Nash Genetic Algorithms : examples and applications

M. Sefrioui
LIP6, University Paris 6
4, Place Jussieu
75252 Paris, France
sefrioui@poleia.lip6.fr

J. Periaux
Dassault Aviation
78 Quai Marcel Dassault
92214 Saint-Cloud, France
periaux@rascasse.inria.fr

Abstract- This article presents both theoretical aspects and experimental results on Nash Genetic Algorithms. Nash GAs are an alternative for multiple objective optimization as they are an optimization tool based on non-cooperative game theory. They are explained in details, along with the advantages conferred by their equilibrium state. This approach is tested on a few benchmark problems, and some comparisons are offered with Pareto GAs, particularly in terms of speed and robustness. The different concepts presented in this paper are then illustrated via experiments on a Computational Fluid Dynamics problem, namely a nozzle reconstruction with multiple criteria (subsonic and transonic shocked flows). The overall results are that Nash Genetic Algorithms offer a fast and robust alternative for multiple objective optimization.

1 Introduction

When it comes to multiple objective optimization, Pareto GAs have now became a sort of standard. With the introduction of non-dominance Pareto-ranking and sharing (in order to distribute the solutions over the entire Pareto front) the Pareto GAs are a very efficient way to find wide range of solutions to a given problem [Goldberg, 1989]. This approach was further developed in [Srinivas and Deb, 1995], and lead to many applications [Poloni, 1995, Makinen et al., 1996, Bristeau et al., 1999]. All of these approaches are cooperative and based on Pareto ranking and use either sharing or mating restrictions to ensure diversity; a good overview can be found in [Fonseca and Fleming, 1995].

However, another multiple objective scheme, this time a non-cooperative one, has been presented by J. Nash in the early 50's [Nash, 1950, Nash, 1951]. This approach introduced the notion of player and aimed at solving multiple objective optimization problems originating from Game Theory and Economics. This paper is mostly devoted to present a GA we developed based on the concept of a non cooperative multiple objective algorithm.

2 Nash GAs

2.1 Definition

Nash optima define a non-cooperative multiple objective optimization approach first proposed by J. F. Nash [Nash, 1951]. Since it originated in Games Theory and Economics, the notion of player is often used and we kept it. For an optimization problem with G objectives, a Nash strategy consists in having G players, each optimizing his own criterion. However, each player has to optimize his criterion given that all the other criteria are fixed by the rest of the players. When no player can further improve his criterion, it means that the system has reached a state of equilibrium called *Nash Equilibrium*. Let E be the search space for the first criterion and F the search space for the second criterion. A strategy pair $(\overline{x}, \overline{y}) \in E \times F$ is said to be a Nash equilibrium iff:

$$f_E(\overline{x}, \overline{y}) = \inf_{x \in E} f_E(x, \overline{y})$$
$$f_F(\overline{x}, \overline{y}) = \inf_{y \in F} f_F(\overline{x}, y)$$

It may also be defined by:
$u = (u_1, \ldots, u_G)$ is a Nash equilibrium iff: $\forall i, \forall v_i$

$$J_i(u_i, .., u_{i-1}, u_i, u_{i+1}, ..u_G) \leq J_i(u_i, .., u_{i-1}, v_i, u_{i+1}, ..u_G)$$

With a classical approach, the main problem with Nash Equilibria is that they are very difficult to find. It is generally easier to prove that a given solution is a Nash Equilibria, but exhibiting such a solution may reveal very hard. And it becomes even harder if the criteria are non-differentiable functions. Next section will show that GAs offer an elegant alternative.

2.2 Merging Nash Equilibrium and GAs

The idea is to bring together genetic algorithms and Nash strategy in order to make the genetic algorithm *build* the Nash Equilibrium [Sefrioui, 1998].

Let $s = XY$ be the string representing the potential solution for a dual objective optimization, where X corresponds to the first criterion and Y to the second one. The first idea is to assign the optimization task of X to a player called *Player 1* and the optimization task of Y to *Player 2*. Thus, as advocated by Nash theory, Player 1 optimizes s with respect to the first criterion by modifying X, while Y is fixed by Player 2. Symmetrically, Player 2 optimizes s with respect to the second criterion by modifying Y while X is fixed by Player 1.

The next step consists in creating two different populations, one for each player. Player 1's optimization task is per-

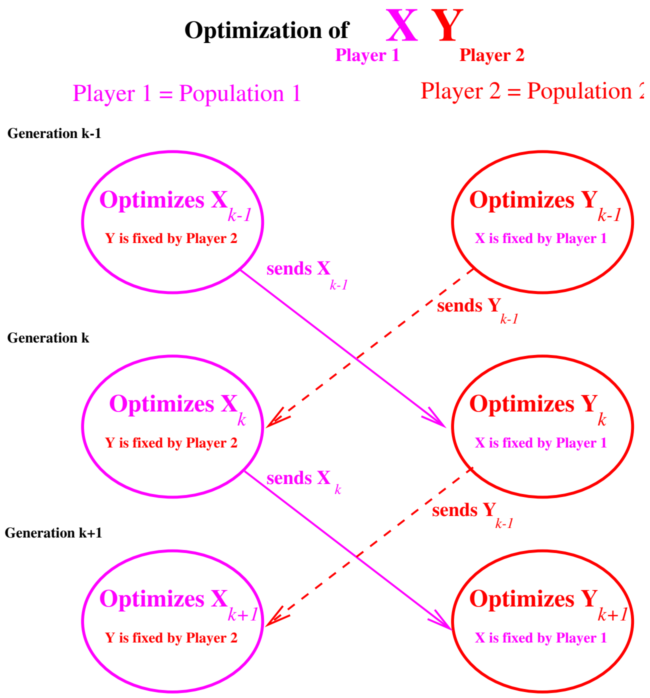

Figure 1: Nash strategy

formed by population 1 whereas Player 2's optimization task is performed by population 2.

Let X_{k-1} be the best value found by Player 1 at generation $k-1$, and Y_{k-1} the best value found by Player 2 at generation $k-1$. At generation k, Player 1 optimizes X_k while using Y_{k-1} in order to evaluate s (in this case, $s = X_k Y_{k-1}$). At the same time, Player 2 optimizes Y_k while using X_{k-1} ($s = X_{k-1} Y_k$). After the optimization process, Player 1 sends the best value X_k to Player 2 who will use it at generation $k+1$. Similarly, Player 2 sends the best value Y_k to Player 1 who will use it at generation $k+1$. Nash equilibrium is reached when neither Player 1 nor Player 2 can further improve their criteria.

This setting may seem to be similar to that of the so-called Island Model in Parallel Genetic Algorithms (PGA [Mühlenbein et al., 1991]). However, there is a fundamental difference in the sense that PGAs use the same criterion for each sub-population whereas Nash GAs use different criteria (thus introducing the notion of equilibrium).

We first developed Nash GAs with binary-coded GAs to solve combinatorial discrete problems [Sefrioui et al., 1996b, Bristeau et al., 1999]. But for this paper, we used a version based on real-coded GAs. So in the following, each player is evolved by a real-coded GA, using a non-uniform mutation scheme [Michalewicz, 1992]. It also uses *distance-dependent mutation*, a technique we evolved to maintain diversity in small populations. Instead of a fixed mutation rate, each offspring has its mutation rate computed after each mating. And this mutation rate depends on the distance between the two parents (for more details, see [Sefrioui et al., 1996a]).

Ensuring diversity is of particular interest, since it makes it possible to have very small populations. And this is quite topical, since we used populations of sizes as small as 10 in the examples presented in the following.

2.3 Example on a simple mathematical function

Let us consider a game with 2 players A and B, with the following objective functions :

$$\begin{cases} f_A(x,y) = (x-1)^2 + (x-y)^2 \\ f_B(x,y) = (y-3)^2 + (x-y)^2 \end{cases}$$

The following section describes how to determine analytically and numerically a Nash equilibrium for these two.

2.3.1 Analytical solving

The definitions offered above for a Nash equilibrium are the mathematical ones, but they are not very practical to use. A way to find a Nash equilibrium is to use the notion of *rational reaction set*. Let D_A be the rational reaction set for A, and D_B the rational reaction set for B.

$$D_A = (\vec{x}, y) \in \bar{A} \times \bar{B} \text{ such as } f_A(\vec{x}, y) \leq f_A(x,y) \quad (1)$$
$$D_B = (x, \vec{y}) \in \bar{A} \times \bar{B} \text{ such as } f_B(x, \vec{y}) \leq f_B(x,y) \quad (2)$$

An intuitive insight on rational reaction sets is that they are the set of the best solutions a player can achieve for different strategies of his opponent.

D_A et D_B can be built by finding the x and y that satisfy the following equations:

$$\begin{cases} D_A = \{x \mid \frac{\partial f_A(x,y)}{\partial x} = 0\} \\ D_B = \{y \mid \frac{\partial f_B(x,y)}{\partial y} = 0\} \end{cases}$$

The Nash Equilibrium is the intersection of the two rational reaction sets D_A and D_B.

If we go back to the example we have presented, the rational reaction set D_A is the solution of the equation:

$$\frac{\partial f_A(x,y)}{\partial x} = 0$$

Which gives:

$$\frac{\partial f_A(x,y)}{\partial x} = 0 \Leftrightarrow 2(x-1) + 2(x-y) = 0 \Leftrightarrow y = 2x - 1$$

It follows that the rational reaction set D_A is the line $y = 2x - 1$. The second rational reaction set D_B is defined by the solution of the equation

$$\frac{\partial f_B(x,y)}{\partial y} = 0$$

.

$$\frac{\partial f_B(x,y)}{\partial y} = 0 \Leftrightarrow 2(y-3) - 2(x-y) = 0 \Leftrightarrow y = \frac{x+3}{2}$$

Hence, the rational reaction D_B is the line $y = \frac{x+3}{2}$.

Since the Nash Equilibrium is the intersection of the two rational sets, it can be determined by solving the system:

$$\begin{cases} y = 2x - 1 \\ y = \frac{x+3}{2} \end{cases} \implies \begin{cases} y = 2x - 1 \\ 3y = 7 \end{cases} \implies \begin{cases} x = \frac{5}{3} \\ y = \frac{7}{3} \end{cases}$$

Which means that the Nash Equilibrium is the point $\begin{pmatrix} \frac{5}{3} \\ \frac{7}{3} \end{pmatrix} = \begin{pmatrix} 1.66 \\ 2.33 \end{pmatrix}$.

2.3.2 Optimization by Nash GAs

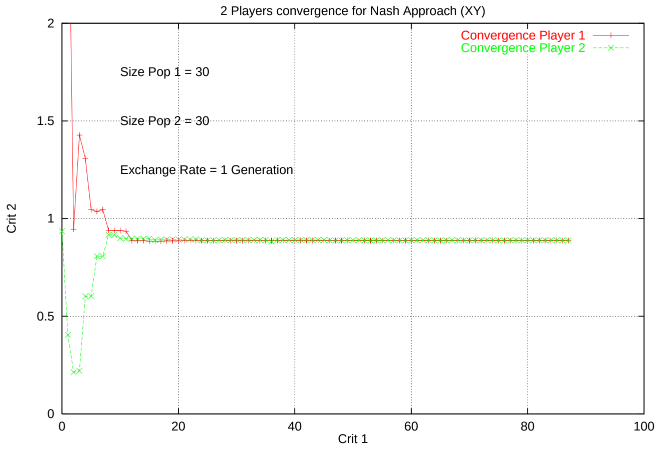

Figure 2: Computed Nash equilibrium

Figure 2 shows the evolution of the best chromosome for the 2 populations. Population 1 which optimizes criterion f_A converges towards 0.88. Population 2 which optimizes criterion f_B converges towards 0.88 as well. Both populations converge after only 50 generations.

Those values seem to be different from the ones found analytically, but this is because the figure shows the optimization process on the objective space. We can easily check that :

$$f_A(\frac{5}{3}, \frac{7}{3}) = 0.88 \text{ and } f_B(\frac{5}{3}, \frac{7}{3}) = 0.88$$

The point $\begin{pmatrix} 0.88 \\ 0.88 \end{pmatrix}$ is actually the point $\begin{pmatrix} f_A(1.66) \\ f_B(2.33) \end{pmatrix}$ in the plan of the criteria, which means that we actually find the right point – that is $\begin{pmatrix} 0.88 \\ 0.88 \end{pmatrix}$ – in the plan (x,y).

3 Testing on a more difficult Function

A more complex test is conducted on one of the multiple objective functions Deb specially constructed so as they would be hard to solve for GAs [Deb, 1998]. For 2 players, let us consider the two criteria f_1 and f_2 with (x_1, x_2) varying in $[0, 1] \times [0, 1]$.

$$f_1(x_1) = 4x_1$$

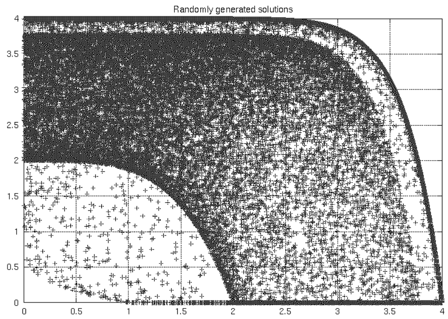

Figure 3: Random solutions repartition

$$f_2(x_1, x_2) = g(x_2) \cdot h(f_1(x_1), g(x_2))$$

with

$$g(x_2) = \begin{cases} 4 - 3\exp(-(\frac{x_2 - 0.2}{0.02})^2) & \text{if } 0 \le x_2 \le 0.4 \\ 4 - 2\exp(-(\frac{x_2 - 0.7}{0.2})^2) & \text{if } 0.4 \le x_2 \le 1 \end{cases}$$

$$h(f_1, g) = \begin{cases} 1 - (\frac{f_1}{g})^\alpha & \text{if } f_1 \le g \\ 0 & \text{otherwise} \end{cases}$$

$$\alpha = 0.25 + 3.75(g(x_2) - 1)$$

This problem has one local optimum Pareto set which is concave and one global optimum Pareto set which is convex. Figure 3 shows the distribution of f_1 and f_2 for 50000 random points. It is obvious that the problem has a local non-convex pareto front, and a global convex front that is much harder to find.

Figure 4 shows more in details those 2 fronts. The displayed fronts are evolved by a Pareto GA and will permit a few comparisons in the following.

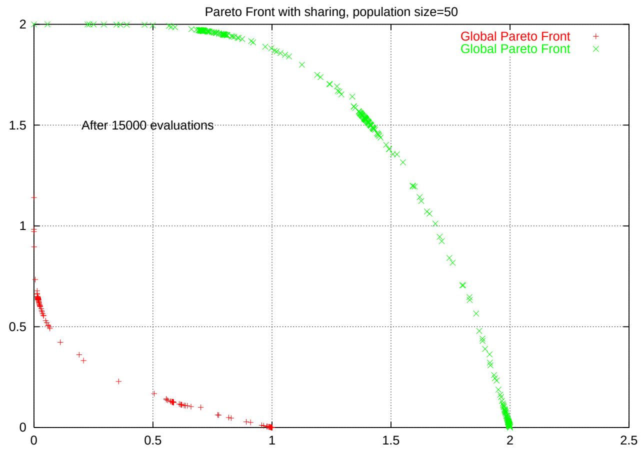

Figure 4: Global and local fronts evolved with a Pareto GA

All the results in the following sections are averaged out of 50 runs for each case.

3.1 Single objective

A single objective approach with a linear combination of the criteria yields the following result :

- 72% of the runs converge towards $\begin{pmatrix} x = 0.235033 \\ y = 0.200288 \end{pmatrix}$ which corresponds to $\begin{pmatrix} f_1 = 0.940134 \\ f_2 = 0.0156209 \end{pmatrix}$ in the plan of the criteria. This point is actually on the global front.
- 28% of the runs converge towards $\begin{pmatrix} x = 0.503219 \\ y = 0.683019 \end{pmatrix}$ which corresponds to $\begin{pmatrix} f_1 = 2.01288 \\ f_2 = 0.0060342 \end{pmatrix}$ in the plan of the criteria. This point is on the deceptive local front.

The GA used is a real-coded GA with a population of 20. The maximum authorized number of evaluations is 2000.

3.2 Pareto

We ran a Pareto based GA to see how well it would capture the global front. The Pareto GA we used is a real-coded algorithm based on NSGA, with ranking and sharing [Srinivas and Deb, 1995]. The population size is 50. Table 3.2 shows the percentage of times the algorithm manages to capture the global front, depending on the number of generations it is allowed to run.

Generations	40	100	200	300	400
Finds global front	60%	60%	90%	90%	100%

Table 1: Convergence towards the global front

Which means that the Pareto GA needs 400 generations of 50 solutions to ensure that it finds the global front in 100% of the cases.

3.3 Nash

Lastly, the same experiment is run with Nash GA. We use a population size of 10. For comparison purposes, the optimization is stopped after 2000 evaluations.

The results are that the Nash GA converges in 100% of the runs towards $\begin{pmatrix} x = 0 \\ y = 0.2 \end{pmatrix}$ which corresponds to $\begin{pmatrix} f_1 = 0 \\ f_2 = 1 \end{pmatrix}$ in the plan of the criteria. This point is actually on the global front, on the extreme left.

3.3.1 Comparison

If we compare the 3 approaches on a basis of 2000 evaluations, Table 3.3.1 shows that Nash GA is definitely the most robust approach as it always converges towards a point of the global.

Weighted GA	Pareto GA	Nash GA
72%	60%	100%

Table 2: Finding the global after 2000 evaluations

Of course, this comparison is a bit unfair towards Pareto GA, as it captures several points on the front, not a single one. Yet, it is interesting to note that to ensure robustness, a Pareto GA needs a large number of evaluations.

In the future, a possible development could actually consist in building a hybrid approach mixing a Nash approach to do the preliminary scouting, and then send over the solution to a Pareto GA so as it can exploit it to find the global front.

4 Application : Nozzle Optimization with Multiple Objectives

4.1 Problem definition

The above strategy is tested via the optimization of the shape of a nozzle for a transonic and a subsonic flow involving a shock. The problem is a reconstruction problem involving 2 target nozzles at 2 different Mach numbers:

1. the first target nozzle (Nozzle 1) is shown in figure 5. It is used with a low Mach number to produce a Mach number distribution shown in figure 7.
2. the second target nozzle (Nozzle 2) is shown in figure 6 and is used in a flow with a high Mach number to produce the second Mach number distribution shown in figure 7.

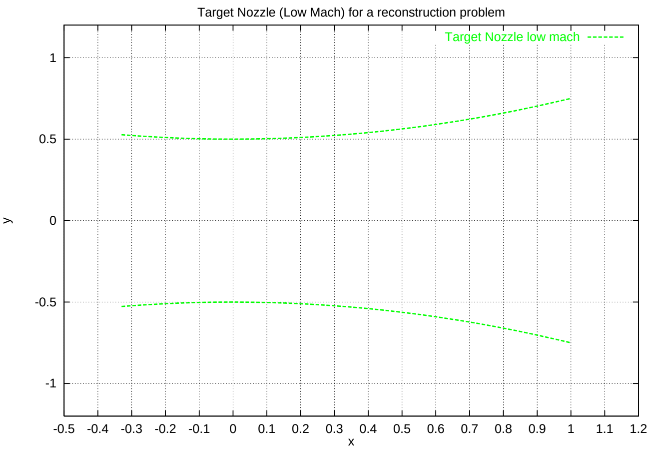

Figure 5: Nozzle target for low Mach

For each target nozzle, the corresponding Mach number distribution is computed, using a quasi-steady one-dimensional approximation for the flow. The equations are solved by a time marching technique using a CUSP scheme with an iterative solver [Jameson, 1995, Srinivas, 1999].

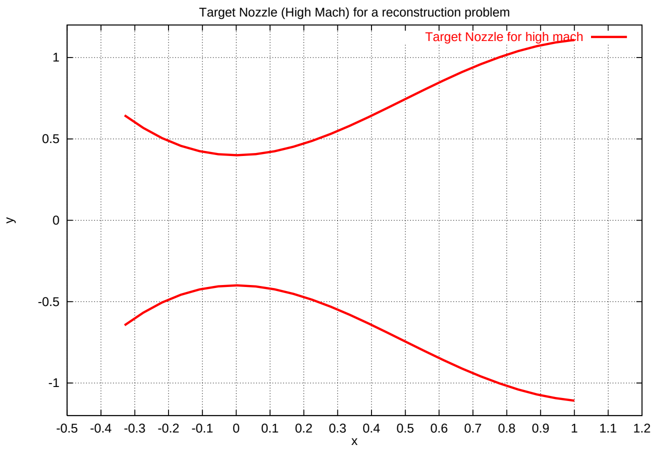

Figure 6: Nozzle target for high Mach

The objective of the optimization task for the GA is to try to find a nozzle that will be the best compromise between the 2 targets:

1. at low Mach, the Mach number number distribution should be as close as possible to that of Nozzle 1.

2. at high Mach, the Mach number number distribution should be as close as possible to that of Nozzle 2.

In other words, for each candidate nozzle, the CFD solver provides a Mach number distribution at low or high Mach, and it is then compared to the relevant target distribution.

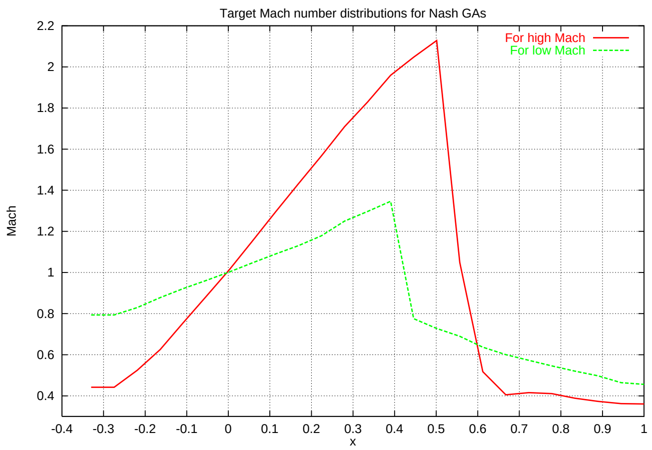

Figure 7: Target Mach distributions for high Mach and low Mach

4.2 Parameterization of the shape of the nozzle

The optimization we present is done by a real-coded Nash GA, with 2 players having a population of size 20 each. As the nozzle is symmetric, we optimize only the upper half of it. The shape of the nozzle is defined by a Bezier curve of order 8 (i.e. a polynomial curve of order 7). The definition of a Bezier curve of order n is:

$$x(t) = \sum_{i=0}^{n} c_n^i t^i (1-t)^{n-i} x_i, \;\; y(t) = \sum_{i=0}^{n} c_n^i t^i (1-t)^{n-i} y_i$$

where $c_n^i = \frac{n!}{i!(n-i)!}$ and (x_i, y_i) are control points of the curve, t is a parameter varying in [0,1].

The curve is defined by 8 control points $P^1, \cdots, P^8$. The x coordinates of the P_x^i are all distributed evenly in $[-0.33, 1]$.

The variables of the optimization are the y coordinates of the control points, $P_y^1, \cdots, P_y^8$. For the target Mach number distribution, we use 25 points along the nozzle for the grid. Let M_i be the value of the Mach number distribution of the candidate nozzle for point i. The fitness function is based on a least square method and depends on the player:

1. for Player 1, the solver uses a low Mach number and the fitness is:

$$f_1 = \sum_{i=1}^{25} (M_i - M_i^{Nozzle1})^2$$

2. for Player 2, the solver uses a high Mach number and the fitness is:

$$f = \sum_{i=1}^{25} (M_i - M_i^{Nozzle1})^2$$

4.3 Results

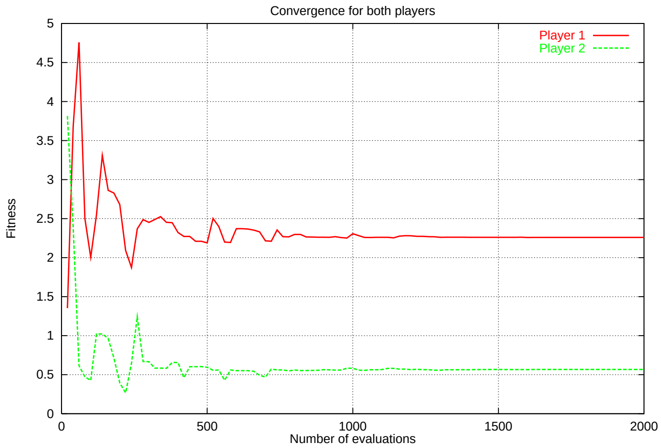

Figure 8: Convergence for both players

Figure 8 shows the convergence of both players in the fitness space. It is quite clear that each of them has reached a Nash equilibrium.

The solution nozzle found by Nash GA is displayed in figure 9.

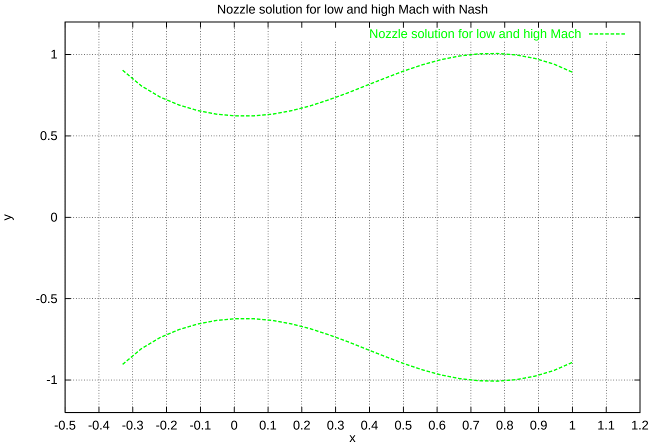

Figure 9: Nozzle solution for low and high Mach with Nash

To have a better idea of how the shape evolved compared to the two target shapes, figure 10 shows the upper half of Nozzle 1, Nozzle 2 and the nozzle solution. The most interesting point is to note that the solution nozzle is by no means a compromise of the two shapes but rather an entirely novel shape. This is quite promising, as the fitness was not based on the similarity to a given shape, but rather on the similarity to a Mach number distribution.

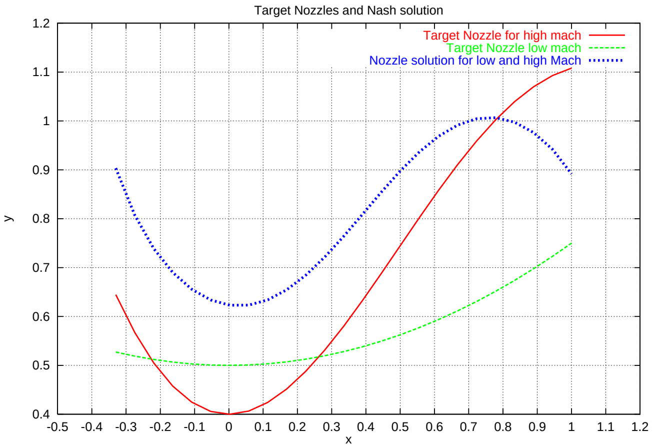

Figure 10: Target Nozzles and Nash solution

This similarity is actually show for both low and high Mach regimes. Figure 11 shows the Mach distribution of Nozzle 1 compared to that of the nozzle solution at low Mach. Figure 12 does the same with the Mach distribution of Nozzle 2 compared to that of the nozzle solution at nigh Mach.

5 Conclusion

From the tests and experiments described in this paper, it is clear that GAs offer a fast and robust alternative when it comes to multi-objective optimization. The point is that they seem to be more robust than cooperative optimizations such

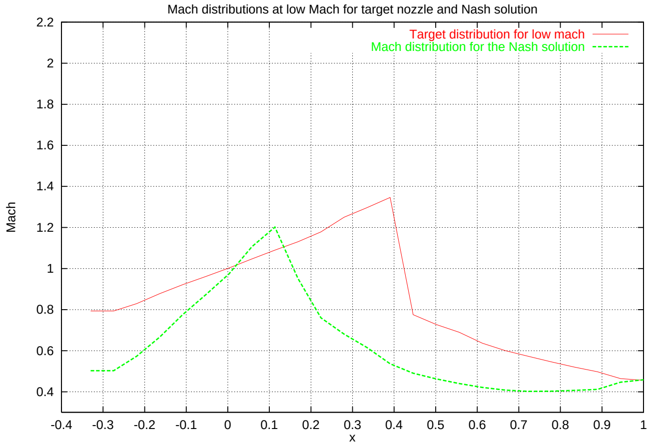

Figure 11: Mach distributions at low Mach for target nozzle and Nash solution

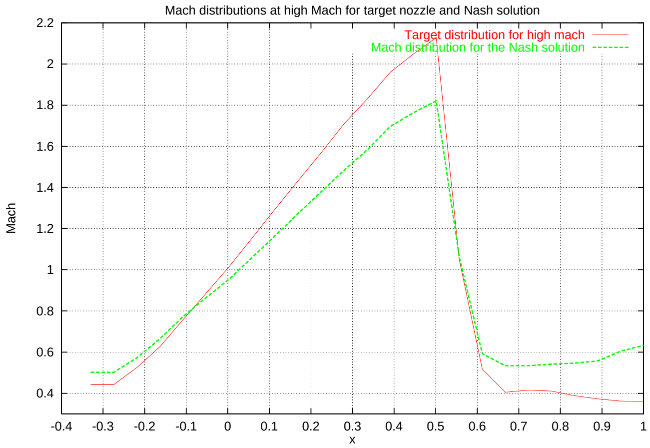

Figure 12: Mach distributions at high Mach for target nozzle and Nash solution

as Pareto GAs and easily parallelizable.

Another point of interest is that they offer solutions that are an equilibrium between 2 players, not necessarily a compromise. That means that new solutions can arise that are quite different from either target objectives.

Acknowledgments

We would like to thank Dr K. Srinivas, from the Dept of Aeronautical Engineering, University of Sydney, for providing the codes for the nozzle optimization problem.

Bibliography

[Bristeau et al., 1999] Bristeau, M.-O., Glowinski, R., Mantel, B., Periaux, J., and Sefrioui, M. (1999). Genetic algorithms for electromagnetic backscattering : Multioblective optimization. In Rahmat-Samii, Y. and Michielssen, E., editors, *System Design Using Evolutionary Optimization: Genetic Algorithms*. John Wiley.

[Deb, 1998] Deb, K. (1998). Multi-objective genetic algorithms: Problem difficulties and construction of test problems. Technical report, Department of Computer Science / LS11, University of Dortmund.

[Fonseca and Fleming, 1995] Fonseca, C. M. and Fleming, P. J. (1995). An overview of evolutionary algorithms in multiobjective optimisation. In *Evolutionary Computation*, volume 3, pages 1–16.

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

[Jameson, 1995] Jameson, A. (1995). Analysis and design of numerical schemes for gas dynamics, 2: Artificial diffusion and discrete shock structure. *Int. J. Comput. Fluid Dynamics*, 5:1–38.

[Makinen et al., 1996] Makinen, R., Neittaanmaki, P., Periaux, J., Sefrioui, M., and Toivonen, J. (1996). Parallel genetic solution for multiobjective MDO. In *Parallel CFD 96*, Capri. Elsevier.

[Michalewicz, 1992] Michalewicz, Z. (1992). *Genetic algorithms + data structures = evolution programs*. Artificial Intelligence. Springer-Verlag, New York.

[Mühlenbein et al., 1991] Mühlenbein, H., Schomisch, M., and Born, J. (1991). The parallel genetic algorithm as function optimizer. *Parallel Computing*, 17(6-7):619–632.

[Nash, 1950] Nash, J. F. (1950). Equilibrium points in n-person games. *Proc. Nat. Acad. Sci. U.S.A*, 36:46–49.

[Nash, 1951] Nash, J. F. (1951). Noncooperative games. *Annals of Mathematics*, 54:289.

[Poloni, 1995] Poloni, C. (1995). Hybrid ga for multi objective aerodynamic shape optimization. In Galan, M., Winter, G., Periaux, J., and Cuesta, P., editors, *Genetic Algorithms in Engineering and Computer Science*, pages 397–415, Las Palmas, Spain. EuroGen 95, John Wiley.

[Sefrioui, 1998] Sefrioui, M. (1998). *Algorithmes Evolutionnaires pour le calcul scientifique. Application l'electromagnetisme et la mcanique des fluides numriques*. PhD thesis, Universite Pierre et Marie Curie, Paris.

[Sefrioui et al., 1996a] Sefrioui, M., Periaux, J., and Ganascia, J.-G. (1996a). Fast convergence thanks to diversity. In Fogel, L. J., Angeline, P. J., and Bäck, T., editors, *Proceedings of the Fifth Annual Conference on Evolutionary Programming*, San Diego, CA. IEEE Computer Society Press, MIT press.

[Sefrioui et al., 1996b] Sefrioui, M., Periaux, J., and Mantel, B. (1996b). RCS multi-objective optimization of scattered waves by active control elements using GA. In *Fourth International Conference on Control, Automation, Robotics and Vision*, Singapore.

[Srinivas, 1999] Srinivas, K. (1999). Computation of cascade flows by a modified cusp scheme. *Computational Fluid Dynamics Journal*, 8(2):285–295.

[Srinivas and Deb, 1995] Srinivas, N. and Deb, K. (1995). Multiobjective optimisation using non-dominated sorting in genetic algorithms. In *Evolutionary Computation 2 (3)*, pages 221–248.