Multi-Objective Optimization in CFD by Genetic Algorithms

Nathalie Marco, Jean-Antoine Desideri, Stephane Lanteri

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Multi-Objective Optimization in CFD by Genetic Algorithms

Nathalie Marco — Jean-Antoine Désidéri — Stéphane Lanteri

N° 3686

Avril 1999

THÈME 4

ISSN 0249-6399

INRIA
SOPHIA ANTIPOLIS

Multi-Objective Optimization in CFD by Genetic Algorithms

Nathalie Marco * , Jean-Antoine Désidéri , Stéphane Lanteri*

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet Sinus

Rapport de recherche n° 3686 — Avril 1999 — 40 pages

Abstract: This report approaches the question of multi-objective optimization for optimum shape design in aerodynamics. The employed optimizer is a semi-stochastic method, more precisely a Genetic Algorithm (GA). GAs are very robust optimization algorithms particularly well suited for problems in which (1) the initialization is not intuitive, (2) the parameters to be optimized are not all of the same type (boolean, integer, real, functionnal), (3) the cost functional may present several local minima, (4) several criteria should be accounted for simultaneously (multiphysics, efficiency, cost, quality, ...). In a multi-objective optimization problem, there is no unique optimal solution but a whole set of potential solutions since in general no solution is optimal w.r.t. all criteria simultaneously ; instead, one identifies a *set of non-dominated solutions*, referred to as the *Pareto optimal front*. After making these concepts precise, genetic algorithms are implemented and first tested on academic examples ; then a numerical experimentation is conducted to solve a multi-objective shape optimization problem for the design of an airfoil in Eulerian flow.

Key-words: Optimization - Multi-Objective Optimization - Pareto front - Criterion of non inferiority - Genetic Algorithms - Euler Equations - Shape parametrization - Parallel computing

* INRIA, 2004 Route des Lucioles, BP. 93, 06902 Sophia Antipolis Cédex-France

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, B.P. 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : 04 92 38 77 77 - International : +33 4 92 38 77 77 —Fax : 04 92 38 77 65 - International : +33 4 92 38 77 65

Optimisation multicritère en mécanique des fluides numérique par Algorithmes Génétiques

Résumé : Ce rapport traite d'optimisation multicritère pour la conception optimale de forme en aérodynamique. L'optimiseur utilisé est une méthode semi-stochastique, plus précisément un Algorithme Génétique (AG). Les AG sont des algorithmes d'optimisation très robustes particulièrement adaptés aux problèmes où (1) l'initialisation n'est pas intuitive, (2) les paramètres à optimiser ne sont pas tous du même type (booléen, entier, réel, fonctionnel), (3) la fonctionnelle coût peut présenter plusieurs minima locaux, (4) plusieurs critères sont à prendre en considération (multiphysique, qualité, efficacité, coût, ...). Dans un problème d'optimisation multicritère, il n'y a pas une solution optimale unique, mais un ensemble de solutions potentielles car en général aucune solution n'est *la meilleure* vis-à-vis de tous les critères simultanément ; on identifie alors *un ensemble de solutions non dominées* qui définissent un *front optimal de Pareto*. Après avoir précisé ces concepts, on met en œuvre des algorithmes génétiques et on les teste d'abord sur des exemples académiques ; puis on conduit une expérimentation numérique dans le cas de l'optimisation multicritère d'un profil d'aile plongé dans un écoulement eulérien.

Mots-clés : Optimisation - Optimisation multicritère - Front de Pareto - Critère de non-dominance - Algorithme génétique - Equations d'Euler - Paramétrisation de forme - Calcul parallèle

Contents

1 Introduction

In everyday's life, we commonly face multi-objective optimization problems such as buying a new television set compromising quality and low cost, or electing the optimal way to go to work minimizing both time spent and distance. In the aerospace engineering community, scientists try to build performant aircraft (minimal weight, maximal security, ...) at low cost. In economics, economists try to conciliate the utilities and the consumption of commodities to maximize the social welfare of each individual. The common part of these optimization problems is that they require several criteria to be taken into consideration simultaneously. In these cases, methods of single objective optimization are no more suitable. Multi-objective optimization should be considered instead. A large number of application areas of multi-objective optimization have been presented in the literature. The book of Cohon [1] is a good reference for a theoretical background on this subject.

In a simple optimization problem, the notion of optimality is straightforward. We seek for the best (the minimum or the maximum) value of an objective function. In a multi-objective optimization problem, the notion of optimality is not so obvious. If we agree in advance that we cannot link the values of the different objectives (i.e. if we refuse to compare apples and oranges), we must find a new definition of optimality, a definition that accounts for all the criteria. Then, the concept of Pareto optimality arises : although, in general, no solution is best w.r.t. all the criteria simultaneously, there exist a set of solutions that are strictly better than the remaining ones in the whole search space when considering all the objectives simultaneously. This set of solutions is known as *the Pareto optimal set* or *set of the non-dominated solutions* . The complementary set of solutions is called *the dominated solutions set.* A solution belonging to the Pareto set is not better than another one belonging to the same set, they are not comparable. Each of them is called a feasible solution (see for example [1]). The choice of a solution rather than another requires a thorough knowledge on the problem to be solved and a good knowledge of a lot of factors linked to the problem. Then, a solution chosen by a decision maker may be different from another decision maker's choice.

There are two different ways to solve a mutli-objective optimization problem. The first one is to make a linear combination of the different criteria with different weights and to minimize the resulting function. There are two main drawbacks with this method : (1) not all the solutions are found, (2) in a "penalty-function" approach,

the weights assigned to some criteria may not be suitable and the resulting function may lack significance. On the other hand, Genetic Algorithms (GAs) operate on an entire population of potential solutions (potential, because they belong to the feasible space). Then, during the optimization, GAs explore the whole feasible space and build a database of solutions determining a cloud. Convergence occurs when the cloud no more evolves; then, its convex hull defines the optimal Pareto set.

In this paper, we present in Section 2 a brief presentation of GAs (there are lot of references on GAs : D. Goldberg in [6], Z. Michalewicz in [11], ...). In Section 3, we give a general definition of a multi-objective optimization problem and its implementation by GAs. In Section 4, we apply GAs to three academic multi-objective optimization problems defined by analytical functions. Finally, in Section 5, we demonstrate the use of GAs for solving a multi-objective optimization problem in CFD (Computational Fluid Dynamics).

2 A brief presentation of GAs

2.1 A definition

Genetic Algorithms (GAs) rely on the analogy with the laws of natural selection and Darwin's most famous principle of *survival of the fittest*. GAs are different from classical optimization methods in the following ways :

- GAs operate simultaneously on an entire population of potential solutions (also called *chromosomes* or *individuals*) instead of producing successive iterates of a single element (as do gradient-based methods such as steepest descent or one-shot methods) ;
- the chromosomes are *coded in binary strings* ;
- the evolution operators make use of *probabilistic rules* ;
- the computation of the gradient of the cost functional is not necessary ;
- GAs have a greater potential to explore the whole search space and to climb the local maxima and to converge to the global optimum (a "point to point" method will generally stall in a local optimum) ;
- GAs, as semi-stochastic methods, are frequently found more robust in the case of non differentiable, multi-modal or non convex functions.

2.2 How do they work ?

A population of chromosomes evolves in the course of successive generations toward the best individual, solution of the optimization problem. The population is submitted to three genetic operators : *selection, crossover* and *mutation*. The individuals can either survive, reproduce or die, according to their fitness value which is related to the value of the cost functional to be optimized.

2.3 The genetic operators

Selection operator – According to its fitness value, a chromosome may either reproduce, survive or die. There are two types of selection :

- *Roulette-wheel* : a fitness value is assigned to each individual. A biased roulette-wheel is simulated. Each current string in the population has a roulette wheel slot sized in proportion to its fitness. In order to create a new population, the weighted roulette-wheel is spun n times, where n is the population size. In the case of a maximization problem, it is clear that the individual associated with greater roulette-wheel slot size will have more chances to survive or reproduce than others (see Fig. 1).
- *Tournament* : the most popular strategy is the *binary tournament*, also called *two-point tournament*. It consists in picking at random two individuals in the population and in comparing them. The best one is stored and both are re-introduced in the population. This procedure is started again until a new population of appropriate size has been obtained.

Crossover operator – As selection does not create new individuals, crossover is needed to increase diversity among the population. It is applied with a probability p_c, close to 1. Two individuals are selected at random for crossing over. Then, an integer k is chosen at random, between 1 and $l-1$ (l is the string length of the chromosomes) and k is the position of crossover. Two new strings (called the *offsprings*) are created by swapping all bits between positions $k+1$ and l inclusively (see Fig. 2).

Mutation operator – A mutation operator is needed because important genetic information (1's or 0's at special locations) may occasionaly be lost as a result of selection and crossover. It is introduced with a very small probability p_m (close to 0.001) and applied at the bit level ; it consists in changing a 1 (resp. 0) into a 0 (resp. a 1).

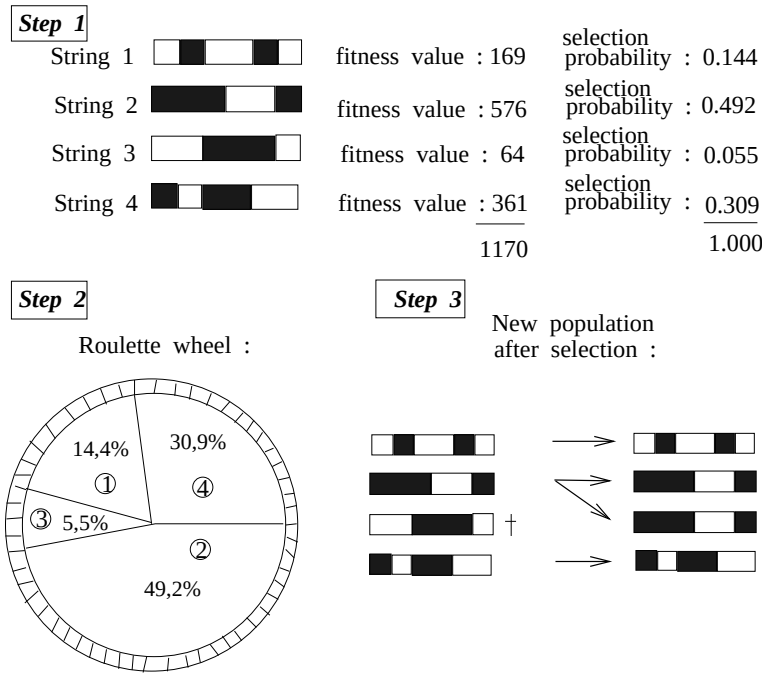


Figure 1: Example of the roulette-wheel selection in the case of 4 individuals

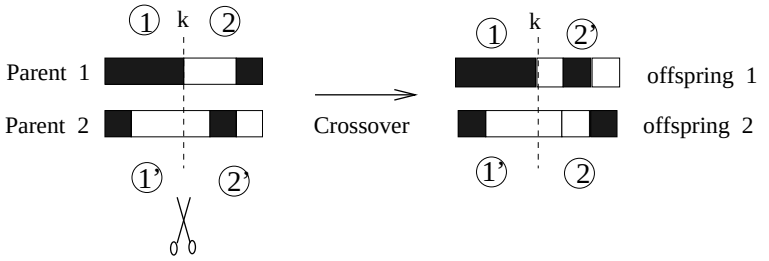


Figure 2: Crossing over of two chromosomes

3 Multi-objective optimization problem

3.1 A general multi-objective problem

A general multi-objective optimization problem consists of a number of objectives to be optimized simultaneously and is associated with a number of inequality and equality constraints (for a theoretical background, see for example the book of Cohon [1]). Such a problem can be stated as follows :

$$\text{Minimize (or Maximize)} \quad f_i(x) \quad i = 1, \cdots, N$$

$$\text{subject to :} \begin{cases} g_j(x) = 0 & j = 1, \cdots, M \\ h_k(x) \leq 0 & k = 1, \cdots, K \end{cases}$$

The f_i are the objective functions, N is the number of objectives, x is a vector whose p components are the design or decision variables.

In a minimization problem, a vector x^1 is said to be partially less than another vector x^2 when :

$$\forall i \quad f_i(x^1) \leq f_i(x^2) \qquad (i = 1, \cdots, N)$$
$$\text{and there exists at least one } i \text{ such that} f_i(x^1) < f_i(x^2)$$

We then say that *solution* x^1 *dominates solution* x^2.

For example, in the case of minimization of 2 criteria,

$$\begin{cases} \text{Minimize} & f(x) = (f_1(x), f_2(x)) \\ \text{such that} & x \in X \text{ (the feasible region)} \end{cases}$$

a potential solution x^1 is said to dominate x^2 iff :

$$f_1(x^1) < f_1(x^2) \text{ and } f_2(x^1) \leq f_2(x^2) \text{ or } f_1(x^1) \leq f_1(x^2) \text{ and } f_2(x^1) < f_2(x^2)$$

3.2 A classical method to solve a multi-objective problem

A common difficulty with a multi-objective optimization problem is the conflict between the objectives : in general, none of the feasible solutions is optimal w.r.t. all the objectives. Then, a solution of the Pareto set is a solution which offers the least objective conflict. One of the most classical methods is the method of objective weighting,

used in the 1900's by the welfare economists (see Cohon [1]). The multiple objective functions $f_i(x)$ are combined in one function $F(x)$ such that :

$$\begin{cases} F(x) = \sum_{i=1}^{N} \omega_i \ f_i(x) \\ \text{where} \quad x \in X, \quad X \text{ is the feasible region} \\ 0 \leq \omega_i \leq 1 \quad \text{and} \quad \sum_{i=1}^{N} \omega_i = 1 \end{cases}$$

It is clear that the preference given to one objective can be modified by changing the corresponding weight. On the other hand, the fact that we weight all the objectives implies the waste of a lot of informations. The only advantage of such a method is that an objective can be controlled by comparing it to another one and that the obtained optimum belongs to the optimal Pareto set. The main drawbacks are that we do not find all the solutions, and that it is possible to give some weights that are totally illogical with the studied optimization problem.

Other classical methods where the objectives are weighted are described, for example, by Srinivas and Deb [20].

3.3 Implementation by Genetic Algorithms

3.3.1 Ranking by fronts

GAs select individuals according to the values of the fitness function. However, in a multi-objective optimization problem, several criteria being considered, the evaluation of the individuals requires that a unique fitness value, referred to as a *dummy fitness* be defined in some appropriate way. To achieve this, by application of the definition of non-dominance, the chromosomes are first classified by fronts. The non-dominated individuals of the entire population define front 1 ; in the subset of remaining individuals, the non-dominated ones define front 2, and so on ; the worst individuals define front f, where f is the number of fronts.

To illustrate this, consider a situation where a population of 30 individuals is sorted according to two criteria (J_1, J_2). To be specific, randomly-drawn values for the criteria J_1 and J_2 have been assigned to the 30 individuals. The resulting fronts have then been identified and are shown on Fig. 3. In this example, the points belonging to fronts I, II, III, IV have greater chances to be reproduced than points belonging to fronts $VII, VIII$.

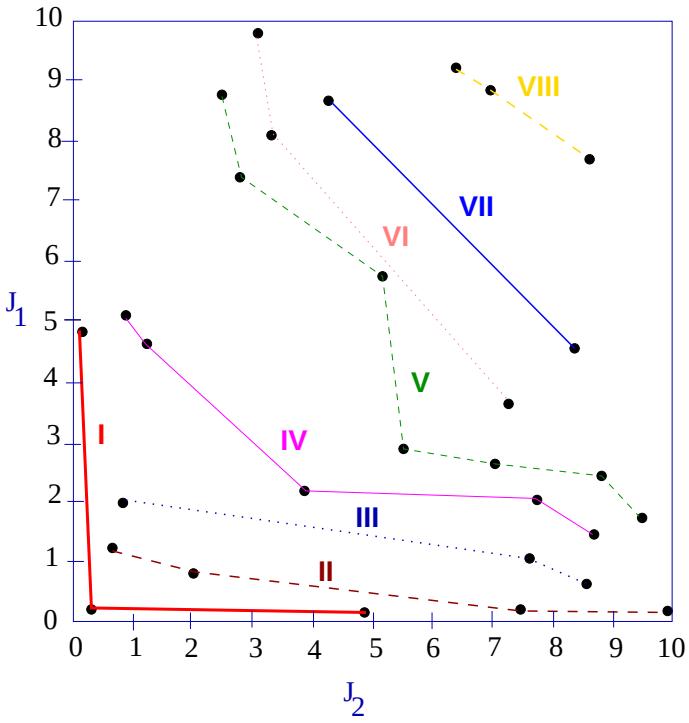


Figure 3: Ranking of a population by fronts

Once the individuals have been ranked by fronts, they are assigned the following dummy fitness values :

$$f_i = \begin{cases} 1/r & \text{in case of a minimization} \\ r & \text{in case of a maximization} \end{cases}$$

where r is the rank of the front.

3.3.2 An early experiment of application of GAs to a multi-objective problem

Because GAs use a population of individuals, their framework permits to identify a whole set of optimal, or more precisely non dominated solutions that define the Pareto set. To our knowledge, a first application of GAs to multi-objective optimization problems has been reported by Schaffer in 1984 in his dissertation [18]. Later, Horn and Nafpliotis [9], Srinivas and Deb [20] and Goldberg [6] made reference to the Schaffer method, called Vector Evaluated GA (VEGA). Schaffer's objective was to minimize a cost and maximize a reliability. In order to perform this, he separated the original population into two sub-populations of equal size and distributed the optimization of the two objectives on these sub-populations. After numerous generations, both sub-populations converged towards one optimum. This strategy opened a new avenue in the research of a method to solve a multi-objective problem. The results were encouraging but the obtained solutions had converged to only the extreme points of the Pareto set. Each of these two points corresponds to the optimum of a given criterion regardless of the other. Thus, the results by VEGA provided an evidence that the population should be characterized by species in order to obtain a whole set of trade-offs among the objectives. In order to get diversity (i.e. maintain individuals all along the Pareto front), a non dominated sorting procedure in conjunction with a sharing technique has been implemented, first by Goldberg in [6] and more recently by Horn in [9], and Srinivas in [20]. Then, the objective is to find a representative sampling of solutions all along the Pareto front.

Our results on multi-objective optimization are based on the algorithm of Srinivas and Deb [20], called the Nondominated Sorting Genetic Algorithm (NSGA). In the next paragraph we describe the NSGA method.

3.4 Nondominated Sorting Genetic Algorithms

The nondominated sorting procedure makes use of a selection method by ranking which emphasizes the optimal points. The sharing technique or niche method is used to

stablilize the sub-populations of the "good" points. We use such a strategy because one of the main defects of GAs in a multi-objective optimization problem is the possible premature convergence. In some cases, GAs may converge very quickly to a point of the optimal Pareto set and as the associated solution is better than the others (it is called a "super individual"), it breeds in the population and, after a certain number of generations, a population composed by copies of this solution only is obtained! Likewise, it is possible to obtain a Pareto set composed only of a few elements (as in the case of the VEGA method which yielded only two points). It is to avoid such a situation that the nondominated sorting technique is combined with the niche method anytime a solution is found in multiple copies ; then, its fitness value is decreased and, in the next generation, new different solutions appear, even if they are not so performant. The fitness values decrease because the different niches to which belong the different optimal solutions have been identified and treated.

The various steps of the method are the following :

- Before performing the selection step on the available population of solutions, we first identify the non dominated individuals (according to the criteria) as explained in paragraph 3.1. These individuals define front 1. The probability of reproduction of these individuals is very high.
- We then assign the same dummy fitness f_i to the non dominated individuals i of front 1 (generally, the dummy fitness f_i is equal to 1).
- To maintain the diversity, the dummy fitness of the individuals is then shared : it is divided by a quantity proportional to the number of individuals around it, according to a given radius. If the individual has numerous neighbours, a large number of similar solutions exist and the fitness value is split in order to favor diversity in the next generation. We call this phenomenon a *niche* and it is quantified as follows :

$$m_i = \sum_{j=1}^{M} Sh(d(i,j))$$

where i is the index of the individual, M is the number of individuals belonging to the current front; $Sh(d)$ is the *sharing function* which is a linear decreasing function of d defined by :

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

where :

- σ_{share} is the maximum phenotypic distance allowed between any two individuals to become members of a niche. Its value is problem-dependent and must be assigned by the user. In [8], Goldberg proposes to implement an adaptive niching scheme to overcome the limitations of fixed sharing schemes.
- $d(i,j)$ is the distance between two individuals i and j. In some multi-objective optimization problems, the *Hamming distance* is used. It is a genotypic (at the string level) distance and it consists in counting the number of different bits in the two strings, divided by the string length. Such a distance has beem used in electromagnetic system design optimization (see [12]) and in problems of finding an optimal distribution of active control elements in order to minimize the backscattering of a reflector in computationnal electromagnetics (see [5]). In this case, the Hamming distance has an actual significance because the bits 1 and 0 do not correspond to a real coding but to the fact that the control elements are active or not. When working with binary-coded real-values, the Hamming distance may be biased because two neighbouring solutions may have a very important Hamming distance (e.g : 0111 and 1000 correspond to 7 and 8, which are neighbours but their Hamming distance is maximal !). Then, in these cases, a phenotypic (at the real parameter level) distance is used, the *Euclidian distance*, of the form :

  $$d(i,j) = \sqrt{\sum_{k=1}^{np} \left(\frac{x_k^i - x_k^j}{max(k) - min(k)}\right)^2}$$

  np is the number of parameters defining the chromosomes i and j; x_k^i is the real value of parameter k of chromosome i; $\max(k) = \max\limits_{i=1,\cdots,M} x_k^i$ and $\min(k) = \min\limits_{i=1,\cdots,M} x_k^i$, M is the number of individuals belonging to the current front.

Then after a niche has been isolated and treated for each individual of the current front, we assign a new dummy fitness value, namely $\dfrac{f_i}{m_i}$.

- After the above sharing has been performed, the non dominated individuals are temporarily ignored from the population. For the new current front, we first assign to the each individual belonging to this front the same dummy fitness which is the minimum of the $\frac{f_i}{m_i}$ found in the previous front.
- We iterate this process until all the population has been visited.
- As a dummy fitness value has now been assigned to each individual in the population, selection, crossover and mutation can be applied in the usual manner. The corresponding flowchart of the method is presented in Fig. 4 (closely inspired from [20]).

Remark : In the case of a minimization problem, the population is sorted by assigning a greater dummy fitness to the best individuals : the individuals of front 1 have a greater fitness than the individuals of front 2 that, in turns, have a greater fitness than the individuals of front 3, and so on ; as a result, the Pareto optimization becomes a *maximization*.

4 Multi-objective optimization of analytical functions

4.1 Optimization problem from Schaffer's doctoral dissertation

The objective is to minimize a function f depending on one real variable x and defined by :

$$\text{Minimize} \quad f(x) = (f_1(x), f_2(x))$$

$$\text{such that} \quad x \in [-6, 6]$$

$$\text{with } f_1(x) = x^2 \quad \text{and} \quad f_2(x) = (x-2)^2$$

Analytical solution – The technique consists in minimizing the function $f(x) = \lambda f_1(x) + (1-\lambda) f_2(x)$ in which $\lambda \in [0, 1]$ is a parameter. Thus, a parametric representation of the optimal Pareto set is obtained by solving :

$$\frac{df}{dx} = f'(x) = 2\lambda x + 2(1-\lambda)(x-2) = 2x - 4 + 4\lambda = 0$$

giving $x = 2 - 2\lambda$; thus, $x \in [0, 2]$, $f_1 \in [0, 4]$ and $f_2 \in [0, 4]$.

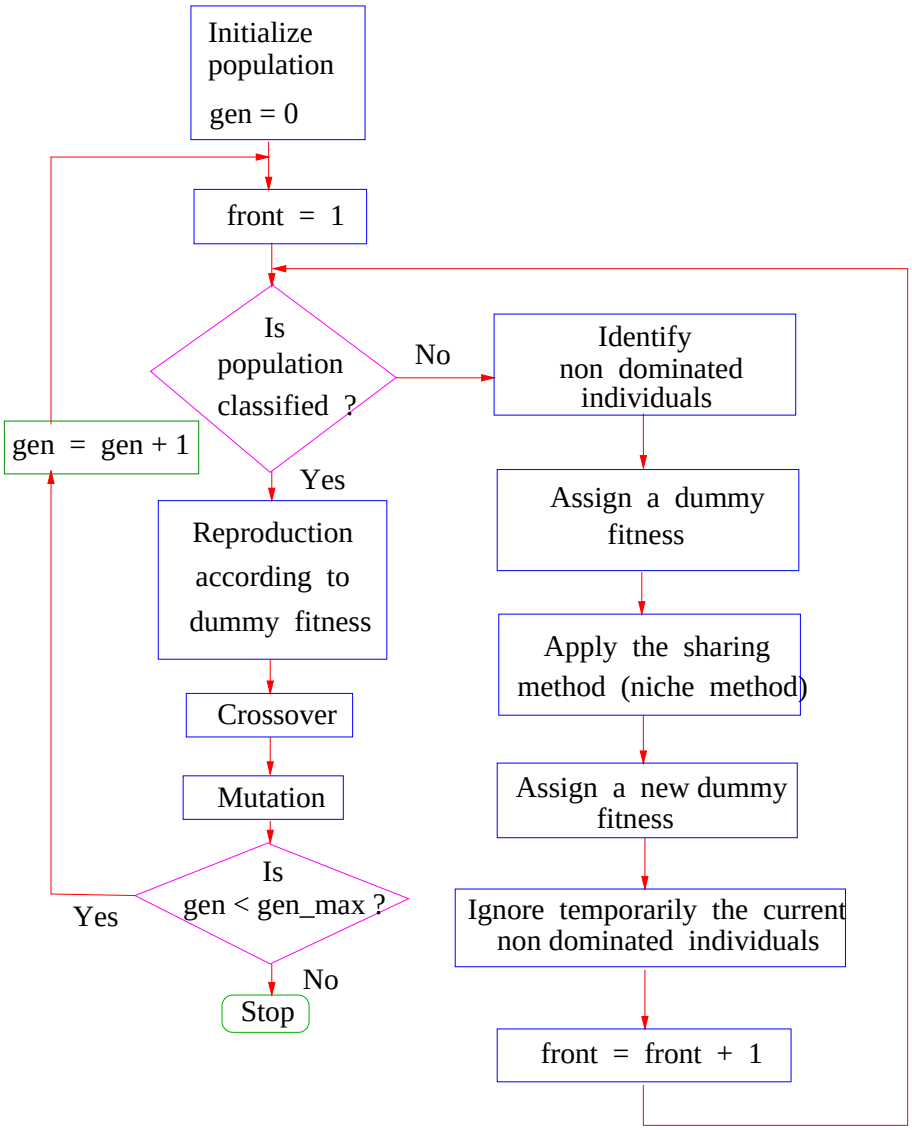


Figure 4: Flow-chart of the Nondominated Sorting GA

Solution by GA – The decision variable x has been coded in a binary string of length equal to 24 (the accuracy is then equal to 10^{-6}). Working with a population of 20 chromosomes, setting the probability of crossover to $p_c = 0.9$, the probability of mutation to $p_m = 0.05$ and $\sigma_{share} = 0.2$, an elitist strategy and a 2-point tournament have been employed.

Fig. 5 represents the two objectives in the decision space (function of the variable x) after the first generation. The Pareto front takes place in the interval $x \in [0, 2]$: while J_2 is decreasing to its minimal value, J_1 is increasing from its minimal value (a certain number of trade-off points appear between the two extreme optima). Fig. 6 depicts the evolution of the population in the objectives plane from the first generation to the last one. It is clear that the optimal Pareto set is located between $f_1 \in [0, 4]$ and $f_2 \in [0, 4]$. So, after 30 generations, all the individuals have converged in the region $x \in [0, 2]$ (see Fig. 7). The optimal Pareto set is shown in Fig. 8.

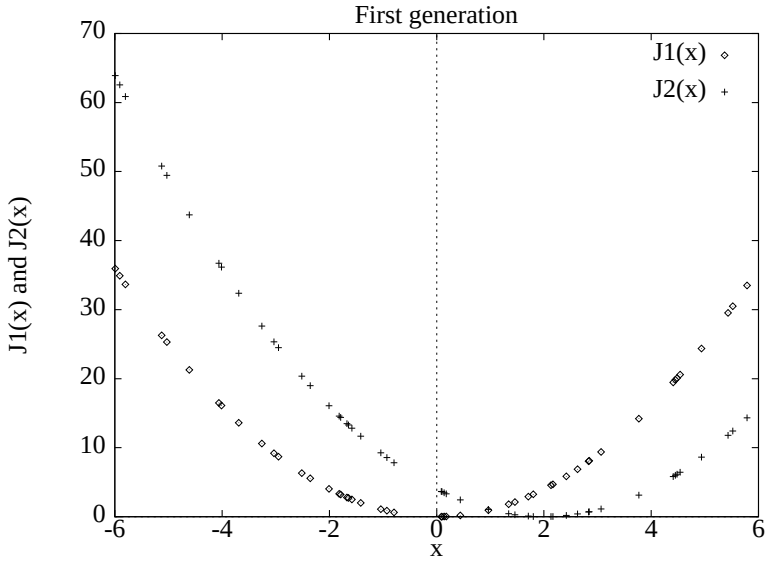


Figure 5: Objectives f_1 and f_2 in the decision space at the first generation.

4.2 A two-objective optimization problem

We now consider two academic analytical functions depending on two parameters. We seek for the solutions minimizing simultaneously f_1 and f_2 :

$$\begin{cases} f_1(x,y) &= (x-1)^2 + (y-3)^2 \\ f_2(x,y) &= (x-4)^2 + (y-2)^2 \end{cases} \quad \text{with} \quad (x,y) \in [-5,5] \times [-5,5]$$

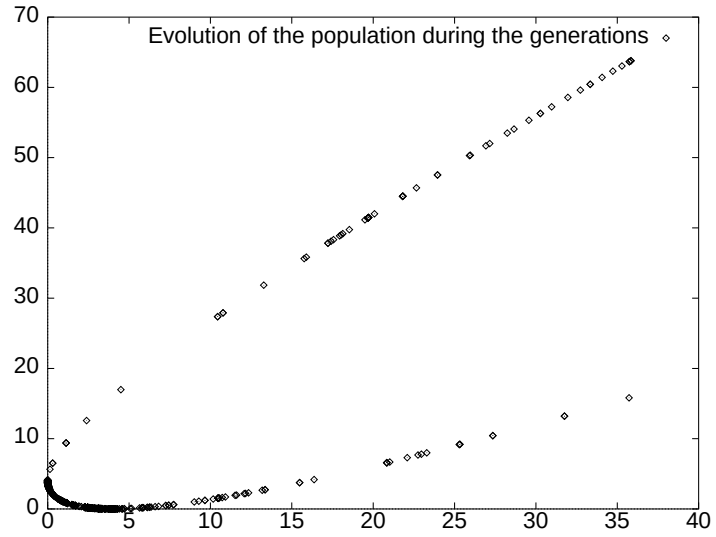


Figure 6: Evolution of the population in the objectives plane from the first generation to the last one.

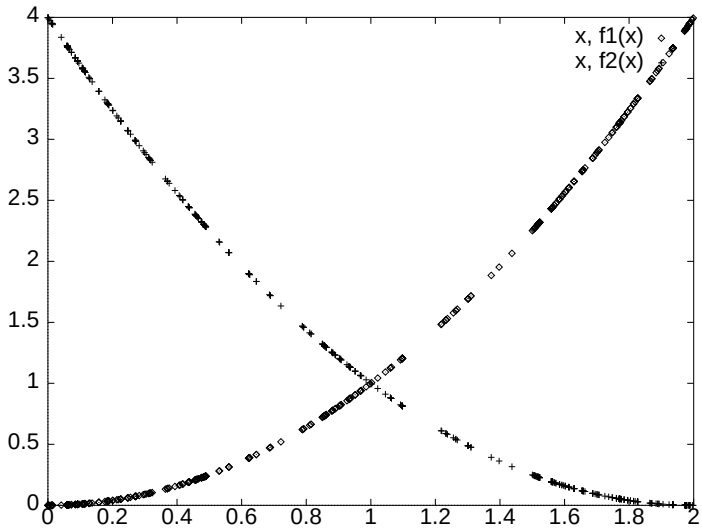


Figure 7: Convergence of the population after 30 generations to the region where there exist trade-offs between the two objectives ($x \in [0, 2]$).

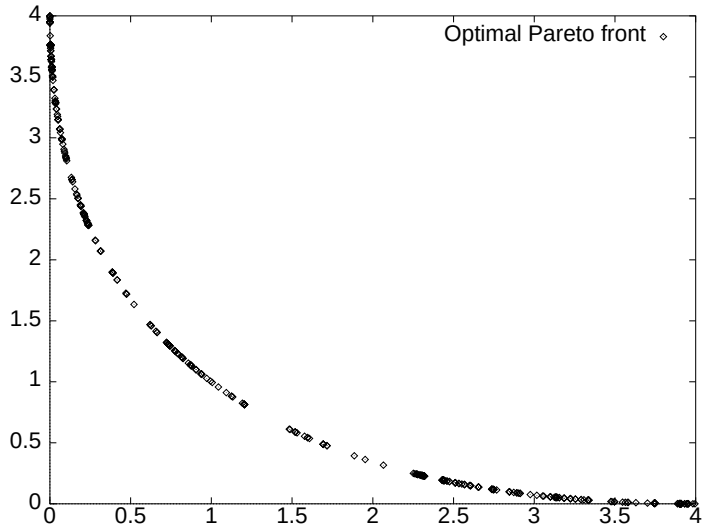


Figure 8: Optimal Pareto set after 30 generations. There are 473 individuals on the Pareto front. Note that individuals are rather uniformely distributed on the Pareto front.

Analytical solution – As previously, the technique is to minimize $f(x,y) = \lambda f_1(x,y) + (1-\lambda) f_2(x,y)$ in which $\lambda \in [0,1]$ is a parameter. Thus, the parametric representation of the optimal Pareto front is obtained by solving :

$$\begin{cases} \dfrac{\partial f(x,y)}{\partial x} = 0 \\ \dfrac{\partial f(x,y)}{\partial y} = 0 \end{cases} \Rightarrow \begin{cases} 2\lambda(x-1) + 2(1-\lambda)(x-4) = 0 \\ 2\lambda(y-3) + 2(1-\lambda)(y-2) = 0 \end{cases} \Rightarrow \begin{cases} x = 4 - 3\lambda \\ y = 2 + \lambda \end{cases}$$

As λ varies in $[0,1]$, x varies in $[1,4]$ and y varies in $[2,3]$. Then, f_1 and f_2 vary in $[0,10]$.

Solution by GA – The decision variables x and y have both been coded in a binary string of length equal to 27 (the accuracy is then equal to 10^{-7}). Working with a population of 26 chromosomes, setting the probability of crossover to $p_c = 0.9$, the probability of mutation to $p_m = 0.08$ and $\sigma_{share} = 0.8$, an elitist strategy and a selection by 2-point tournament have been used.

Fig. 9 represents the two objectives f_1 and f_2 in the decision space (x,y). Fig. 10 depicts the evolution of the population in the objectives plane from the first generation to the last one. It is clear that the optimal Pareto set is located between $f_1 \in [0,10]$ and $f_2 \in [0,10]$. After 100 generations, we obtain the optimal Pareto set shown on Fig.

11. There are 218 individuals on the set. The optimal individuals, functions of f_1 and f_2, are shown on Fig. 12-a and 12-b. The GAs solution is the same as the analytical solution : x varies in $[1, 4]$ and y varies in $[2, 3]$.

Remarks

1. If we do not apply the non dominated sorting GA technique (see section 3.4) i.e. if the fitness values of the individuals are of the form $f_i = \frac{1}{\text{rank of } i}$, the population converges towards only two solutions of the Pareto front. This result is presented on Fig. 13.

2. The notion of Pareto equilibrium originates from game theory, originally introduced in economics. Games are usually classified in two types : *cooperative games* and *non-cooperative games*. Pareto games are cooperative, which means that the players co-operate in the evolution process. Inversely, Nash and Stackelberg games are non-cooperative which means that there is no communication between the players, although each player's strategy depends on the others. In his interesting thesis, Sefrioui [19] illustrated the three different types of equilibrium (Pareto, Nash and Stackelberg) by means of academic examples.

4.3 A three-objective optimization problem

Now, we consider three academic analytical functions depending on two parameters. The objective is to minimize simultaneously the following three functions :

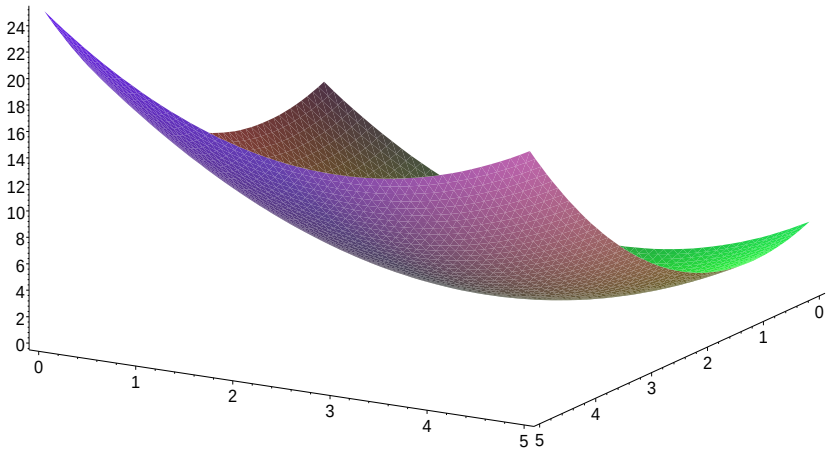
$$\left\{ \begin{array}{lcl} f_1(x,y) & = & x^2 + (y-1)^2 \\ f_2(x,y) & = & x^2 + (y+1)^2 + 1 \\ f_3(x,y) & = & (x-1)^2 + y^2 + 2 \end{array} \right. \quad \text{with} \quad (x,y) \in [-2,2] \times [-2,2]$$

Analytical solution – Here, the technique is to minimize $f(x,y) = \alpha f_1(x,y) + \beta f_2(x,y) + \gamma f_3(x,y)$ in which $\alpha, \beta, \gamma \in [0,1]$ are parameters such that $\alpha + \beta + \gamma = 1$. The optimal Pareto front is thus obtained by solving :

$$\left\{ \begin{array}{l} \dfrac{\partial f(x,y)}{\partial x} = 0 \\ \dfrac{\partial f(x,y)}{\partial y} = 0 \end{array} \right. \Rightarrow \left\{ \begin{array}{l} 2\alpha x + 2\beta x + 2\gamma(x-1) = 0 \\ 2\alpha(y-1) + 2\beta(y+1) + 2\gamma y = 0 \end{array} \right. \Rightarrow \left\{ \begin{array}{l} x = \gamma \\ y = \alpha - \beta \end{array} \right.$$

then, $x^{opt} \in [0,1]$ and $y^{opt} \in [-1,1]$, that means, $f_1^{opt} \in [0,5]$, $f_2^{opt} \in [1,6]$, $f_3^{opt} \in [2,4]$.

Function f1(x,y)



Function f2(x,y)

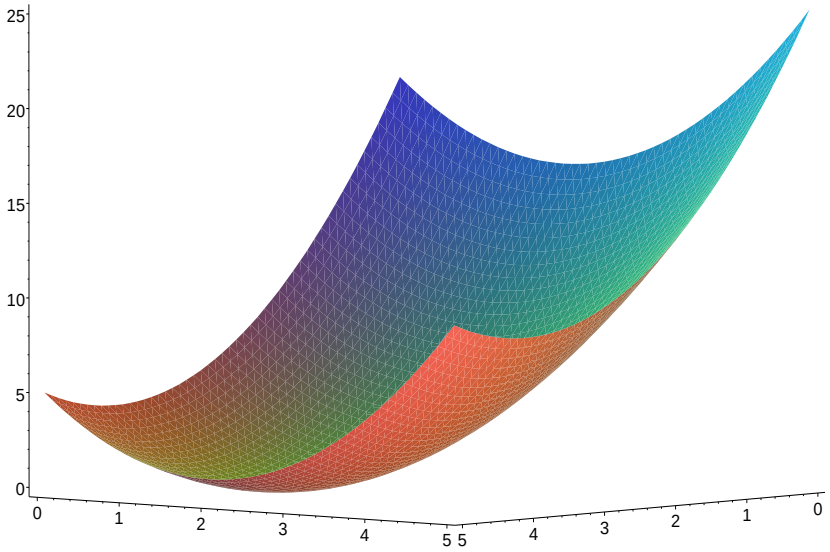


Figure 9: Objectives f_1 and f_2 in the decision space (x, y).

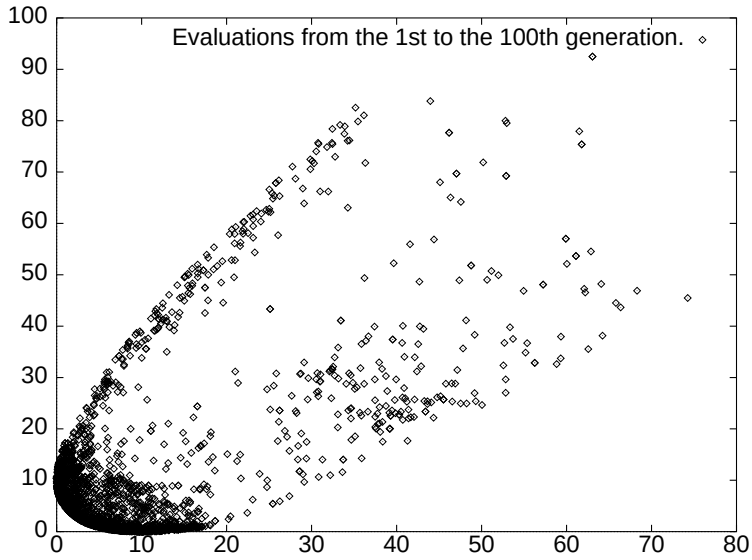


Figure 10: Evolution of the population in the objectives plane from the first generation to the last one (100 generations).

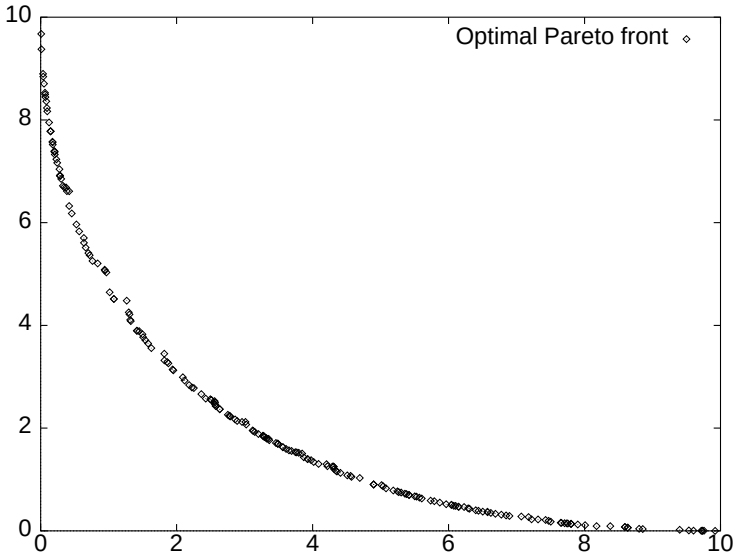
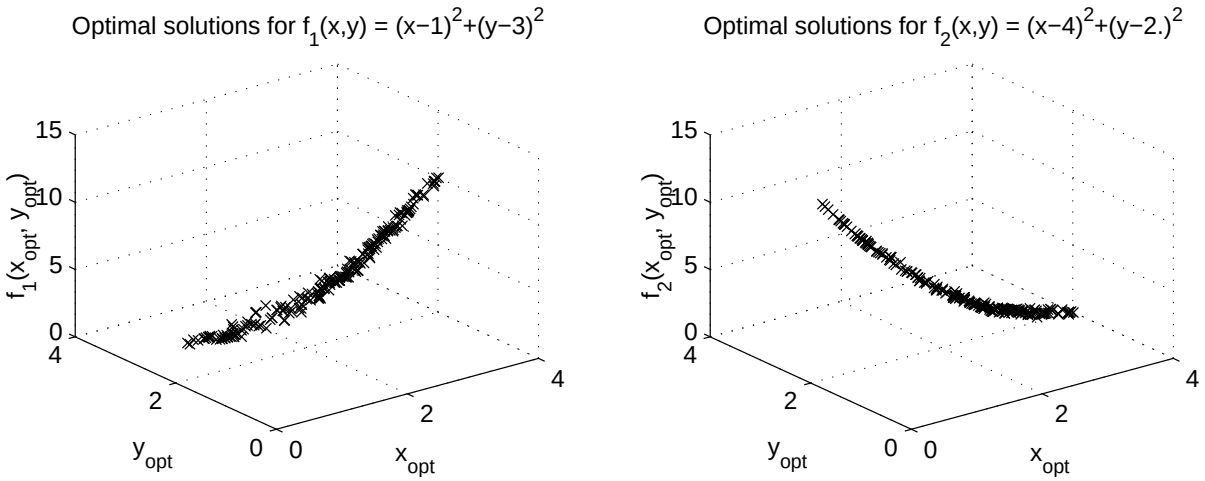


Figure 11: Optimal Pareto set after 100 generations. There are 218 individuals fairly uniformely distributed on the front.



Optimal individuals, function of f_1 Optimal individuals, function of f_2

Figure 12: The optimal individuals are plotted for f_1 and f_2 (after 100 generations).

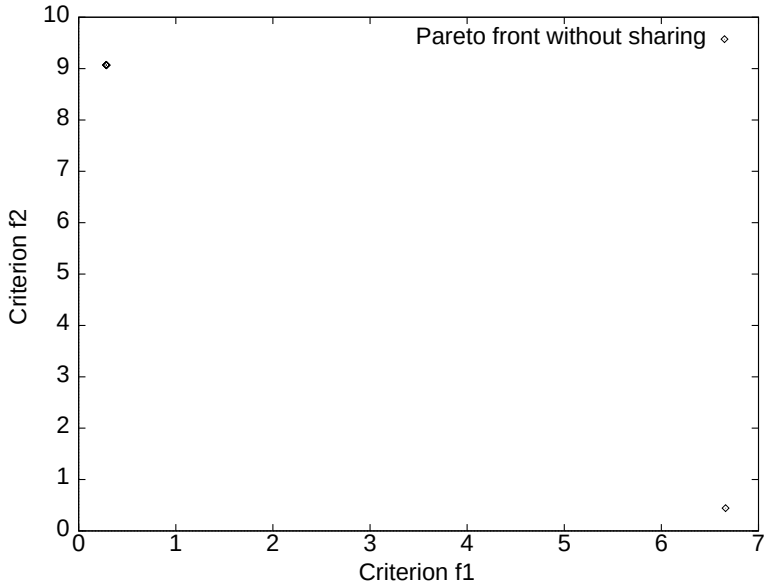


Figure 13: Pareto set without sharing. There are only two solutions.

Solution by GA – The decision variables x and y have both been coded in a binary string of length equal to 26 (the accuracy is then equal to 10^{-7}). Working with a population of 40 chromosomes, setting the probability of crossover to $p_c = 0.9$, the probability of mutation to $p_m = 0.05$ and $\sigma_{share} = 12.$, an elitist strategy and a selection by 2-point tournament have been used.

Fig. 14 depicts the evolution of the population in the objectives space from the first generation to the last one. We note that the optimal Pareto set is bounded by the intervals $f_1 \in [0, 5]$, $f_2 \in [1, 6]$ and $f_3 \in [2, 4]$. Fig. 15 provides the representation of f_1, f_2 and f_3 for the optimal solutions as well as the Pareto set as obtained after 100 generations ; this set is made of 1692 individuals. Finally, Fig. 16 depicts the optimal individuals $(x^{opt}\ ,\ y^{opt})$.

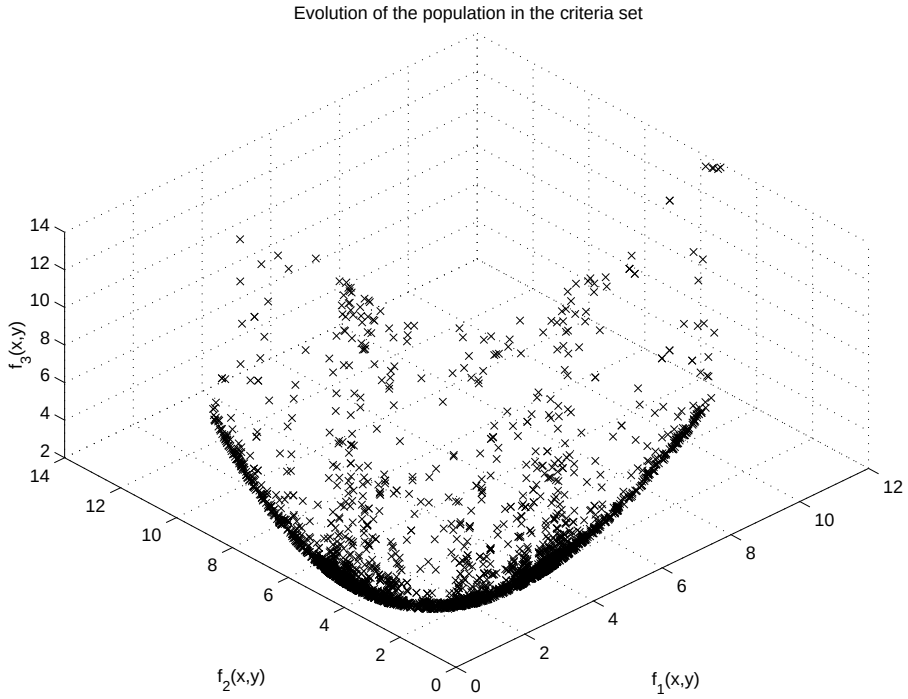


Figure 14: Evolution of the population from the first to the 100^{th} generation.

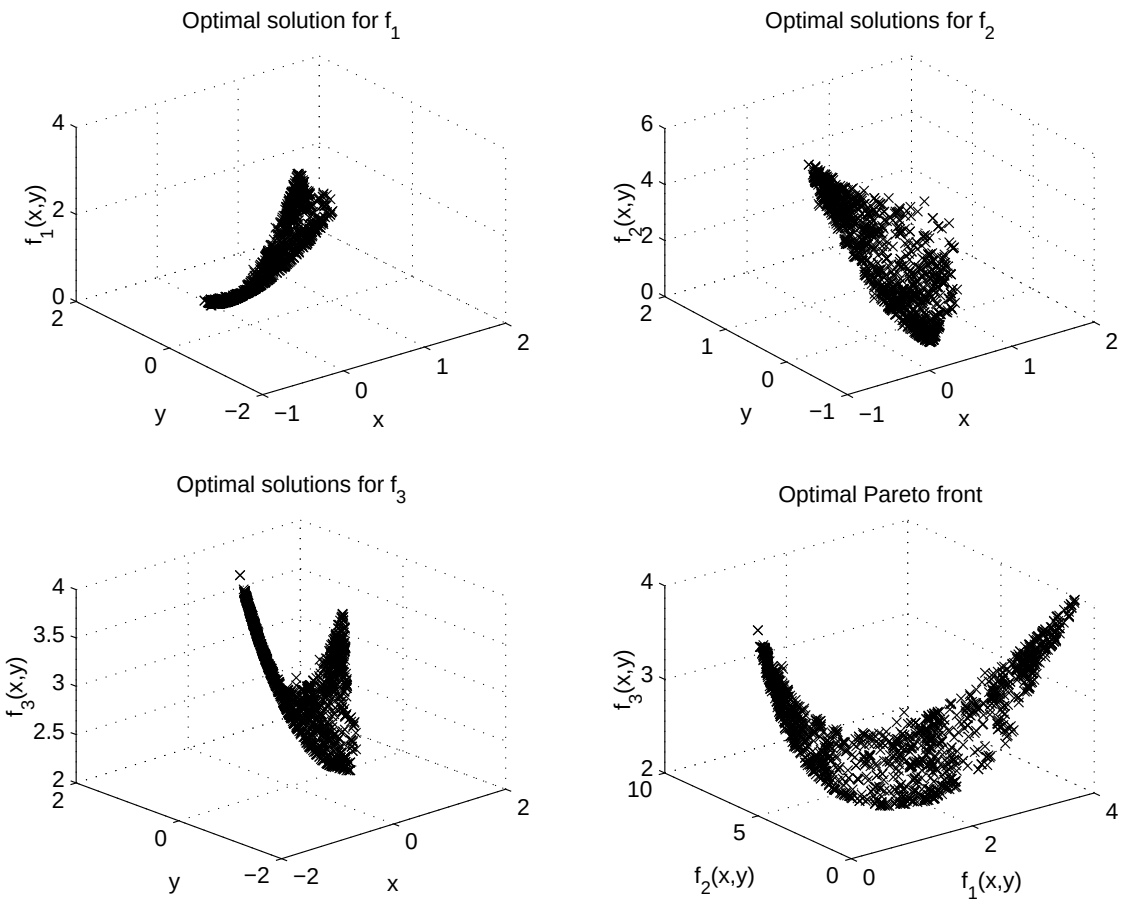


Figure 15: The optimal individuals for each criterion and the Pareto set in the space of the objectives.

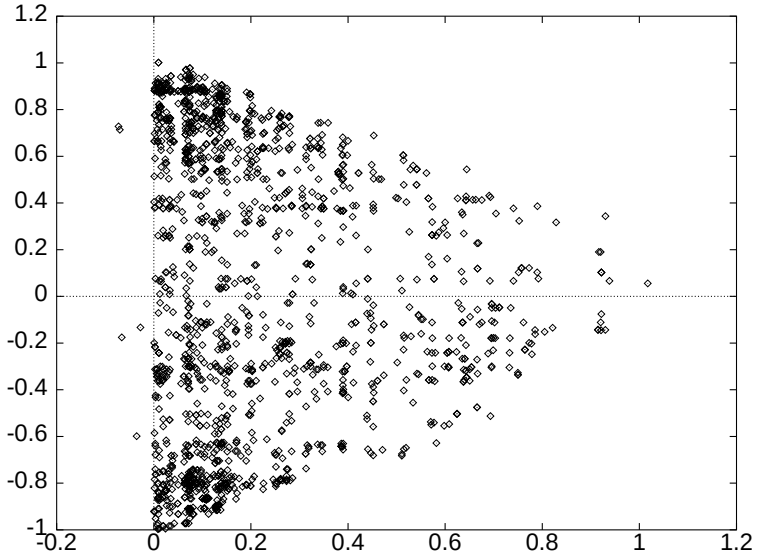


Figure 16: The optimal individuals.

5 Multi-objective optimization for Shape Optimum Design

We consider here the application of GAs to the multi-objective optimization of airfoil profiles. The objective functions involve aerodynamic coefficients that are deduced from the numerical simulation of the compressible flow around an airfoil geometry. We first describe the main ingredients of the underlying flow solver restricting ourselves to inviscid flows modelled by the Euler equations.

5.1 The 2D Euler flow solver

The underlying flow solver solves the 2D Euler equations :

$$\frac{\partial W}{\partial t} + \vec{\nabla}.\vec{\mathbf{F}}(W) = 0 \quad , \quad W = \left(\rho \ , \ \rho\vec{U} \ , \ E\right)^T \quad , \quad \vec{\nabla} = \left(\frac{\partial}{\partial x} \ , \ \frac{\partial}{\partial y}\right)^T \tag{1}$$

where $\vec{\mathbf{F}}(W) = (F_1(W), F_2(W))^T$ is the vector of convective fluxes whose components are given by :

$$F_1(W) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{pmatrix} \quad , \quad F_2(W) = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \end{pmatrix}$$

In the above expressions, ρ is the density, $\vec{U} = (u, v)^T$ is the velocity vector, E is the total energy per unit of volume and p denotes the pressure which is obtained using the perfect gas state equation $p = (\gamma_p - 1)(E - \frac{1}{2}\rho \parallel \vec{U} \parallel^2)$ where $\gamma_p = 1.4$ is the ratio of specific heats. The flow domain Ω is assumed to be a polygonal bounded region of $\mathbb{R}^2$. Let τ_h be a standard triangulation of Ω. A vertex of the mesh is denoted by S_i, and the set of its neighboring vertices by $N(S_i)$. At each vertex S_i, a control volume C_i is constructed by joining the middle of the edges $\{S_i, S_j\}$ for $S_j \in N(S_i)$ with the centroids of the triangles sharing S_i as a common vertex ; see Fig. 17. The boundary of C_i is denoted by ∂C_i, and the unit vector of the outward normal to ∂C_i by $\vec{\nu_i}$. The union of all these control volumes constitutes a dual discretization of Ω.

The spatial discretization method adopted here uses a finite volume upwind formulation. Briefly speaking, for each control volume C_i associated with a vertex S_i, one has to solve for n_i *one-dimensional Riemann problems* at the control volume boundary, n_i being the number of neighbors of S_i. The spatial accuracy of the Riemann solver depends on the accuracy of the interpolation of the physical quantities at the control volume boundary. For first order accuracy, the values of the physical quantities at the control volume boundary are set equal to the values in the control volume itself. An extension to second order accuracy can be performed via a "Monotonic Upwind Scheme for Conservative Laws" (MUSCL) technique of Van Leer [21] . It consists in providing the Riemann solver with an interpolated value, taking into account some gradient of the physical quantities. One can use a *finite element gradient* (P_1-Galerkin) computed on a particular triangle, or an *averaged nodal gradient*, which is taken as a particular average of the finite element gradients on the set of triangles sharing a given vertex.

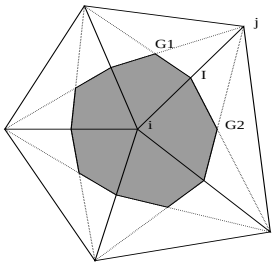


Figure 17: A 2D control volume C_i

Integrating Eq. (1) over C_i yields :

$$\begin{aligned} \iint_{C_i} \frac{\partial W}{\partial t}\vec{x} \;+\; \sum_{j\in N(S_i)} \int_{\partial C_{ij}} \vec{\mathbf{F}}(W).\vec{\nu_i}d\sigma \qquad & < 1 > \\ + \int_{\partial C_i \cap \Gamma_w} \vec{\mathbf{F}}(W).\vec{\nu_i}d\sigma \;+\; \int_{\partial C_i \cap \Gamma_\infty} \vec{\mathbf{F}}(W).\vec{\nu_i}d\sigma = 0 \quad & < 2 > \end{aligned} \tag{2}$$

where $\partial C_{ij} = \partial C_i \cap \partial C_j$; Γ_w and Γ_∞ respectively denote the wall boundary and the downstream/upstream boundary. A first order finite volume discretisation of $< 1 >$ is :

$$< 1 > = \; W_i^{n+1} - W_i^n + \Delta t \sum_{j\in N(S_i)} \Phi_{\mathbf{F}}(W_i^n, W_j^n, \vec{\nu_{ij}}) \tag{3}$$

where $\Phi_{\mathbf{F}}$ denotes a numerical flux function such that :

$$\Phi_{\mathbf{F}}(W_i, W_j, \vec{\nu_{ij}}) \approx \int_{\partial C_{ij}} \vec{\mathbf{F}}(W).\vec{\nu_i}d\sigma \quad , \quad \vec{\nu_{ij}} = \int_{\partial C_{ij}} \vec{\nu_i}d\sigma \tag{4}$$

Upwinding is introduced here in the computation of Eq. (4) by using Roe's [17] approximate Riemann solver thus computing $\Phi_{\mathbf{F}}$ as follows:

$$\Phi_{\mathbf{F}}(W_i, W_j, \vec{\nu_{ij}}) = \frac{\vec{\mathbf{F}}(W_i) + \vec{\mathbf{F}}(W_j)}{2}.\vec{\nu_{ij}} - \mid \mathbf{A}_R(W_i, W_j, \vec{\nu_{ij}}) \mid \frac{(W_j - W_i)}{2} \tag{5}$$

where $\mathbf{A}_R$ is Roe's mean value of the flux Jacobian matrix $\dfrac{\partial \vec{\mathbf{F}}(W)}{\partial W}.\vec{\nu}$. Following the MUSCL technique [21] , second order accuracy is achieved in Eq. (4) via a piecewise linear interpolation of the states W_{ij} and W_{ji} at the interface ∂C_{ij} (see Fig. 18) :

$$\tilde{W}_{ij} = \tilde{W}_i + \frac{1}{2}(\vec{\nabla}\tilde{W})_i.\vec{S_iS_j} \quad , \quad \tilde{W}_{ji} = \tilde{W}_j - \frac{1}{2}(\vec{\nabla}\tilde{W})_j.\vec{S_iS_j} \tag{6}$$

where $\tilde{W} = \left(\rho \;,\; \vec{U} \;,\; p\right)^T$. An *averaged nodal gradient* $(\vec{\nabla}\tilde{W})_i$ is obtained by averaging the P_1-Galerkin gradients computed on each triangle of C_i. Finally, the Van Albada limitation procedure [4] is introduced in the interpolation (6) in order to preserve the monotony of the approximation .

5.2 Time integration

As stated earlier, we are only interested here in *steady* solutions to the Euler equations. Therefore, time integration is only introduced as an artifice to set up an iterative solution algorithm and it must be as efficient as possible regardless of time accuracy. For this purpose, an implicit formulation that allows the use of large pseudo-time steps

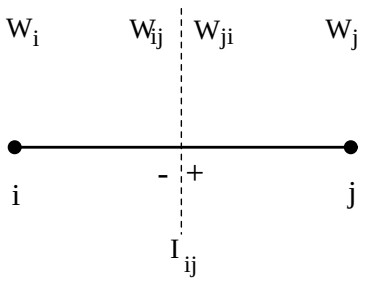


Figure 18: Interpolated physical states W_{ij} and W_{ji} on an edge $\{S_i, S_j\}$

is adopted. Assuming that $W(\vec{x}, t)$ is constant over C_i (in other words, a mass lumping technique is applied to the temporal term), we obtain :

$$\mathrm{vol}(C_i)\frac{dW_i}{dt} + \Psi(W)_i = 0 \quad , \quad i = 1, \cdots, N_v \tag{7}$$

where $W_i = W(\vec{x_i}, t)$, N_v is the number of control volumes, and :

$$\Psi(W)_i = \sum_{j \in N(S_i)} \Phi_{\mathbf{F}}(W_{ij}, W_{ji}, \vec{\nu_{ij}}) + \int_{\partial C_i \cap \Gamma} \vec{\mathbf{F}}(W).\vec{\nu_i} d\sigma \tag{8}$$

where $\Gamma = \Gamma_w \cup \Gamma_\infty$. Applying a first order linearization to the flux $\Psi(W^{n+1})$ yields the Newton-like formulation [4] :

$$\left(\frac{\mathrm{vol}(C_i)}{\Delta t^n} + J(W^n)\right)(W^{n+1} - W^n) = -\Psi(W^n) \tag{9}$$

where $J(W^n)$ denotes the associated Jacobian matrix. At each time step, the above linear system is approximately solved using Jacobi relaxations.

5.3 Shape parametrization

The population of individuals represent airfoil shapes. Following a strategy already adopted by several authors [10]-[14]-[15]-[16], the shape parametrization procedure is based on Bézier curves. A few control points are enough to represent the whole shape and the smoothness properties of Bézier curves permits to avoid non feasible shapes by the crossover operator. A Bézier curve of order n is defined by the *Bernstein polynomials* $B_{n,j}$:

$$B(t) = \sum_{i=0}^{n} B_{n,i} P_i \quad \text{with} \quad B_{n,i} = C_n^i t^i (1-t)^{n-i} \quad , \quad C_n^i = \frac{n!}{i!(n-i)!}$$

where $t \in [0,1]$, $P_i = (x_i, y_i)$ are the coordinates of the control points. In this work, a Bézier representation of order 8 has been used : 2 fixed points, P_0 and P_8, and 7 control points.

$$\begin{cases} x(t) = \sum_{i=0}^{8} C_8^i t^i (1-t)^{8-i}\ x_i \\ y(t) = \sum_{i=0}^{8} C_8^i t^i (1-t)^{8-i}\ y_i \end{cases}$$

Two Bézier curves are defined for the upper and lower surfaces respectively ; the points $P_0 = (0,0)$ and $P_8 = (1,0)$ corresponding to the leading and trailing edges of the airfoil are fixed. All the values $x_i \in [0,1]$ are fixed, and only the ordinates $y_1, \cdots, y_7$ (upper surface) and $y_9, \cdots, y_{15}$ (lower surface) are allowed to vary (see Fig. 19). Before coding a chromosome in a binary string, it is defined by a vector in $\mathbb{R}^{14}$:

$$chromosome = (\underbrace{y_1, \cdots, y_7}_{upper}, \underbrace{y_9, \cdots, y_{15}}_{lower})$$

Moreover, we have used a geometrical constraint on the ordinates : the y_i's vary on intervals of the form $[min_i, max_i]$; this has a direct implication on the search space. The parameters constituting the chromosomes are then coded as binary strings (for decoding a binary string into the real parameters see, for example, the book of Michalewicz [11] pp. 18-20).

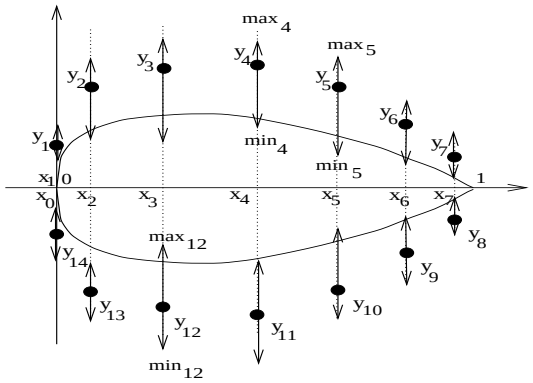


Figure 19: Representation of an airfoil using control points of Bézier (the y_i's).

5.4 The optimization problem

First, we evaluate a subsonic Euler flow at a Mach number equal to 0.2 and an incidence equal to 10.8° on a "high-lift" profile (see Fig. 20) ; we calculate the pressure, P_1.

Second, we evaluate a transonic Euler flow at a Mach number equal to 0.77 and an incidence equal to 1° on a "low-drag" profile (see Fig. 20), and we calculate the pressure P_2. The objective is to minimize simultaneously the two following cost functionals :

$$\begin{cases} J_1(W(\gamma)) &= \int_\gamma (P(W) - P_1)^2 \, d\sigma \quad \text{at } Ma = 0.2 \text{ and } \alpha = 10.8^\circ \\ J_2(W(\gamma)) &= \int_\gamma (P(W) - P_2)^2 \, d\sigma \quad \text{at } Ma = 0.77 \text{ and } \alpha = 1^\circ \end{cases}$$

We want to find all the profiles existing between the low-drag profile and the high-lift profile. For each chromosome determining the profile of an airfoil γ, we evaluate the two Euler flows at the two previously mentioned regimes. On Fig. 21 we visualize the steady iso-mach lines on the two profiles corresponding to the two flow regimes.

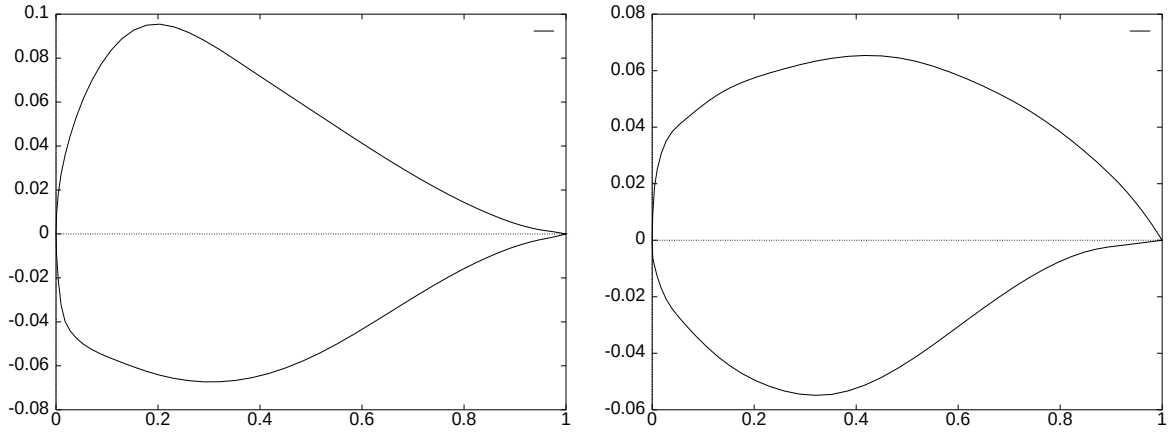


Figure 20: High-lift and low-drag profiles.

5.5 Mesh update procedure

Once a new shape has been determined, the overall computational mesh has to be updated prior to a new flow calculation. In this work, an unstructured dynamic mesh is represented by a pseudo structural model where a fictitious linear spring is associated with each edge connecting two vertices S_i and S_j. The vertices located on the downstream and upstream boundaries are held fixed while the new positions of those points located on the wall boundary are determined from shape parametrization procedure. Then, the new position of the interior vertices is determined from the solution of a displacement driven pseudo structural problem via a Jacobi iterative procedure performed on appropriate static equilibrium equations. This procedure is adapted from the work presented in Farhat and Lanteri [3] for unsteady flow simulation on dynamic meshes.

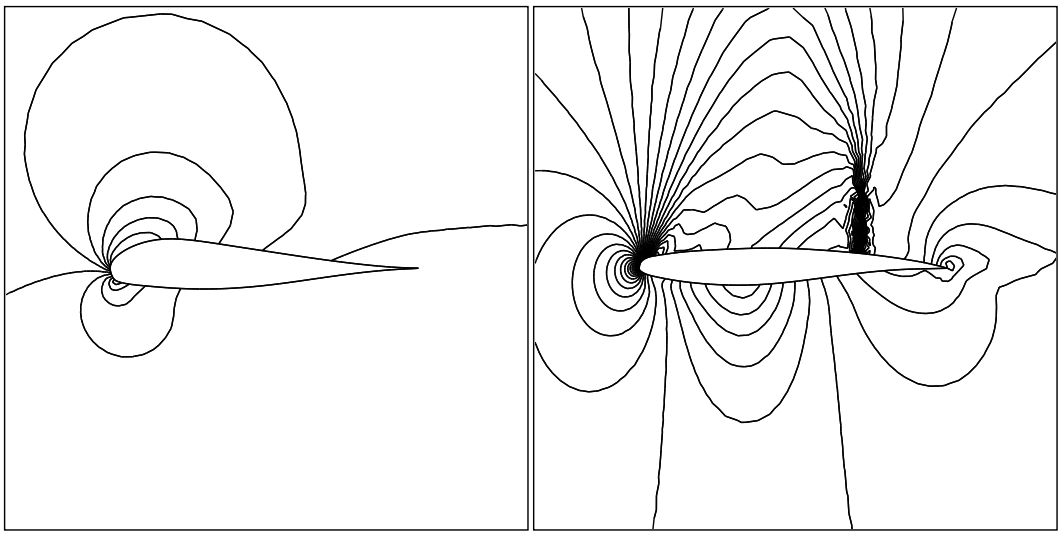


Figure 21: Steady iso-mach lines on the "highlift" and "lowdrag" profiles.

5.6 Parallelization strategy

One important concern related with the use of GAs in shape optimum design is the computational effort needed. In hard problems, GAs require large populations and this translates directly into higher computational effort. In the case of the multi-objective optimization problem considered here, we have to solve for two Euler flows for each individual, which means that the computational effort is very high. One of the main advantages of GAs is that they can easily be parallelized. At each generation, the fitness values associated with each individual can be evaluated in parallel. We will not go into the details for the parallelization strategy we have adopted as this is not the major topic of this report. A complete description can be found in [10]. In brief, we have employed a two-level parallelization strategy : the first level is the parallelization of the flow solver, which combines domain partitioning techniques with a message-passing programming model (where calls to the MPI library are used). The second level is the parallelization of GAs. We have chosen to exploit the notion of process groups which is one of the main features of the MPI environment. Two groups are defined, each group being responsible for the evaluation of the criteria for one individual, and each group contains the same number of processes. Then, two individuals are taken into account in parallel.

5.7 Results

The reference profile is a `NACA0012` airfoil. Results have been obtained using a coarse and a fine mesh of the `NACA0012` airfoil. The coarse mesh is composed of 1351 nodes and the fine one is composed of 14498 nodes. On Fig. 22, we visualize the steady iso-mach lines at the two different regimes for both meshes.

The flow solver – For each flow simulation, steady-state is assumed to be reached when the initial normalized density residual has been reduced by a factor 10^3. At each pseudo time-step, the linear system solution is interrupted when the residual has been reduced to one percent of the initial value.

The GA – Results have been obtained with a population of 40 individuals, setting the probability of crossover to $p_c = 0.8$, the probability of mutation to $p_m = 0.007$ and $\sigma_{share} = 0.7$. The selection operator was chosen to be the binary tournament with an elitist strategy.

5.8 Coarse mesh results

5.8.1 Parallel performance results

Results have been obtained on an `SGI Origin 2000` MIMD parallel computing platform equipped with `Mips R10000/195 Mhz` processors. We compare timing measurements for the overall optimization using one and two process groups. Timings are given for 100 generations in Tab. 1 and Tab. 2 where N_g and N_p respectively denote the number of process groups and the total number of processes ($N_g = 2$ and $N_p = 4$ means 2 processes for each group), "Total" is the total execution time and "Flow" is the accumulated flow solver time ; finally, $S(N_p)$ is the calculated speed-up. As could be anticipated, the present mesh is too coarse to demonstrate reasonable parallel speed-up of the optimization process; this point will be further illustrated later with comparisons between coarse and fine mesh calculations.

5.8.2 Numerical results

Fig. 23 visualizes the evolution of the population until a steady state is reached. We note the formation of the cloud of points and its convex hull. The optimal Pareto set is shown on Fig. 24, it describes the convex hull. After 100 generations, there are 131 individuals on the Pareto front. All these individuals correspond to optimal solutions

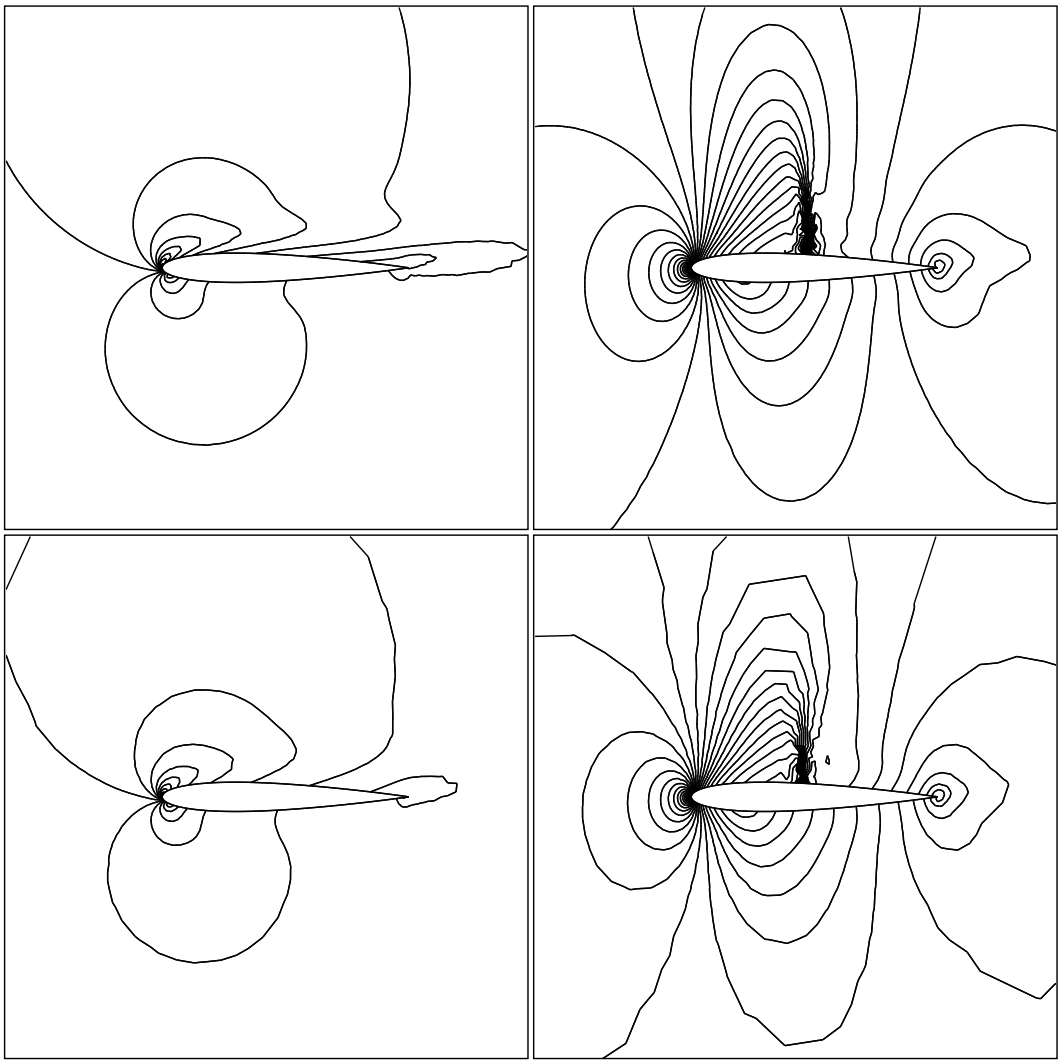


Figure 22: Top-left : fine mesh - highlift regime. Top-right : fine mesh - lowdrag regime. Bottom-left : coarse mesh - highlift regime. Bottom-right : coarse mesh - lowdrag regime.

Table 1: Parallel perfomance results on the `SGI Origin 2000` system
Coarse mesh calculations and one process group

N_g	N_p	Total (sec)	Flow (sec)	$S(N_p)$
1	2	5 h 45 mn	5 h 20 mn	1.0
1	4	4 h 01 mn	3 h 28 mn	1.4

Table 2: Parallel perfomance results on the `SGI Origin 2000` system
Coarse mesh calculations and two process groups

N_g	N_p	Total (sec)	Flow (sec)	$S(N_p)$
2	2	4 h 43 mn	4 h 25 mn	1.0
2	4	3 h 34 mn	3 h 13 mn	1.3

of the mutli-objective minimization This optimization has required 7730 cost functional evaluations.

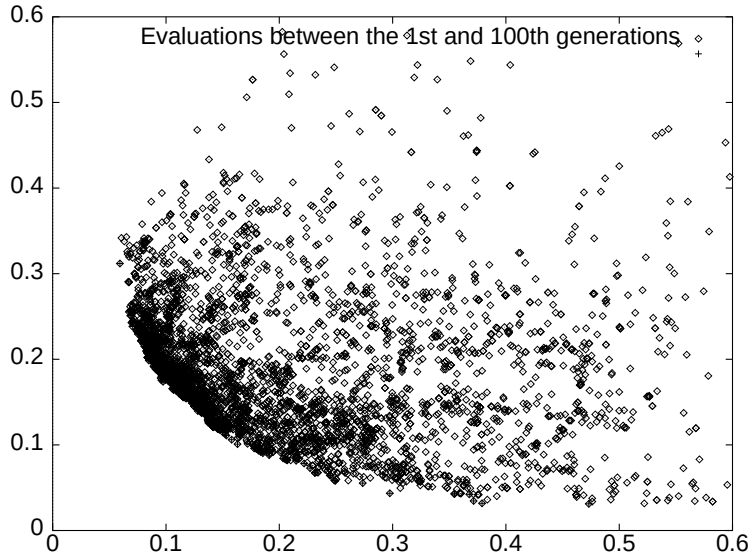


Figure 23: Evolution of the individuals in the objectives plane, from the first generation to the last one (100 generations).

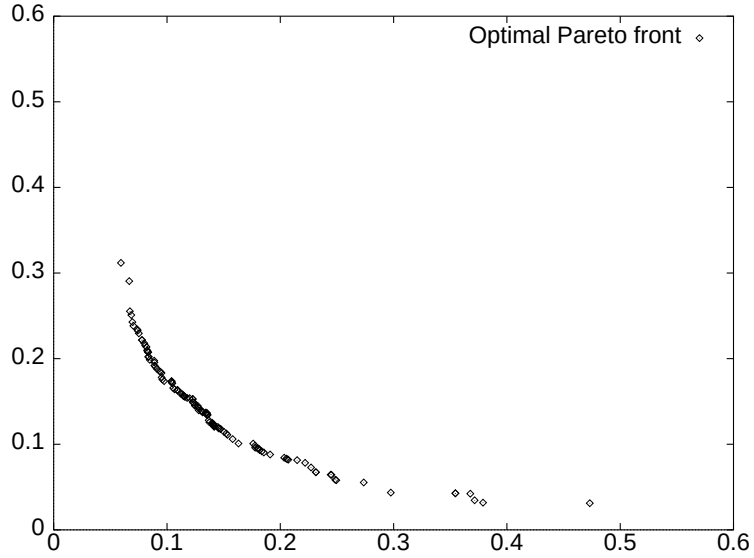


Figure 24: The optimal Pareto set with 131 individuals uniformly distributed, after 100 generations.

5.9 Fine mesh results

5.9.1 Parallel performance results

The complete optimization process required 51 generations and, for $N_g = 2$ groups and $N_p = 8$ processors, 22 hours on the `SGI Origin 2000` system. We have decided to compare the parallel performances between coarse and fine meshes, for only 10 generations, for the same data. Results are given in Tab. 3.

Table 3: Parallel perfomance results on the `SGI Origin 2000` system
Fine and coarse mesh calculations, two process groups

	N_p	Total	Flow	Com
Coarse mesh	8	18 min 43 sec	13 min 56 sec	5 min 10 sec
Coarse mesh	16	17 min 26 sec	11 min 50 sec	7 min 10 sec
Fine mesh	8	3 h 58 min 20 sec	3 h 9 min 46 sec	17 min 19 sec
Fine mesh	16	1 h 50 min 5 sec	1 h 24 min 56 sec	13 min 32 sec

The coarse mesh optimization has required 772 cost functional evaluations for the 10 generations, while for the fine mesh, 702 cost functionals have been evaluated. As previously, we note that on the coarse mesh, it is not worth to use a lot of processors.

Clearly, the situation is more favorable for the fine mesh calculations; a quasi-ideal speed-up is obtained when switching from 8 to 16 processors.

5.9.2 Numerical results

Results on the fine mesh are more accurate than the ones obtained on the coarse one. We illustrate this point by comparing the clouds composed by the different cost evaluations for both meshes and also by comparing the optimal Pareto fronts for both meshes (Fig. 25 and 26). When using the finest mesh, the two cost functionals have been better evaluated and the local optima have been better captured. On Fig. 27 a sample of different profiles between the high-lift and low-drag profiles is presented. We note that these profiles are slightly different than those resulting from the coarse mesh optimization.

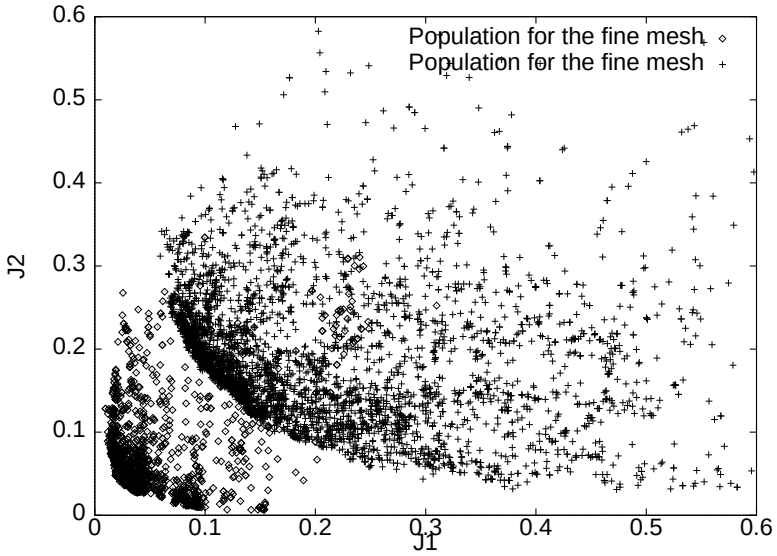


Figure 25: Evolution of the individuals in the objectives plane, for both runs on the two meshes (51 generations and 3918 evaluations for the fine mesh and 100 generations and 7730 evaluations for the coarse one).

6 Concluding remarks

The multi-objective optimization with Genetic Algorithms is independent of the decision makers : GAs work with a population of points, and they are able to find the Pareto optimal solutions simultaneously ; in effect, they collect a kind of database

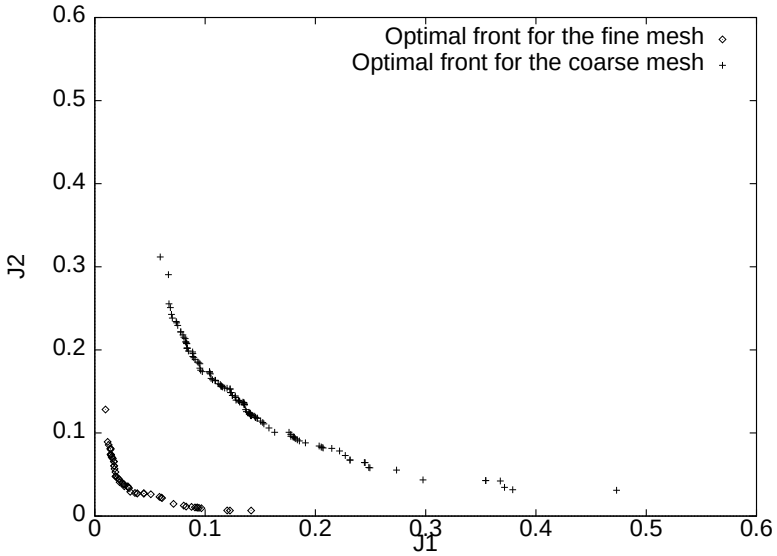


Figure 26: Optimal Pareto sets; the one obtained with the finest mesh (51 generations and 51 individuals) is dominating the one obtained with the coarse mesh (100 generations and 131 individuals).

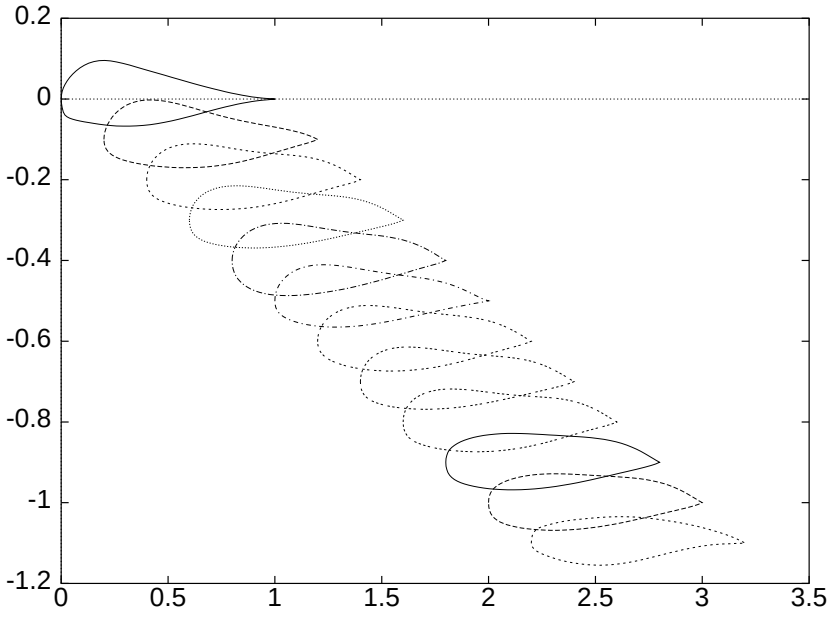


Figure 27: A sample of profiles between high-lift and low-drag, obtained with the fine mesh.

informative of the dynamics of the system. Then, whatever option is elected by the decision maker subsequently to the fabrication of the database, the optimal solution of the corresponding problem can be found on the Pareto front. Indeed, the common practice, in multi-objective optimization, is to fix some weights in order to combine linearly the objectives and to optimize only one function. Then, a wrong weighting may drive any optimization process toward wrong answers. Usually, the weights tuning is an expensive cut-and-try empirical procedure that implies necessarily a large number of optimization runs before reaching satisfactory results. Identifying the Pareto set permits to avoid this disadvantage.

In some multi-objective optimization problems, we do not succeed in obtaining all the feasible optimal solutions. The few points we obtain are optimal solutions but the trade-offs between the objectives are lacking. A procedure of sharing technique with a classification by fronts is then necessary to obtain the diversity between the optimal solutions all along the Pareto front. In our first approach on multi-objective optimization, we have studied only one sharing method, inspired by Srinivas and Deb [20]. But this technique has been improved, developped in many other optimization problems, for example by Goldberg and Richardson in [7], Deb and Goldberg in [2] who have used the tournament selection because it puts more controlled selection pressure and faster convergence characteristic. Oei and al. propose in [13] another niching technique. More recently, Goldberg [8] has proposed an adaptive niching scheme whose the objective is to overcome the limitations of fixed sharing methods. Further investigation of these tecniques is currently undergone.

The multi-objective optimization is used in optimum design when the problem is to find the different shapes of airfoils at different speeds. The optimal solutions of the Pareto front represent a family of shapes for the different mach numbers, corresponding to different weighted cost functionals. In [15], Poloni gives results concerning to the geometric transition from low-drag to high-lift airfoils.

References

[1] J.L. Cohon. *Multiobjective programming and planning.* Academic Press, 1978.

[2] K. Deb and D.E. Goldberg. An investigation of niches and species formation in genetic function optimization. In J.-D. Schaffer, editor, *Third international conference on Genetic Algorithms*, pages 42–50. Morgan Kaufmann, 1991.

[3] C. Farhat and S. Lanteri. Simulation of compressible viscous flows on a variety of mpps: computational algorithms for unstructured dynamic meshes and performance results. *Comp. Meth. in Appl. Mech. and Eng.*, 119:35–60, 1994.

[4] L. Fezoui and B. Stoufflet. A class of implicit upwind schemes for Euler simulations with unstructured meshes. *J. of Comp. Phys.*, 84:174–206, 1989.

[5] R. Glowinski, J. Periaux, M. Sefrioui, and B. Mantel. Optimal backscattering of a reflector by means of genetic algorithms. In J.-A. Désidéri *et al.*, editor, *Third ECCOMAS Computational Fluid Dynamics Conference and Second ECCOMAS Conference on Numerical Methods in Engineering*, pages 251–257. John Wiley & Sons, 1996. Conference computational methods in applied sciences.

[6] D.E. Goldberg. *Genetic Algorithms in search, optimization and machine learning.* Addison Wesley Company Inc., 1989.

[7] D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Second international conference on Genetic Algorithms : Genetic Algorithms and their applications*, pages 41–49. Morgan Kaufmann, 1987.

[8] D.E. Goldberg and L. Wang. Adaptive niching via coevalutionary sharing. In C. Poloni D. Quagliarella, J. Périaux and G. Winter, editors, *Genetic Algorithms and evolution strategies in engineering and computer science*, pages 21–38. John Wiley & Sons, 1997.

[9] J. Horn and N. Nafpliotis. Multiobjective optimization using the niched pareto genetic algorithm. In *First IEEE conference on evolutionary computation, IEEE world congress on computational intelligence*, volume 1, 1994.

[10] N. Marco and S. Lanteri. A two-level parallelization strategy for genetic algorithms applied to shape optimum design. Technical Report 3463, INRIA, 1998.

[11] Z. Michalewicz. *Genetic Algorithms + data structures = evolution programs.* Springer Verlag, 1992. Artificial intelligence. Genetic Algorithms in engineering and computer science.

[12] E. Michielssen and D.S. Weile. Electromagnetic system design using genetic algorithms. In M. Galàn G. Winter, J. Périaux and P. Cuesta, editors, *Genetic Algorithms in engineering and computer science*, pages 345–368. John Wiley & Sons, 1995.

[13] C.K. Oei, D.E. Goldberg, and S.J. Chang. Tournament sele'ction, niching and the preservation of diversity. Technical Report 91011, IlliGAL Report, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, 1992.

[14] J. Périaux, M. Sefrioui, B. Stoufflet, B. Mantel, and E. Laporte. Robust genetic algorithms for optimization problems in aerodynamic design. In M. Galàn G. Winter, J. Périaux and P. Cuesta, editors, *Genetic Algorithms in engineering and computer science*, pages 371–396. John Wiley & Sons, 1995.

[15] C. Poloni. Hybrid GA for multi-objective aerodynamic shape optimization. In M. Galàn G. Winter, J. Périaux and P. Cuesta, editors, *Genetic Algorithms in engineering and computer science*, pages 397–415. John Wiley & Sons, 1995.

[16] D. Quagliarella. Genetic algorithms applications in computational fluid dynamics. In M. Galàn G. Winter, J. Périaux and P. Cuesta, editors, *Genetic Algorithms in engineering and computer science*, pages 417–442. John Wiley & Sons, 1995.

[17] P.L. Roe. Approximate riemann solvers, parameters vectors and difference schemes. *J. of Comp. Phys.*, 43:357–371, 1981.

[18] J.D. Schaffer. Some experiments in machine learning using vector evaluated genetic algorithms, 1984. Unpublished doctoral dissertation, Vanderbilt University, Nashville (TN).

[19] M. Sefrioui. *Algorithmes évolutionnaires pour le calcul scientifique : application à l'électromagnétisme et à la mécanique des fluides numériques.* PhD thesis, University of Paris VI and Laforia, 1998.

[20] N. Srinivas and K. Deb. Multiobjective optimization using non-dominated sorting in genetic algorithm. *Evolutionary Comput.*, 2(3):221–248, 1995.

[21] B. Van Leer. Towards the ultimate conservative difference scheme V : a second-order sequel to Godunov's method. *J. of Comp. Phys.*, 32:361–370, 1979.

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

Unité de recherche INRIA Lorraine : Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - B.P. 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 - B.P. 105 - 78153 Le Chesnay Cedex (France)

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