the classical parameterisation produced only very few feasible airfoils. Most of the designs were unfeasible, very often producing flow solver errors, because of some geometry distorsions.
We then decided to change the parameterisation, in particular we were searching for the abscissa distribution that might represent the best configuration, under a certain geometry tolerance, with the most regular Bézier control polygon possible. We have repeated four different primary optimisation steps, each of these step being followed by a procedure modifying the abscissa distribution accordingly to a secondary objective. In this way it was possible to improve the fitness function of more than 22 %. In particular, a last further improvement has been obtained by fixing the Bézier ordinates of the best configuration and considering as variables just the abscissas, and this method was just used as a final refinement due to the small geometry variations that were admitted.
Finally, our experiment indicated that a self-adaptive parameterisation results in *(i)* a larger number of feasible solutions (indicating a larger explored region), *(ii)* a notably more efficient

design optimisation, and *(iii)* a non-intuitive distribution of abscissas for the Bézier control points, indicating the necessity and usefulness of constructing self-adaptive algorithms.

Contents

1 INTRODUCTION **3**

2 TEST CASE AND MESH DESCRIPTION **4**
2.1 Geometry of the M6 Onera wing 4
2.2 Computational mesh 5

3 PARAMETERISATION AND OPTIMISATION STRATEGY **7**
3.1 Programs for the parameterisation 7
3.1.1 INGEOM program 8
3.1.2 BEZIER program 8
3.1.3 CREATECURVE program 8
3.2 Primary optimisation 9
3.3 Secondary optimisation 11
3.3.1 EXAM1.FOR and OPT4.FOR programs 12

4 RESULTS OF THE OPTIMISATION **13**
4.1 The optimisation process 13
4.2 Comparison of the best solution with the original one 15
4.3 Further optimisation of control points abscissas 19
4.4 A last remarck 20

5 CONCLUSIONS **20**

References

[1] D.Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Mass., 1989

[2] C.Poloni, V.Pediroda, "GA Coupled with Computationally Expansive Simulations: Tools to Improve Efficiency" , *Genetic Algorithms and Evolution Strategies in Engineering and Computer Science*, D.Quagliarella, J.Periaux, C.Poloni, G.Winter ed., John Wiley & sons, 1997

[3] S.Obayashi, A.Oyama, "Three-Dimensional Aerodynamic Optimisation with Genetic Algorithm", *Proceedings of the Third ECCOMAS CFD Conference*, Wiley, Chichester, U.K., 1996

[4] V.Schmitt, F.Charpin, "Pressure Distribution on the Onera M6 Wing at Transonic Mach Numbers", *Experimental Data Base for Program Assessment, Report of the Fluid Dynamics Panel Working Group 04*, AGARD AR 138, May 1979

[5] *StarCD 3.0 Reference Manual*, Computational Dynamics, London, 1998

[6] A.H.Shapiro, *The Dynamics and Thermodynamics of Compressible Fluid Flow*, Vols 1&2, Ronald Presse, 1954

[7] J.Lepine, F.Guibault, J-Y.Trepanier, F.Pepin, *Optimized Nonuniform Rational B-Spline Geometrical Representation for Aerodynamic Design of Wings*, AIAA JOURNAL, Vol.39, No.11, November 2001

[8] L.Piegl, W.Tiller, *The NURBS Book*, Springer, Berlin, 1995

[9] J.A.Samereh, *A Survey of Shape Parameterization Techniques*, NASA Langley research center, Hampton, VA, 1999

[10] G.Farin, *Curves and Surfaces for Computer Aided Geometric Design*, pp. 51-55, Academic Press, Inc., San Diego, 1990

[11] D.Spicer, J.Cook, C.Poloni, P.Sen, *EP20082: Industrial Multiobjective Design Optimisation*, ECCOMAS 98, J.Wiley&sons, 1998

[12] M. Hestenes, *Conjugate Gradient Methods in Optimization*, Springer-Verlag, New York, Inc., 1980

[13] J.A.Desideri, Z.L.Tang, *Self-Adaptive Parameterisation of Bézier Curves for Airfoil Design*, INRIA report (in preparation)

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399