Parametric Study of a Genetic Algorithm using a Aircraft Design Optimization Problem

Andre C. Marta
Dept. of Aeronautics and Astronautics
Stanford University
Stanford, California 94305
acmarta@stanford.edu

ABSTRACT

This work shows how a preliminary aircraft design can be achieved by means of genetic algorithms (GA). The aircraft major parameters are mapped into a chromosome like string. These include the wing, tail and fuselage geometry, thrust requirements and operating parameters. GA operators are performed on a population of such strings and natural selection is expected to occur. The design performance is obtained by using the aircraft range as the fitness function. Different GA parameters and selection methods – fitness and ranking – are tested and their impact on the algorithm efficiency is analyzed. The constraints implementation is also studied.

1. Introduction

At first glance, the definition of the best aircraft design is very simple: the fastest, most efficient, quietest, most inexpensive airplane with the shortest field length. Unfortunately, such an airplane cannot exist. One can only make one thing best at a time, as illustrated in figure 1. The most inexpensive airplane would surely not be the fastest, as well as the most efficient would not be the most comfortable. In other words, the overall airplane is always a compromise in some sense. Various quantities are included in a function termed the figure of merit or objective, such as weight, flight controls, structures, manufacturing, aerodynamics, noise and propulsion characteristics, whose relative weights depends on the intended application for the aircraft.

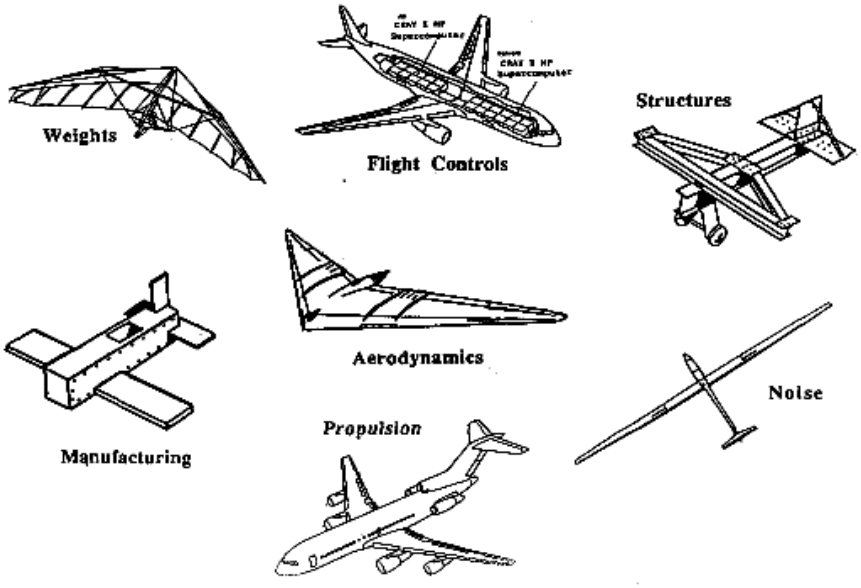


Figure 1 Make one thing best at a time

Once the definition of best has been decided, one must find a way of relating the design variables to the goal. This process is show schematically in figure 2.

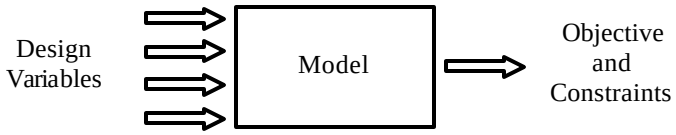


Figure 2 Modeling

In aircraft design, this process can be extremely complex. The number of parameters needed to completely specify an airplane is astronomical, so a combination of approximation, experience and statistical information on similar aircraft has to be used to reduce the number of design variables.

There are several methods by which one chooses the design variables leading to the best design. Two approaches to optimization are commonly used: 1) Analytical: this approach is very useful for fundamental studies but requires great simplification (often oversimplification); 2) Numerical: in most aircraft design problems, the analysis involves iterations, table look-ups or complex computations. In these cases, direct search methods are employed, namely grid searching, random searches, nonlinear simplex and gradient methods.

In aircraft design, problems are often constraint-bounded, that is, the constraints dictate the values of the design variables. When many constraints are active at the optimum, the value of the gradient (for continuously differentiable objective functions) is not zero. For these problems, the methods have to be modified. One approach to constrained optimization is simply to add a penalty to the objective function when the constraints are violated.

2. Background

In the aircraft design field, many optimization works have been done, ranging from simple parametric studies to highly detailed configurations, involving thousands of design variables.

Today, the aircraft design community no longer seeks the best aerodynamic or the best structural solutions, but rather the optimal overall solution, in what is called multidisciplinary optimization (MDO). Typically, such works make use of gradient methods to find the optimal solution, requiring massive computations to determine the numerical derivatives, as seen in Alonso, Martins and Reuther (2003).

When the objective function is not continuous, the gradient based methods cannot be immediately applied. In such cases, the use of other search algorithms might be considered. Within this context, genetic algorithms (GA) are becoming more and more popular in many science fields when all other conventional algorithms seem to fail. However, in the aircraft design field, GAs are yet far from being widely used despite their robustness and proven efficiency.

3. Statement of the Problem

In order to show how an aircraft design problem can be solved by means of GA, a preliminary design problem was implemented, using a restricted number of design variables and the Breguet equation as the objective function. This equation expresses the aircraft range *R*, that is, maximum distance flown, given the propulsive, aerodynamic and structural characteristics of the aircraft.

$$R = \frac{V}{SFC}\frac{L}{D}\ln\left(\frac{W_i}{W_f}\right) \qquad \text{(eq. 1)}$$

In equation 1, *V* is the cruise velocity, *SFC* is the specific fuel consumption at cruise speed and altitude, *L* is the airplane lift, *D* is the airplane drag, *Wi* is the initial airplane weight and *Wf* is the final airplane weight. Further details can be seen in Shevell (1983).

The right-hand side terms in equation 1 can be estimated by combining several estimates described in Kroo (2003), using the following design variables: fuselage diameter, fuselage length, cruise altitude, take-off weight, wing span, wing chord, horizontal tail span, vertical tail span, angle of attack, location of wing and location of the tail.

As stated in section 1, each airplane is designed to meet different constraints, which are imposed by the specified requirements. In this paper, the constraints selected refer to a medium size airplane, typically a regional jet, flying at subsonic speed. There were seven constraints, in addition to the allowable range for each of the eleven design variables, whose values are given in table 1.

Table 1 Range of design variables and other constraints

Design variable	Admissible values
Fuselage diameter	$60 < d < 200$ in
Fuselage length	$70 < l < 150$ ft
Cruise altitude	$10{,}000 < h < 50{,}000$ ft
Take-off weight	$60{,}000 < tow < 100{,}000$ lbs
Wing span	$50 < b < 150$ ft
Wing chord	$5 < c < 20$ ft
Horizontal tail span	$20 < bh < 70$ ft
Vertical tail span	$5 < bv < 20$ feet
Angle of attack	$1 < alpha < 16$ deg
Location of wing	$0.2 < xwoverl < 0.7$
Location of tail	$0.5 < xtoverl < 1.0$

Others constraints	Admissible values
Number of seats	$60 < seats < 80$
Mach number	$M < 0.5$
Tail after wing	$xwoverl < xtoverl$
Horizontal tail volume	$0.4 < Vh < 1.0$
Vertical tail volume	$0.04 < Vv < 0.09$
Moment coefficient about c.g.	$-0.1 < cm\_cg < 0.1$
Zero fuel weight	$zfw < tow$

The chosen design variables together with ten additional constants (fuselage nose and tail fineness, seat pitch, Osvald efficiency factor, wing sweep angle, airfoil thickness-to-chord ratio, horizontal and vertical tail aspect ratio, number of engines and engine specific fuel consumption at sea level) make possible to compute all the necessary characteristics to evaluate equation 1.

4. Methods

The eleven design variables were mapped into a chromosome like string. To achieve the desired accuracy, different granularities were used for each one of the variables, corresponding to different length bit groups. Putting all the bit groups together, resulted in a seventy one bit long chromosome. The detailed mapping can be seen in table 2.

Table 2 Mapping of the design variables into the chromosome string

Design variable	Admissible values	Granularity	Number of bits	Bit position in chromosome
Fuselage diameter	$60 < d < 200$ in	128	7	1 to 7
Fuselage length	$70 < l < 150$ ft	128	7	8 to 14
Cruise altitude	$10{,}000 < h < 50{,}000$ ft	128	7	15 to 21
Take-off weight	$60{,}000 < tow < 100{,}000$ lbs	128	7	22 to 28
Wing span	$50 < b < 150$ ft	256	8	29 to 36
Wing chord	$5 < c < 20$ ft	128	7	37 to 43
Horizontal tail span	$20 < bh < 70$ ft	128	7	44 to 50
Vertical tail span	$5 < bv < 20$ feet	64	6	51 to 56
Angle of attack	$1 < alpha < 16$ deg	32	5	57 to 61
Location of wing	$0.2 < xwoverl < 0.7$	32	5	62 to 66
Location of tail	$0.5 < xtoverl < 1.0$	32	5	67 to 71

From the specification of the problem in section 3 and the mapping defined in table 2, the tableau for the problem was then completed (table 3).

Table 3 Tableau for the preliminary aircraft design problem

Objective:	Find the globally optimum combination of eleven aircraft parameters.
Representation scheme:	• Structure = fixed length string; • Alphabet size K = 2 (binary); • String length L = 71; • Mapping from points in search space of the problem to structures in the population = (see table 2)
Fitness cases:	Only one.
Fitness:	Aircraft range with penalty for constraint violations. More is better.

Parameters:	• Population size M = 200; • Maximum number of generations G = 100.
Termination criteria:	The genetic algorithm has run for G generations.
Result designation:	The best-so-far individual in the population.

At this point, a special remark must be made concerning the constraints handling. The constraints were taken into consideration by penalizing the fitness value. Two cases were included: severe constraints and non-severe constraints. The severe constraints included the positioning of the tail after the wing and the zero fuel weight less than the take-off weight (meaning positive fuel weight allowance). If one severe constraint was violated then the fitness value was set to zero, whereas if one of the non-severe constraints were violated, the fitness value was decreased by an amount proportional to the constraint deviation from its desired bounds. An all-severe constraint approach was initially adopted, but that reduced considerably the number of valid individuals in the initially random generated population (less than 5% of valid individuals), causing an unacceptable loss of genetic information.

The population size was set so that a genetically diverse initial population could be generated which, by the nature of the GA, improves its efficiency. Several trials were made starting with a population size of 50 individuals, but only when a size of 200 individuals was reached did the initial population have a reasonable number of diversely valid individuals.

The maximum number of generations was determined experimentally so that the fitness of the best-of-generation individual would reach a steady value.

The GA used to implement the specified problem was the Genetic Search implementation System (GENESIS Version 5.0) by John J. Grefenstette. This constitutes an independent optimizer, in which the user as only to provide the fitness evaluation function routine for the particular problem of interest. Its detailed documentation can be found at Grefenstette (1984). Hence, the problem specific fitness evaluation function was coded so that, using the design variables specified in table 2 as inputs, the fitness measure provided by equation 1 was obtained. Further detail concerning the code may be obtained by contacting the author.

Detailed information regarding the GA theory can be found in Goldberg (1989) and Koza (1992).

5. Results

In order to evaluate the influence of the GA parameters, several runs were made. They included different seeds for the random number generator, different crossover and mutation probability rates and different selection procedures. These lead to six distinct runs, whose parameters are specified in table 4. Repeated runs were made but the results obtained were the same, as the random number generator used depends on the seed provided by the user.

Table 4 GA parameters for the different runs

Case	Random seed	Crossover probability	Mutation probability	Selection procedure
1	1234567	60 %	3.3 %	Relative fitness spinning wheel
2	123456789	60 %	3.3 %	Relative fitness spinning wheel
3	123456789	60 %	3.3 %	Ranking algorithm, R=0.75
4	123456789	60 %	3.3 %	Ranking algorithm, R=0.0
5	123456789	80 %	3.3 %	Relative fitness spinning wheel
6	123456789	60 %	10 %	Relative fitness spinning wheel

Each experiment consisted of 20,000 fitness evaluations, as a result of a population of 200 individuals run for 100 generations.

The GA software was installed in a Sun Enterprise 6500 workstation, running Solaris 8 operating system, with sixteen 400 MHz processors and 16 GB of RAM. The single processor run time of each experiment was about 24 minutes.

The evolution of the best-of-generation and the generation average for each run are presented in figure 3.

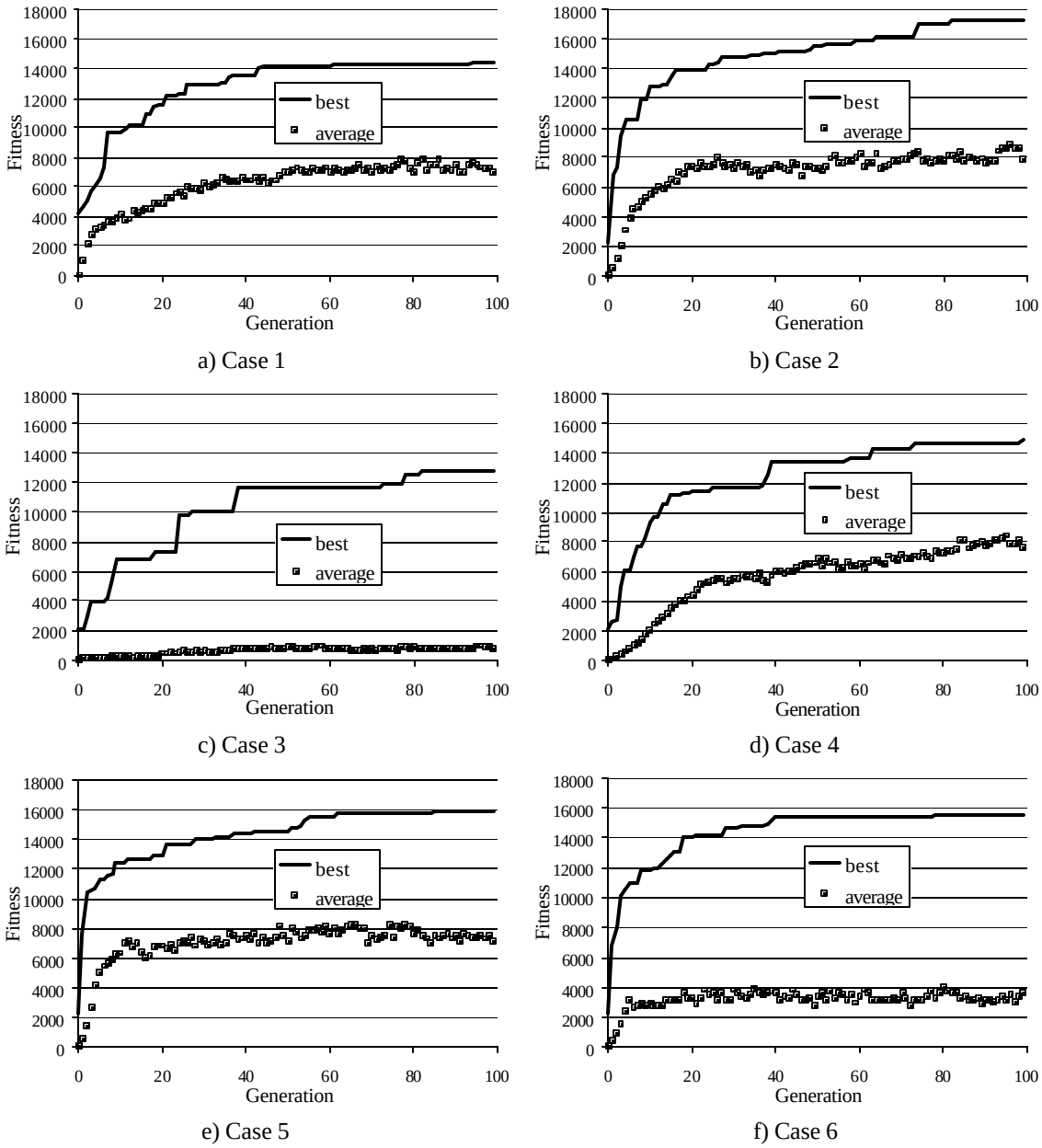
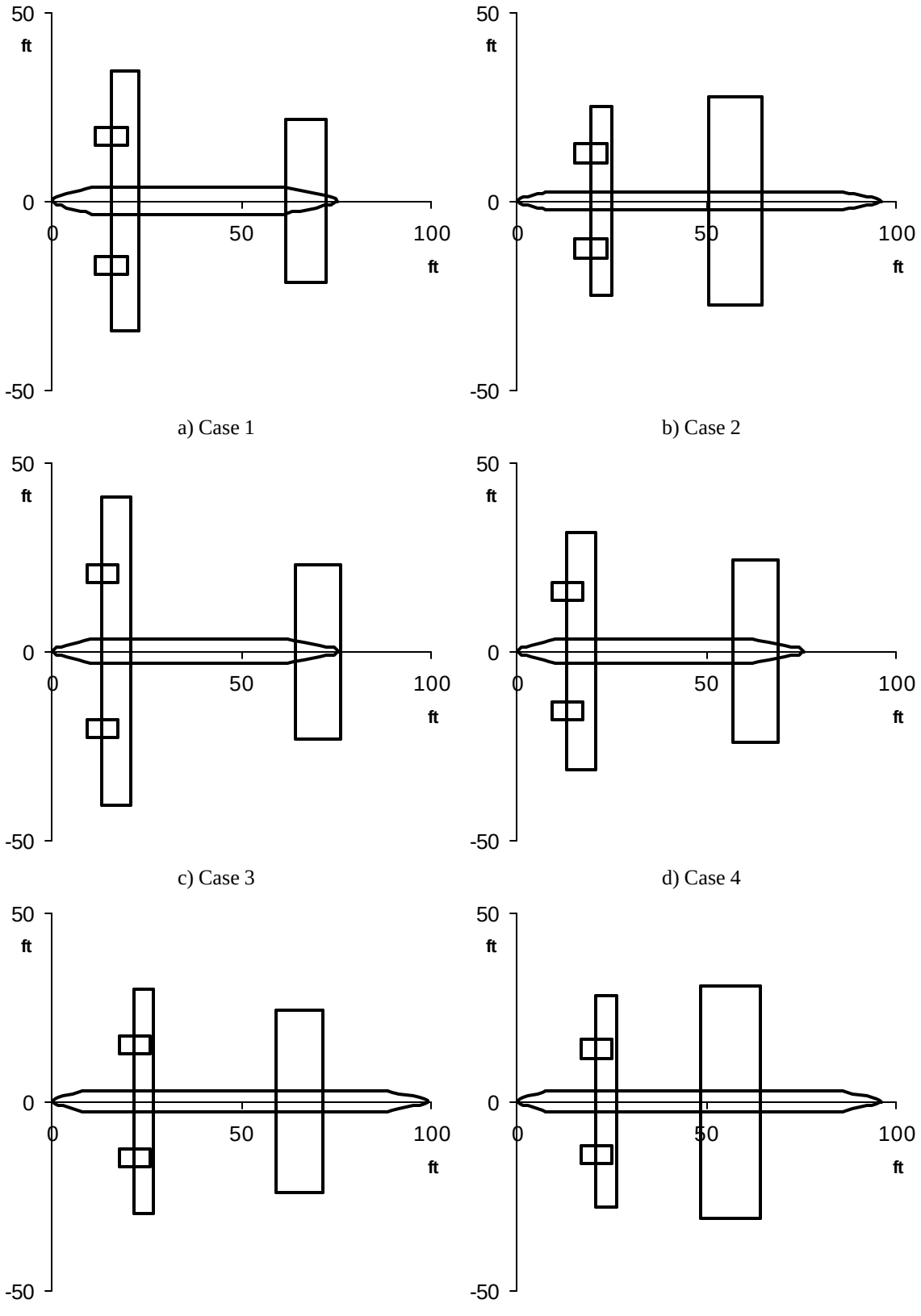


Figure 3 Performance curves for the aircraft design problem

The best-of-run and the generation average fitness values after 100 generations are summarized in table 5, where the fitness values (aircraft range) are expressed in nautical miles. The aircraft configurations corresponding to the best-of-run are shown in figure 4.

Table 5 Best-of-run fitness and generation average fitness after 100 generations

Case	Best-of-run (generation produced)	Average after 100 generations
1	14,372 n.mi (generation 94)	6,956 n.mi
2	17,260 n.mi (generation 82)	7,865 n.mi
3	12,721 n.mi (generation 82)	834 n.mi
4	14,826 n.mi (generation 99)	7,674 n.mi
5	15,831 n.mi (generation 85)	7,135 n.mi
6	15,562 n.mi (generation 78)	3,672 n.mi



a) Case 1

b) Case 2

c) Case 3

d) Case 4

e) Case 5

f) Case 6

Figure 4 Aircraft configuration corresponding to the best-of-run individual after 100 generations

The variable values corresponding to the explicit constraints included in the fitness evaluation function are summarized in table 6.

Table 6 Values of constraints for the best-of-generation after 100 generations

Constraints	Admissible values	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
Number of seats	$60 < seats < 80$	60	60	60	60	62	60
Mach number	$M < 0.5$	0.483	0.450	0.453	0.500	0.505	0.461
Tail after wing	$xwoverl < xtoverl$	0.232< 0.855	0.216< 0.565	0.200< 0.887	0.200< 0.790	0.232< 0.629	0.232< 0.548
Horizontal tail volume	$0.4 < Vh < 1.0$	0.405	0.402	0.409	0.469	0.409	0.417
Vertical tail volume	$0.04 < Vv < 0.09$	0.060	0.052	0.085	0.059	0.076	0.063
Moment coef. about c.g.	$-0.1 < cm\_cg < 0.1$	-0.099	-0.094	-0.099	-0.098	-0.095	-0.096
Zero fuel weight	$zfw < tow$	42,422<99,685	41,554<100,000	44,983<99,685	42,185<100,000	43,007<100,000	43,780<100,000

The detailed discussion of all these results can be found in the next chapter.

6. Discussion of the Results

As a first remark, the optimal values obtained for the aircraft range, the fitness measure used in the proposed problem, are somehow overestimated (table 5). This was due to the fact that the implemented fitness function had several simplifications regarding the airplane drag estimate, making the simulated airplane to have underestimated total drag, thus better aerodynamic properties than in reality planes have.

Based on the six different experiments run, the effects of different GA parameters on the algorithm performance can be measured. In addition, the constraint implementation efficiency is also analyzed. These analysis are made in the following subchapters.

6.1. Effect of random seed

The effect of the seed for the random number generator can be analyzed by comparing cases 1 and 2.

From figure 3 a) and b), one can see that the seed had a significant impact on the results. In a way, the chosen seed determines the initial population, and therefore, the initial genetic information available to the GA. The richer this information is, the more efficiently the algorithm performs. From these figures, the random seed in case 2 lead to a genetically wealthier initial population, which in turn, made the optimum search more efficient. Both the best-of-generation and the generation average fitness grew significantly faster in case 2 than in case 1. This is particularly true for the first simulated generations.

Looking at table 5, again comparing cases 1 and 2, it can be seen the importance of the initial population for the final results. Although the final (generation 100) generation average fitness differ by about 10%, the best-of-run does differ by 20%.

It is also very interesting to see the aircraft configuration that resulted from cases 1 and 2. Looking at figures 4 a) and b), it seems that the poorly performing experiment, namely case 1, lead to a very conventional aircraft design, while the best performing experiment (case 2), lead to a somehow revolutionary design. Indeed, this revolutionary design is nothing but the so called canard configuration, where the horizontal stabilizer is located in front of the main wing as opposed to the conventional design. The way the GA got this solution was to move the wing to the front while reducing its size, and moving the tail to the middle of the airplane fuselage while increasing its size. This was indeed a very pleasant surprise.

6.2. Effect of selection procedure

By comparing cases 2, 3 and 4, it is possible to evaluate the effect of the selection procedure on the GA efficiency.

In case 2, the selection procedure, also called reproduction, is based on the idea of allocating each structure to a portion of a spinning wheel proportional to the structure's relative fitness. Then, by spinning the wheel, the structures to be copied into the new generation population are selected. This is the conventional reproduction usually found in every GA .

The ranking algorithm was used in cases 3 and 4. In these cases, the selection of structures to be reproduced to the new generation is based on the ranking that the structure has amongst all the individuals of the population of the present generation. Despite this algorithm prevents that the above average individuals take over the entire population, it makes the algorithm convergence slower. This is clearly seen in figures 3 b), c) and d).

The cases 3 and 4 differ in the parameter R (not to be confused with the range fitness function *R*). This parameter is used in GENESIS to rank the structures in the population. The code attributes the value R to the worst individual in the population and 2-R to the best. Consequently, in case 3 (R=0.75), the ranking varies between 0.75 and 1.25, whereas in case 4 (R=0), it ranges from 0 to 2. As so, in case 3, the individuals are much more evenly ranked, which causes the selection operator to more frequently reproduce not so fitted individuals from the population. This is the cause for the very poor efficiency of the corresponding run (see figure 3 c).), in which the generation average fitness seems not to improve as the number of generations increase. However, the best-of-generation does improve as the number of generations increase since all the runs were made using the elitist selection strategy, which stipulates that the best performing individual in the population is reproduced to the next generation. Consequently, by means of crossover and mutation, the best-of-generation improves in the long run. This is very explicit in figure 3 c), in which the best-of generation fitness plot resembles a step function, keeping its value constant for quite a significant number of generations between improvements.

Comparing the run using the ranking selection algorithm with R=0.75 (case 4) and the run using the standard fitness proportional selection algorithm (case 2), it can be seen that the generation average fitness does increase slower for the ranking selection algorithm but it keeps increasing as the number of generations increase, not showing the steadiness that the standard selection algorithm typically shows (figure 3 b).). As far as the average fitness is concerned, this leads to very close results after 100 generations (see table 5), but it is expected that the ranking selection algorithm would lead to better generation average fitness in the long run. Looking at the aircraft configurations for the best-of-run for these two cases (figure 5 b) and d)), it can be seen that the ranking algorithm (case 4) lead to a more conventional or conservative design.

6.3. Effect of crossover probability

The effect of the crossover probability on the algorithm efficiency can studied by comparing cases 2 and 5. These two cases had all but one parameter the same, in particular, the probability of crossover. In case 2, a conservative value of 60% was used, while a higher value of 80% was used in case 5.

Looking at the evolution of the best-of-generation and generation average fitness shown in figures 3 b) and e), it can be seen that the higher probability of crossover leads to a faster initial fitness increase of both indicators, but the more conservative crossover probability case reaches slightly higher (about 10%) final value indicators. The aggressive crossover probability value of 80% seems to cause some loss of good individuals that are destroyed from one generation to the other due to the crossover operator, which might explain the slight poorer long run results as compared to the conservative crossover probability parameter value.

As seen in figures 5 b) and e), the resulting aircraft configurations do not differ significantly, but it is clear that the aggressive crossover probability case was not reached the refined configuration of its conservative counterpart, since it is not yet converted into a canard configuration but rather an undefined dual wing configuration.

6.4. Effect of mutation probability

Different mutation probability values were used in cases 2 and 6. While a conservative value of 3.3% was used in the first case, an aggressive value of 10% was used in the second case.

One effect that can immediately be seen by comparing figures 3 b) and f) is that the generation average fitness is considerably lower for the high mutation rate case. Despite the mutation produces, by nature, new schema that may not exist in previous generations, thus broadening the search space, it can also destroy good individuals. By doing so, the average fitness is highly penalized, as seen in figure 3 f) or table 5.

Although the average fitness is lower for the aggressive mutation rate case, its best-of-run fitness value is still acceptable, when compared to the best of all fitness value obtained. This reinforces the point that the mutation has the advantage of creating new schema in the search space. From figure 4 f), it can be seen that by broadening the search space, the innovative canard wing configuration was also found.

6.5. Constraints implementation

As explained in chapter 4, the constraints were classified as either severe or non-severe. The severe constraints, whose violation caused the fitness value to be set to zero, thus highly penalized, were expected to be fully satisfied by the best-of-run solution. From table 6 it is possible to confirm that all severe constraints, namely the positioning of the tail after the wing and the zero fuel weight less than the take-off weight, are indeed satisfied.

All the non-severe constraints might or might not be fully satisfied by the fact that the fitness value was penalized proportionately to the constraint deviation. It was expected that some of these constraints might be slightly violated. This only happened for the best-of-run in case 5, where the Mach number exceeded the constraint by 1%, whereas all other non-severe constraints were fully satisfied.

It is also interesting to see that the algorithm does maximize the objective function, making active the necessary constraints. As seen in table 6, the number of seats, the wing positioning, the horizontal tail volume and the moment

coefficient about the center of gravity were set to their minimum allowable value, while the Mach number and the take-off weight were set to their maximum allowable value. The only constraint not active was the vertical tail volume.

7. Conclusion

The entire search space of the problem expressed in table 3 has a size of 2^71 = 2.36e21. Such vast search space would make virtually impossible for a random search algorithm to find the optimal value. However, using a genetic algorithm it was possible to solve the problem with only 20,000 = 2e4 fitness evaluations, representing an impressive saving of seventeen orders of magnitude.

Being the GA a probabilistic algorithm, its results are not deterministic, that is, each GA run might lead to different results. The different seed cases run were an example of such probabilistic behavior. In order to improve the chances of reaching the global optimum, it is then mandatory to run several experiments with different initial populations.

From the selection procedure comparison, between the fitness roulette wheel and the ranking proportional algorithm, better results were obtained using the former. When the ranking proportional selection procedure was used with R=0, its behavior resembled the greedy operator, in which, only the best individuals are reproduced from one generation to the next. As expected, this leads to low efficiency as the genetic information contained within the population is greatly reduced.

Although a high crossover rate is desired in order to combine highly fitted schema, when this value becomes excessive, 80% in the runs made, the efficiency of the GA can actually decrease as other higher fitted individuals might be destroyed.

Being the aircraft design problem a highly nonlinear problem, the mutation rate might have a significant impact in the discovery of innovative solutions by broadening the search space. This was seen when making the run with a 10% mutation probability rate, which resulted in the discovery of the not so conventional canard configuration.

As far as the constraints implementation is concerned, it was clear that they should be implemented in such a way that the fitness of the corresponding individual is penalized proportionately to the deviation, and not simply set the fitness to zero, as was implemented in the initial experiment trials. Since the problem solved had several constraints, if the fitness was set to zero when a constraint was violated, the number of valid individuals in the initial population would be a very small percentage, about 5%, of the total population, thus eliminating too much genetic information.

Still regarding the constraints, the GA efficiency was here demonstrated as almost all constraints were active. This proves that the GA is in fact a very efficient search algorithm.

The example run referred to a small regional jet preliminary design. However, by simply changing the search space, that is, the domain of the design variables, and the explicit constraints included in the fitness evaluation function, new types of aircrafts can be readily designed. This ease of implementing new problems is true not only with aircraft design problems but also with all other type of problems solved using GA.

8. Future Work

Future work might be divided into two distinct tasks: 1) improve the GA efficiency; 2) extend the design variables and implement a higher fidelity fitness evaluation function.

Concerning the GA efficiency, there is still work to be done concerning both the parametric optimization, namely the optimal crossover and mutation rates, and further study of the selection procedure. The later might include the tournament selection, modified fitness wheel and greedy selection algorithms. Also, a parallel implementation of the GA should be considered since the relatively small problem solved in this paper took almost half an hour to run 100 generations with a population size of 200. For higher fidelity simulation problems, with a very large number of design variables and much more complex fitness evaluation functions, both the search space and the individual fitness evaluation time are considerably increased, leading to a much larger overall execution time, thus the parallel processing need. There is also work to be done regarding the trade-off between the population size and the number of generations to be run. It may be interesting to investigate if a further increase of the population size would lead to a decrease in the number of generations necessary for the problem to converge. The ultimate goal is to minimize the number of fitness evaluations (population size X number of generations) necessary to obtain the solution.

The problem solved in this paper was a preliminary aircraft design, as it is called in the field, in which only the major aircraft characteristics are defined. After this is done, it is time to go for the detailed aircraft design. In order to do so, the design variables have to be greatly increased, to the order of thousands, and the fitness evaluation function has to be adjusted accordingly. Most likely, an interface between the GA solver and a combined structural and fluid flow solver, which takes the role of the fitness evaluation function, has to be developed. So far no detailed aircraft design has been solved using a GA, in a field where the gradient methods still dominate, what might constitute an interesting field of research.

References

Alonso, J., Martins, J., and Reuther, J. 2003. *High-Fidelity Aero-Structural Design of Complete Aircraft Configurations with Aeroelastic Constrains*, 16th AIAA Computational Fluid Dynamics Conference, AIAA Paper AIAA-2003-3429, Orlando, FL, June 23-26, 2003.

Goldberg, David E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley Pub. Co..

Grefenstette, John J. 1984. *A user's guide to GENESIS*. Vanderbilt University Computer Science Department technical report CS-83-11. 1984.

Grefenstette, John J. 1984. GENESIS: a system for using genetic search procedures. *Proceedings of the Conference on Intelligent Systems and Machines* Rochester, MI, Pages 161–165.

Koza, John R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press.

Kroo, Ilan 2003. *Aircraft Design: Synthesis and Analysis*. Sanford, CA: Desktop Aeronautics Inc..

Shevell, Richard S. 1983. *Fundamentals of Flight*. Englewood Cliffs, NJ: Prentice Hall.