AERODYNAMIC AND AEROACOUSTIC OPTIMIZATION OF AIRFOILS VIA A PARALLEL GENETIC ALGORITHM

Brian R. Jones[*] William A. Crossley[†] Anastasios S. Lyrintzis[‡]
Purdue University
West Lafayette, IN 47907-1282

ABSTRACT

A parallel genetic algorithm (GA) was used to generate, in a single run, a family of aerodynamically efficient, low-noise rotor blade designs representing the Pareto optimal set. The *n*-branch tournament, uniform crossover genetic algorithm operates on twenty design variables, which constitute the control points for a spline representing the airfoil surface. The GA takes advantage of available computer resources by operating in either serial mode or "manager/worker" parallel mode. The multiple objectives of this work were to maximize lift-to-drag of a rotor airfoil shape and to minimize an overall noise measure including effects of loading and thickness noise of the airfoil. Constraints are placed on minimum lift coefficient, pitching moment and boundary layer convergence. The program XFOIL provides the aerodynamic analysis, and the code WOPWOP provides the aeroacoustic analysis. The Pareto-optimal airfoil set has been generated and is compared to the performance of a typical rotorcraft airfoil under identical flight conditions.

NOMENCLATURE

c_d	Section drag coefficient
c_l	Section lift coefficient
c_m	Section moment coefficient about ¼ chord
c_p	Pressure coefficient
c_f	Skin friction coefficient
f	Fitness function
g_i	Constraint function
M	Chord Mach number
n	Number of tournament branches (objectives)
N_{pop}	Population size
P_j	Penalty function
P_{mut}	Mutation probability
R_j	Penalty coefficients
Re	Chord Reynolds number
x/c	Normalized airfoil station
x_s/c	Normalized chordwise separation location
y/c	Normalized airfoil ordinate
α	Geometric angle of attack

INTRODUCTION

Since John Holland originally described the concept of a genetic algorithm (GA) in the 1960's, it has increasingly been applied to engineering design and optimization problems[1]. The GA's ability to deal with large numbers of continuous and integer design variables as it searches highly multimodal, discontinuous design spaces frequently provides an edge over more traditional, gradient based optimization methods when analyzing certain problems. The GA maintains another advantage by locating the global optimum without requiring an initial design point. While not an optimization technique *per se*, the GA possesses many characteristics that allow it to be applied to optimization problems with great success. Based on Darwin's "survival of the fittest" evolutionary concept, the GA employs selection, crossover and mutation operators to "evolve" a population of individual designs over many generations. Since each individual is actually a coding of the relevant design variables, this process ultimately transforms an initial pool of randomly selected designs into a population of individuals which have "adapted" to the design space.

Specifically, design variables are represented as a string of binary numbers, which serves as a chromosome. The genetic algorithm randomly generates an initial population of individuals, and then decodes the chromosome of each individual into the appropriate design variables before running them through an analysis code. The results of the analysis code are then used to determine a value for the individual's fitness. Individuals with better fitness values are more "optimal".

Applications of the genetic algorithm to relevant, single objective, aerospace design problems include airfoil aerodynamics for fixed-wing applications,[2,3,4]

[*] Graduate Student, School of Aeronautics and Astronautics, Student Member AIAA
[†] Assistant Professor, School of Aeronautics and Astronautics, Member AIAA
[‡] Associate Professor, School of Aeronautics and Astronautics, Associate Fellow AIAA

Copyright © 1998 by B. R. Jones, W. A. Crossley, and A. S. Lyrintzis. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission

rotor applications,[5] and rotor system design for aeroacoustics[6]. GA applications to multi-objective and multidisciplinary design of airfoils have also been investigated.[7] Airfoil design is truly one such multi-objective, multidisciplinary application, although past work frequently focused solely on aerodynamics. In addition to the traditional objective of aerodynamic efficiency across a range of flight conditions, the modern airfoil designer must consider issues that include aeroacoustics, manufacturability, radar-cross section, and structural concerns.

This paper discusses the application of an efficient, parallel, genetic algorithm to such a multidisciplinary airfoil design problem. Specifically, airfoils were simultaneously optimized, under typical flight conditions seen by a helicopter rotor, to maximize lift-to-drag and to minimize both thickness noise and loading noise. The optimization is not only multidisciplinary, but multi-point as well, in that multiple flow conditions are considered when evaluating each individual (i.e. airfoil). Further, the GA identified, in a single run, a family of airfoils representing the Pareto-optimal surface. Consequently, designers can apply their knowledge and experience to the selection of appropriate airfoils from this resulting family, whose tradeoffs between aerodynamic and aeroacoustic performance have already been calculated.

This research addresses the direct design problem by tasking the genetic algorithm to manipulate the control points that define the airfoil's splined surface. An inverse airfoil design problem uses an optimization scheme to produce an airfoil whose pressure distribution matches the desired distribution. This entails the risk of defining a distribution that cannot be physically described. Solving the direct problem with the GA also allows the discovery of airfoil shapes not possible using other methods. The fact that the GA does not require an initial design point further enhances this global search capability.

Unfortunately, increased computational cost accompanies the use of a GA to solve the direct problem. Because the fitness of each individual in the GA's population is evaluated independently of the others, the GA is well suited to course grain parallelization. The code developed for this effort employs the Message Passing Interface (MPI) to parallelize the code according to the manager/worker model, garnering the computational benefits of a distributed computer system. The final result is a computationally resourceful genetic algorithm that makes this multi-objective, multidisciplinary design problem tractable.

IMPLEMENTATION

Genetic algorithms have been the subject of research and discussion for a number of years now, and, as such, a multitude of features have been examined and tested in the hopes of improving the basic approach. Those features specific to the GA implemented here include Gray coded design variables, *n*-branch tournament selection, and uniform crossover. Empirically derived relationships for the automatic determination of population size and mutation rate were also used.

The GA begins the evolutionary process by randomly generating sequences of ones and zeros to represent the chromosomes of each individual in the initial generation. Total chromosome length depends on the range and resolution desired for each decoded design variable. The determination of other initial values, especially population size, N_{pop}, and mutation rate, P_{mut}, is more difficult and often left up to the user's experience. The current implementation, however, utilizes empirical relations developed by Williams and Crossley[8] to calculate values for these parameters.

Genetic Operators

After creation of the initial population, evolution begins, following several possible methods. The first step in this process requires the selection of a pool of parents from the available individuals. While a variety of selection routines exist, all depend on fitness values returned by the relevant analysis application. Decoding an individual's binary chromosome into design variables begins this process. Gray codes, a variation of the traditional binary number coding where adjacent numbers differ in only one bit location, are employed to avoid possible numerical difficulties within the genetic algorithm.[1] The GA decodes these Gray codes, using bit operations for computational efficiency, into real and/or integer valued numbers representing a set of design variables. The analysis code then analyzes the given design and returns those values required to calculate the corresponding fitness value.

The genetic algorithm under study employs *n*-branch tournament selection with uniform crossover. A total of *n* branches exist, corresponding to the number of objectives considered. This method is an extension of the two-branch method developed by Crossley, *et. al*[9,10,11]. Thus, a user defined number of individuals are randomly selected from the "gene pool" and competed against each other, with a separate tournament held for each branch, to create a parent pool. Thus, each branch of the tournament competes the fitness values derived from one of the *n* objective functions. Figure 1 illustrates the tournament process.

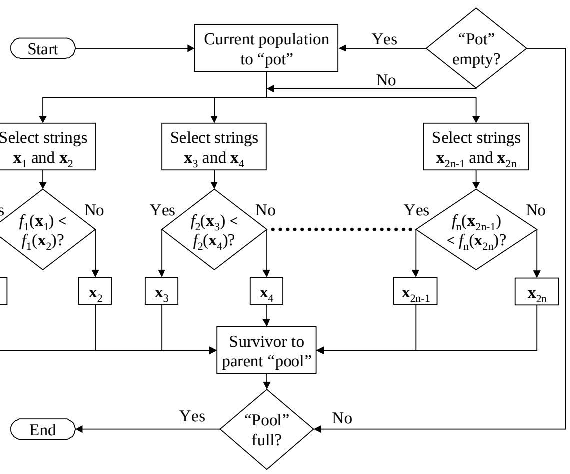


Figure 1: Flowchart of *n*-Branch Tournament Selection

As illustrated, the individual with best fitness "wins" and is simultaneously selected as a parent and removed from the "pot". The tournament competes every member of the population using one of the *n* fitness functions before moving on to the next branch. Restocking the "pot" as necessary, selection continues until the number of individuals in the parent pool matches the number in the current generation.

After selection, two members of the parent pool are randomly selected without replacement and "mated" in order to pass on their traits to two children. This operation is implemented using uniform crossover. Uniform crossover deviates slightly from a truly Darwinian process, but it has also been shown to improve the optimization capabilities of the GA[12]. For every bit position in the chromosome, a 50% chance exists that a bit from one chromosome will swap positions with the corresponding bit from the other. Thus, the "first" child receives the bit from the "second" parent and vice versa. After performing this operation along the entire length of the chromosome, two new children result. This process repeats until an entire new generation of individuals is created.

Augmenting the selection and crossover operators, mutation occurs at random intervals with a very low probability. The mutation operation randomly flips a bit in the selected chromosome to its opposite value (i.e. 1 to 0, or 0 to 1). This ensures that the evolution proceeds toward a truly global solution by introducing binary patterns that may not exist in the present population. Conceptually, the best characteristics of each individual are passed to each new generation, until the solution converges on the global optimum.

After processing each generation, the GA performs statistical operations on the population. These include standard, fitness-based measures of population uniformity and tallies of mutation and crossover rates. Additionally, for a single objective problem, the GA determines if the current generation contains the best individual to date. When optimizing a multiobjective problem, however, it examines every individual for possible inclusion in the Pareto optimal set.

Pareto Set Determination

As previously mentioned, the GA's use of a population of design points lends itself to multi-objective optimization. Instead of attempting to create a single objective function that accurately represents the importance of multiple design objectives, it is much easier to let the genetic algorithm handle each objective separately and simply display the tradeoffs between them when finished. The *n*-branch tournament selection considers each objective equally, leaving only the filtering process that determines whether an individual's fitness values places it on the Pareto surface.

For inclusion in the Pareto optimal set,[13] individuals must be both feasible and non-dominated. Thus, none of the design constraints are violated and no other design exists that is equal or better in each of the *n* fitness values and better in at least one of the fitness values. Expressed mathematically, a design with a vector of objective values, $\vec{U}$, is dominated by a design with a vector of objective values, $\vec{V}$:

$$\vec{V} \text{ dom } \vec{U} : \forall_i \, V_i \le U_i \text{ and } \exists_i \, V_i < U_i \qquad (1)$$

The GA evaluates thousands of designs during a run, creating the challenge of identifying the Pareto optimal set of designs for the given problem. Since the population moves across the design space as it evolves, any feasible non-dominated design evaluated during the run is of interest. Consequently, the GA stores all the Pareto optimal individuals in a linked list after applying the necessary filter logic. This data structure best handles the fluctuating length of this optimum population, as members are frequently inserted and removed. Having the GA re-evaluate the Pareto set following every generation is valuable, but not without cost. While locating the Pareto surface using only the final generation is certainly more computationally efficient, the current method ensures the inclusion of Pareto optimal individuals that might otherwise not survive until the final generation.

Evolution halts when either the maximum allowed number of generations is reached or the convergence criterion is satisfied. Convergence occurs when the population fails to change significantly over a user-specified number of generations. This occurs when the Pareto optimal set, for the multi-objective case, ceases to change.

Parallelization

While the GA has proven effective for global optimization, this capability is purchased at a high computational cost. The current GA may solve smaller problems in serial mode, but typical aerospace design problems incur unreasonable computational time when solved in such a manner. To make use of the independence of each individual's fitness evaluation, the parallel GA distributes the individuals among all available processors for analysis. Such a parallel model is frequently described as "manager/worker," and this work employs the Message Passing Interface (MPI) to implement it efficiently. The manager node performs all GA operations and distributes individuals to the worker nodes for analysis. The manager dynamically balances the work across all of the workers, whose sole purpose is then to evaluate fitnesses. The manager and workers communicate exclusively via blocking communication.

Blocking communication dictates that neither the sending nor receiving processor can begin another task until the communication is complete. While fairly simple and safe, this method is typically slower than asynchronous communication, where the sending processor starts the send and moves on to another task, trusting the receiving processor to pick it up. Using a dynamic load-balancing model, the manager processor distributes individuals only as worker processors become available. While more difficult to implement, this method avoids the potentially large bottleneck associated with static load balancing. Static load balancing evenly distributes the individuals to be evaluated among the total supply of workers. Thus, workers that have completed their evaluation of a simple design case sit idle, waiting for the worker tasked with a difficult design case to finish. Because a GA initiates design evolution from a random population, large differences may exist in the time needed to evaluate different designs. Dynamic load balancing alleviates this problem by allowing idle workers to begin new tasks without waiting for slower workers to finish.

The Message Passing Interface was specifically selected for this work to avoid limiting the GA to any particular parallelization package or hardware. Due to the rapidly increasing performance and availability of the average personal computer, distributed computing environments now include a variety of hardware platforms running a multitude of operating systems. MPI operates efficiently on these distributed systems, in addition to dedicated parallel computers. MPI is a specification that defines a fixed library of calls that allow different processors to pass information to each other. As a result, a large number of vendors have produced MPI libraries, with several existing in the public domain. MPI may not offer the fastest parallel implementation for a given system architecture, but the free acquisition cost and the ability to run on platforms ranging from supercomputers to non-homogenous networks has seen it become almost universally accepted and supported.

MULTIOBJECTIVE PROBLEM STATEMENT

For this effort, a multiobjective rotor airfoil design problem was posed to address aerodynamic and aeroacoustic concerns. Building upon the ideas described in Ref. 5, flight conditions for the airfoil analysis were taken from the example helicopter in forward flight presented in Prouty's text.[14] The aerodynamic objective was to minimize the airfoil's drag coefficient at flight conditions seen by the rotor blade as it passes through the 90°, 180° and 270° azimuth angles. Reasonable results are ensured by imposing constraints on the lift and moment coefficients at these three conditions. These constraints are derived from the desire to produce a design that at least equals the performance of the NACA 0012 section used in Prouty's example helicopter. A third constraint utilized the boundary layer solution error to ensure that reasonable flow conditions were achieved. The aerodynamic fitness function is thus expressed as:

$$\text{minimize } f_1 = \sum_{i=1}^{3}\left[c_d + \sum_{j=1}^{n_{con}} r_j P_j\right]_i \tag{2}$$

In this form, drag is minimized, subject to constraints imposed via an exterior penalty method, as a sum over all three (i = 1, 2, 3) design conditions.

The constraints for this problem are expressed in the following manner:

$$g_1 = 1 - \frac{c_\ell}{(c_\ell)_{Allowed}} \tag{3}$$

$$g_2 = \frac{|c_m|}{(c_m)_{Allowed}} - 1 \tag{4}$$

$$g_3 = \frac{Err_{BL}}{tol} - 1 \tag{5}$$

With these definitions, all constraints are enforced via linear penalty functions of the form:

$$P_j = \max\lfloor 0, g_j \rfloor \tag{6}$$

To complete the fitness function, f_1, values for r_j were chosen such that the magnitude of the penalty matched the order of the objective function. Table 1 presents the flight conditions and corresponding allowable constraint values. The inclusion of three azimuth angles, in addition to the two fitness functions, produces a multi-point, multiobjective rotor airfoil optimization.

Table 1: Flight Conditions.

	i = 1 (Ψ = 90°)	i = 2 (Ψ = 180°)	i = 3 (Ψ = 270°)
Re	8.67×10^6	6.20×10^6	3.73×10^6
M	0.61	0.44	0.26
α	1.50°	5.50°	9.00°
c_ℓ	≥ 0.16	≥ 0.60	≥ 1.00
$\lvert c_m \rvert$	< 0.03	< 0.03	< 0.03

To promote computational efficiency, the three flight conditions are defined by the Reynolds number, Mach number, and angle of attack seen by Prouty's example helicopter rotor in forward flight. Specifying angle of attack, rather than lift coefficient, pre-supposes a twist and inflow angle for the rotor. This approach, however, avoids the much more costly and numerically sensitive iteration scheme required by specifying C_l.

The aeroacoustic fitness function is also based upon the three flight conditions, and has the form:

$$\text{minimize } f_2 = P^* \sum_{i=1}^{3} [OASPL]_i \tag{7}$$

This formulation minimizes the sum of Overall Averaged Sound Pressure Level (OASPL) values produced by the airfoil over the multiple flight conditions. A scaled penalty factor, P^*, enforces the aeroacoustic constraints. This factor is scaled relative to the first objective and associated penalty values. Thus, constraint violations encountered by the acoustic fitness have the same scale as those seen by aerodynamic fitness.[11] Defined mathematically the scaled penalty factor becomes:

$$P^* = \frac{f_1}{\sum_{i=1}^{3} (c_d)_i} \tag{8}$$

This now completes the expression of all of the value needed by the GA to compute the required fitness functions, but leaves open the means employed to obtain these values.

Airfoil Analysis Methods

As the quality of the GA's results relates directly to the validity of the aerodynamic and aeroacoustic coefficients and parameters, care was taken to employ well-established computational analysis tools. The airfoil analysis and design code, XFOIL[15], forms the core of the analysis suite utilized by the GA. XFOIL combines the Karmen-Tsien compressibility correction with a solution generated from closely coupled viscous and inviscid methods. Specifically, XFOIL employs a two equation, lagged dissipation, integral boundary layer solution strongly interacted with a linear-vorticity, stream function panel method via the surface transpiration model.

While reduced by eliminating the interactive design and plotting features, as well as modified to create a callable function, the analytic capabilities of XFOIL remain unchanged. As a result, lift, drag and moment coefficients can be obtained for airfoils operating through transonic flight conditions in relatively brief periods of time. Improvements in accuracy could be obtained by solving higher-order equations (e.g. Euler or Navier-Stokes). Unfortunately, these higher-order methods entail a significant increase in computational cost. When considering the thousands of airfoil designs evaluated during a single run of the GA, this cost currently renders the higher-order analysis undesirable, at least for the preliminary stages of this work.

While XFOIL forms the core of the analysis application, several other components round out the package. The design variables for this problem comprise the ordinates of B-spline control points, excluding the fixed leading and trailing edge points; this provides an airfoil description suitable for XFOIL. All control point chordwise coordinates remain fixed and a blunt trailing edge is defined. Using a heavy knot weighting, the B-spline algorithm does not require the resulting surface to pass through the control points. This is more forgiving of the random, irregular "airfoils" that are encountered in early generations. Figure 2 illustrates such an airfoil while illustrating the usage of B-splines.

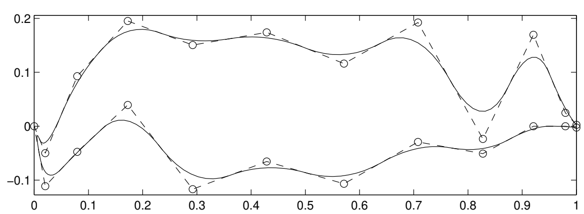


Figure 2: Generation Zero Airfoil Illustrating B-splined Surface

Obviously, this shape fails to represent a practical airfoil. Flow separation would occur at a variety of locations and any structural engineer would go to great lengths to avoid constructing such a form. Indeed, the fact that initial design cases are randomly generated could easily produce an airfoil with crossed surfaces. While flow separation is penalized in the form of increased drag and reduced lift, structural integrity requires further involvement. Substitutions of new, physically feasible shapes replace impossibly shaped airfoils prior to evaluating each individual.

With the description of physically valid airfoils, XFOIL directly returns the variables needed for aerodynamic optimization, c_l, c_d, and c_m. The airfoil's loading and thickness noise, however, requires a separate calculation. These computations are accomplished using the Ffowcs-Williams and Hawkings equations.[16,17] Both thickness and loading noise depend primarily upon the pressure and thickness distribution over the airfoil. XFOIL also provides this information and passes it to the aeroacoustic analysis code, WOPWOP.[18]

Because the problem of interest addresses the design of two-dimensional airfoil sections and WOPWOP natively handles a three-dimensional rotor system, some modeling simplifications were made. To begin, the "rotor" defined for WOPWOP is assumed to have a single blade. The blade extends, in the radial direction, from the 75% radius station to the tip, incorporating the rotor dimensions of Prouty's example helicopter. Tip effects are not computed, as the design of the two-dimensional airfoil does not address this three-dimensional effect. The rotor is placed in a "hover-like" condition by assuming that the loading given by XFOIL for the airfoil is constant in both radial station and in azimuthal position. Using the airfoil angle of attack from the design condition, and assuming a hover-like inflow, allows the computation of the associated rotor pitch angle. Repeating this for all three design conditions provides an individual, simplified rotor model for each of the three specified cases as well as a loading condition suitable for analysis. This allowed the computation of a noise measure associated with the 2D airfoil shape as well as computational savings.

This analysis then provides a measure of the loading and thickness noise as incorporated into the overall averaged sound pressure level. This value is calculated as experienced by an observer stationed in the rotor plane and 50 meters in front of the rotor hub. Summing this measure over the three design conditions supplies the required aeroacoustic fitness value for the genetic algorithm.

While it is understood that a one-bladed rotor as modeled would never be constructed, this approach was adopted to provide a noise measure for the two-dimensional airfoil shape, rather than an accurate prediction of actual noise levels. Here, the implementation assumes that if the simple model associated with one airfoil generates less noise than a model associated with another airfoil, that the first airfoil is a lower-noise design.

RESULTS

Multiobjective Test Case

The GA's ability to generate a Pareto optimal set was first tested using a simple mathematical multiobjective function. Figure 3 plots the two equations as functions of position, displaying the actual fitness functions in the legend, while Figure 4 displays the Pareto front in the corresponding function space.

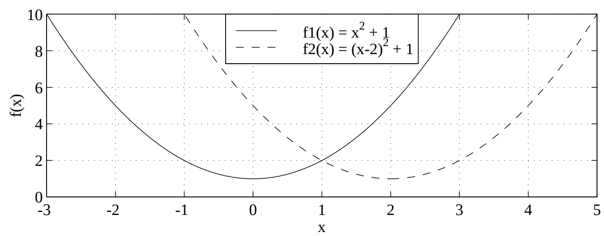


Figure 3: Mathematical Test Functions

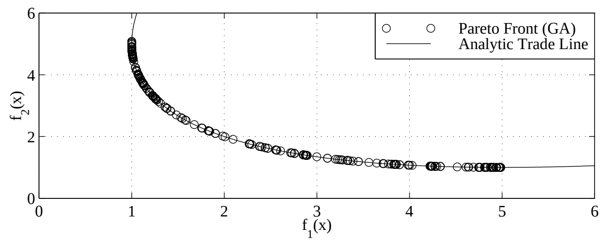
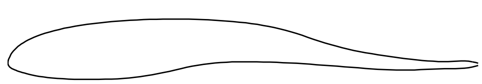


Figure 4: Successful location of the Pareto Front in Function Space

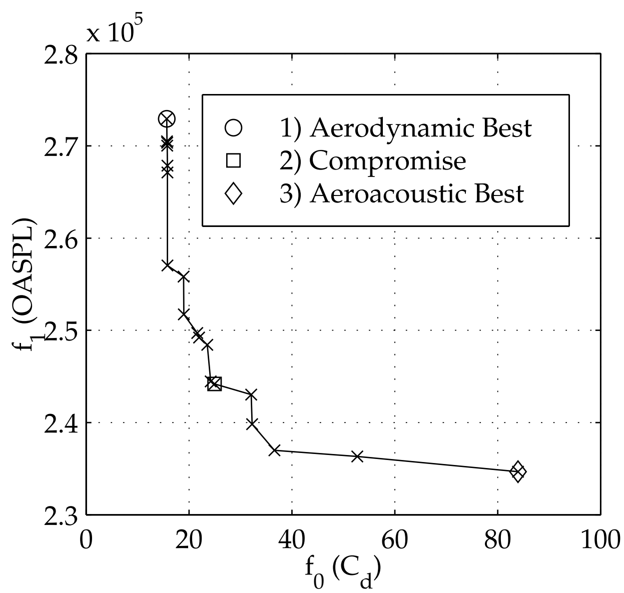
The genetic algorithm succeeds, defining the Pareto surface in a single run. Individual members of the Pareto optimal set, represented by the small circles in Figure 4, perfectly overlay the analytic trade line. Since the GA deals with a population of points, this proves relatively efficient when compared to the traditional method of performing successive runs of the optimization code while progressively altering weighting values applied to the objective functions[19]. All individuals in the Pareto optimal set are both feasible and non-dominated, existing between the expected range of $0 \le x \le 2$.

With the multi-objective performance of the GA now demonstrated, the Pareto optimal set of airfoils describing the set of compromises between aerodynamic and aeroacoustic performance is described. For the results in this paper, the design variable ranges and derived GA parameters are listed in Table 2.

α	C_l	C_m	C_d	OSPL
1.50	0.5498	−0.0254	0.0069	89.74
5.50	0.9867	−0.0188	0.0099	90.95
9.00	1.2944	−0.0194	0.0140	85.89



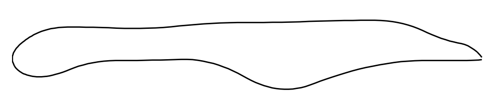
1) Aerodynamic Best



α	C_l	C_m	C_d	OSPL
1.50	0.3953	0.0009	0.0081	83.39
5.50	0.9131	−0.0082	0.0090	88.91
9.00	1.2580	−0.0135	0.0111	84.62

2) Compromise

α	C_l	C_m	C_d	OSPL
1.50	0.2888	−0.0029	0.0334	79.70
5.50	0.8045	−0.0280	0.0199	89.13
9.00	1.1906	−0.0431	0.0165	86.20



3) Aeroacoustic Best

Figure 5: Pareto Front and Representative Airfoils

Table 2: GA Variable Values

Variable	Value
y/c_u	-0.02 < y/c < 0.13
y/c_l	-0.13 < y/c < 0.02
Resolution	0.005
Chromosome length	100 bits
N_{pop}	400
P_{mut}	0.001262

The GA output discussed here resulted from a run across 64 nodes of the San Diego Supercomputer Center IBM SP2 for a total of 3.5 hours, completing 247 generations and creating a final Pareto front consisting of 19 individuals. The Pareto front, and three representative airfoils with performance summaries, are shown in Figure 5. The Pareto front itself displays the counts of drag against the overall averaged sound pressure level (dB re μPa). Since the plot compares fitnesses, the results account for any penalties and are summed across the three different design conditions. The representative airfoils display a broad cross-section of the Pareto front by including the airfoil with the best measured aerodynamic performance, the best measured aeroacoustic performance and a compromise airfoil selected from the midpoint of the front.

Admittedly, the generated airfoils are atypical. Because the GA approach to shape design requires no initial starting point it is possible to "discover" non-traditional airfoil shapes. These atypical designs were not surprising.

The problem formulation and analysis methods also contribute to the generation of unusual shapes. The GA, like most optimization techniques, will often exploit weaknesses in the analysis methods as well as shortcomings in the problem formulation. The GA generated airfoils are valid and feasible given the analysis tools used, but they should not be viewed as final airfoil designs. Using the GA approach has provided interesting shapes that should be used as the starting point for further refinement.

There are certainly further influential aspects of the design problem to consider. For example, some regions of the flight envelope begin to suffer from transonic effects, yet wave drag remains unaccounted for with the current analysis. Additionally, preliminary runs of the GA illustrated how easily the problem statement can affect the results. In an initial attempt to ensure low drag airfoils, a constraint was imposed to force flow separation to occur no further forward than 95% of the chord. This constraint severely limited the design space, preventing the generation of reasonable airfoils. Because XFOIL models separation well, including the resulting effects on lift and drag, the additional constraint proved not only redundant, but detrimental.

As further validation of the airfoils shown in Figure 5, the pressure coefficient distributions of the midpoint, or compromise, airfoil are displayed in Figure 6.

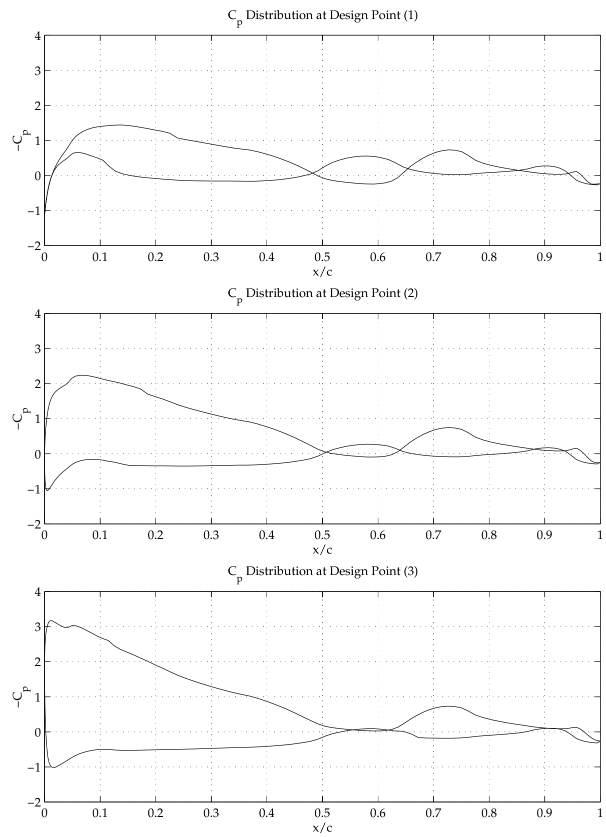
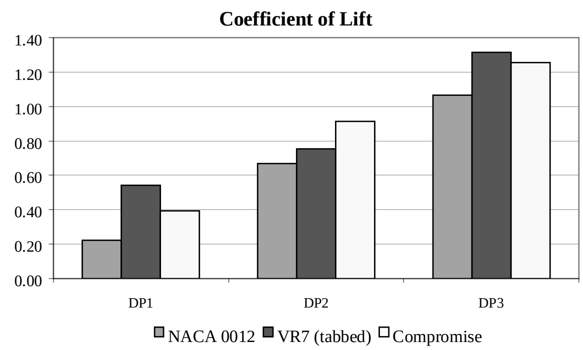
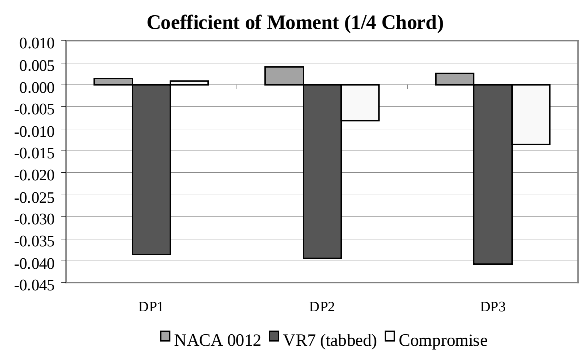


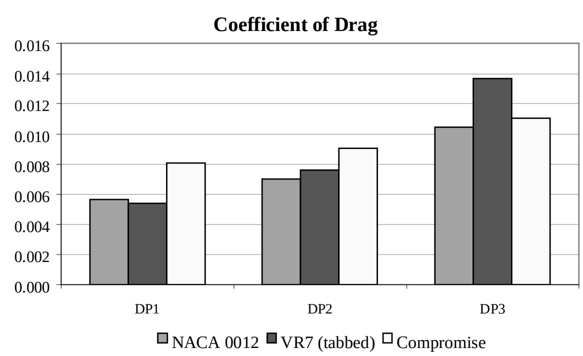
Figure 6: Compromise Airfoil Pressure Profiles

As indicated, these profiles behave as expected in the light of basic aerodynamics. Drops in the pressure profile correspond to dips in the airfoil surface. While not shown, the flow does indeed separate from the lower surface protuberance of airfoil (3). A noticeable pressure recovery exists along the upper surface of airfoil (1). Additionally, many of the features evident on the example airfoils are quite reasonable when examined independently. For example, most of the airfoils have a noticeable camber toward the leading edge. Increased camber benefits c_l while positioning the camber toward the leading edge reduces potential moment problems. Larger radius leading edges help prevent flow separation at higher angles of attack while thinner airfoils typically have less profile drag and reduced thickness noise. Cusped trailing edges, while difficult to manufacture, provide obvious drag benefits. These features exist, in multiple combinations, even in the limited sampling of airfoils shown in Figure 5.

To provide a comparison for the GA generated airfoils, both the NACA 0012, of Prouty's example helicopter, and the more modern Boeing Vertol VR7 (tabbed) airfoil were analyzed under identical flight conditions using the actual analysis codes employed within the GA. Figure 7 displays both the related aerodynamic coefficients and overall averaged sound pressure level (dB re 20 μPa) for each design condition.







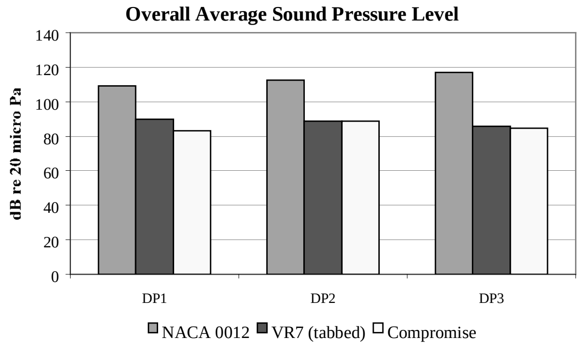


Figure 7: Representative Airfoil Comparison

As can be seen, the GA airfoil representing the compromise between aerodynamics and aeroacoustics compares rather well in light of the specified constraints. Before making detailed comparisons, it should be noted that XFOIL was unable to converge on a boundary layer solution for the 2nd design point. Additionally, the VR7 (tabbed) fails to converge at the 1st design point. This may suggest drag predictions of less than desirable accuracy. Observation taken during the course of several runs indicates, however, that the 2nd design point is by far the most difficult of the three.

While the NACA 0012 performs well in pitching moment, it shows its age in all other categories. The VR7, however, performs well in all categoies with the exception of moment. The GA compromise airfoil is slightly quieter than the VR7, but pays for it with aerodynamic performance. Herein lies the advantage of the multi-objective GA, however, in that the Pareto front allows a designer to immediately re-consider the tradeoffs between the different objectives and select an airfoil more suitable to the stated needs. For example, the GA's best aerodynamic airfoil provides better lift values, better moment values and similar drag values. Considering that the GA had no incentive to improve c_l beyond the given c_l constraints the GA compromise airfoil compares favorably.

As stated before, results depend heavily on the problem formulation. It is dangerous for designers to inject their knowledge into the problem formulation without careful consideration. While perhaps entirely valid, such knowledge may limit the initial design space and thus eliminate many valuable design possibilities. No more pertinent example of this exists than the authors' attempts to constrain the boundary layer separation location. Separation is known to cause drag problems and an adaptive penalty based on separation appeared a reasonable means of reducing boundary layer convergence problems. In actuality, applying such a constraint severely limited the initial design space and produced a very narrow Pareto front. As XFOIL is quite capable of handling separation, the natural penalties of separation on lift and drag proved adequate. At the same time, however, care must be taken to consider only valid results when constructing fitness values. The pressure distribution WOPWOP employs to calculate loading noise means nothing if the XFOIL boundary layer solution didn't converge first.

Parallel Evaluation

The additional capability granted to the genetic algorithm through parallel operation cannot be underestimated. GA's are often called "embarrassingly parallel" and thus offer a means to compensate for the computational expense. While the account expense of running on the SDSC SP2 pre-empted a formal scalability study, different parallel runs were conducted using 8, 16, 32, and 64 nodes. The observed performance increase corresponded very closely to the 1/(n-1) expected from a manager/worker arrangement.

CONCLUSIONS

This research applied an advanced *n*-branch genetic algorithm to a multidisciplinary, multiobjective design problem to obtain a set of helicopter rotor airfoils representing the family of compromises between aerodynamic efficiency and minimum noise. Given the level of analysis tools employed, the resulting airfoils appear to be promising, non-traditional shapes for improved aerodynamic and aeroacoustic performance. These designs may provide good starting points for further development.

Using MPI to implement a manager/worker parallel model made this computationally expensive approach tractable. The naturally parallel structure of the genetic algorithm has made this fairly straightforward.

Employing the n-branch tournament selection for this work also allowed a range of designs to be generated in a single run of the GA. This feature allows designers to choose an airfoil best suited to their needs.

ACKNOWLEDEMENTS

The calculations for this work were performed on the 128 node IBM SP2 at SDSC (San Diego Supercomputing Center) which is part of the National Partnership for Advanced Computational Infrastructure (NPACI).

REFERENCES

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

[2] Doorly, D. J., and Peiro, J., "Supervised Parallel Genetic Algorithms in Aerodynamic Optimization," AIAA Paper 97-1852, *Proceedings of the 13th AIAA CFD Conference*, Snowmass, CO, June 30-July 2, 1997, pp. 210-216.

[3] Obayashi, S. and Takanashi, S., "Genetic Optimization of Target Pressure Distributions for Inverse Design Methods," *AIAA Journal*, Vol. 34, No. 5, 1996, pp. 881-6.

[4] Quagliarella, D. and Cioppa, A. D., "Genetic Algorithms Applied to the Aerodynamic Design of Transonic Airfoils," *Journal of Aircraft*, Vol. 32, No. 4, Jan. 1995.

[5] Fanjoy, D. W., and Crossley, W. A., "Aerodynamic Shape Design for Rotor Airfoils via Genetic Algorithm," *Journal of the American Helicopter Society*, Vol. 43, No. 3, Jul. 1998, pp. 263-270.

[6] Wells, V. L., Han, A. Y., and Crossley, W. A., "Acoustic Design of Rotor Blades Using a Genetic Algorithm," *Proceedings of the AGARD Symposium on Aerodynamics and Aeroacoustics of Rotorcraft*, Berlin, Germany, August, 1995, pp. 35-1 – 35-10.

[7] Doorly, D. J., Peiro, J. and Oesterle, J-P., "Optimization of Aerodynamic and Coupled Aerodynamic-Structural Design Using Parallel Genetic Algorithms," *Proceedings of the 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Vol. 1, 1996, pp. 401-9.

[8] Williams, E. A. and Crossley, W. A., "Empirically-Derived Population Size and Mutation Rate Guidelines for a Genetic Algorithm with Uniform Crossover," WSC2: 2nd Online World Conference on Soft Computing in Engineering Design and Manufacture, June 23-27, 1997.

[9] Crossley, W. A., "Genetic Algorithm Approaches for Multiobjective Design of Rotor Systems," AIAA Paper 96-4025, *Proceedings of the 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, WA, Sep. 4-6, 1996, pp. 384-394.

[10] Norris, S. R., and Crossley, W. A., "Pareto-Optimal Controller Gains Generated by a Genetic Algorithm," AIAA Paper 98-1010, presented at AIAA 36th

Aerospace Sciences Meeting and Exhibit, Reno, NV, Jan. 12-15, 1998.

[11] Crossley, W. A., Cook, A. M., Fanjoy, D. W., and Venkayya, V. B., "Using the Two-Branch Tournament Genetic Algorithm for Multiobjective Design," AIAA Paper 98-1914, to be presented at the 39th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Long Beach, CA, Apr. 20-23, 1998.

[12] Syswerda, G., 1989, Uniform crossover in genetic algorithms, *Proceedings of the third international conference on genetic algorithms*, George Mason University, Fairfax, VA, pp. 2-9.

[13] Vincent, T. L., and Grantham, W. J., *Optimality in Parametric Systems*, John Wiley & Sons, NY, 1981.

[14] Prouty, R. W., *Helicopter Performance, Stability and Control*, Krieger Publishing Company, Malabar, FL, 1990.

[15] Drela, M., “XFOIL: An Analysis and Design System for Low Reynolds Number Airfoils,” Conference on Low Reynolds Number Airfoil Aerodynamics, University of Notre Dame, June 1989.

[16] Brentner, K. S. and Farassat, F., “Helicopter Noise Prediction: The Current Status and Future Direction,” *Journal of Sound and Vibration*, Vol. 170(1), pp. 79-96, 1994.

[17] Schmitz, F. H., “Rotor Noise,” Chapter 2, pp. 65-149 in *Aeroacoustics of Flight Vehicles*, H. H. Hubbard, ed., Acoustical Society of America, Woodbury, NY, 1995.

[18] Brentner, K. S., “Prediction of Helicopter Rotor Discrete Frequency Noise,” NASA Technical Memorandum 87721, October 1986.

[19] Vanderplaats, G. N., *Numerical Optimization Techniques for Engineering Design: With Applications*, McGraw-Hill, 1984.