AIAA JOURNAL
Vol. 42, No. 2, February 2004

Swarm Algorithm for Single- and Multiobjective Airfoil Design Optimization

T. Ray* and H. M. Tsai†
National University of Singapore, Singapore 119260, Republic of Singapore

Shape optimization of airfoils involves highly expensive, nonlinear objective(s) and constraint functions often with functional and slope discontinuity that limits the efficient use of gradient-based methods for its solution. Gradient-based methods are not capable of generating a set of pareto solutions as required in multiobjective problems as they work with a single solution and improve it through successive iterations. Population-based, zero-order, stochastic optimization methods are therefore an attractive choice for shape optimization problems as they are easy to implement and effective for highly nonlinear problems. We present a swarm algorithm that is applicable for optimization problems in general, but is here explored for airfoil design optimization studies. The algorithm is based on a sociobehavioral model, and it offers the designer the desired flexibility to solve various unconstrained/constrained, single-/multiobjective forms of the airfoil shape optimization problem. The algorithm handles objectives and constraints separately via pareto ranking and is thus immune to problems of scaling and aggregation that commonly affect penalty-function-based constraint handling schemes. Three different airfoil design optimization problems have been solved to illustrate the algorithm's flexibility and its computational efficiency, which compare favorably with existing stochastic search methods.

Introduction

DIFFERENT airfoil design optimization approaches have evolved over the years. Airfoil design problems are characterized by highly nonlinear and computationally expensive objective functions and a variety of constraints. A number of airfoil design optimization studies have been reported in the literature that use different shape representation schemes (e.g., Hicks–Henne[1,2] bump functions, PARSEC representation,[3,4] Beziers,[5] B-splines,[6,7] and NURBS[8]) coupled with a variety of optimization methods (genetic and evolutionary algorithms, simulated annealing, gradient-based methods, and adjoint formulations based on control theory).

Unconstrained single-objective airfoil design optimization problems have been solved using evolutionary algorithms,[6] genetic algorithm (GA) with real number encoding,[3] and hybrids comprised of GA and gradient-based methods.[9] Constrained single-objective airfoil design problems have also been solved using GA[10,11] and multiobjective formulations of the airfoil design optimization problem have been solved using nondominated sorting genetic algorithm (NSGA),[5] multiobjective GA,[7] and NSGA coupled with artificial neural networks. In all of the preceding studies, specific forms of the airfoil design optimization problem were solved using different algorithms, and the methodologies to handle constraints were either through penalty functions or left out of discussion. Adjoint formulation[12,13] is another more recent method for shape optimization that aims to derive a boundary shape based on control theory. The key benefit lies in its efficiency to converge to optimal solutions using minimal function evaluations. When function evaluations are prohibitively expensive, this is a powerful approach. However, it lacks flexibility as an inclusion of an objective, or a constraint requires significant amount of formulation changes that is not easy for a designer to implement.

Genetic and evolutionary algorithms are popular among designers as they are easy to use and effective for highly nonlinear problems with functional and slope discontinuity. For such problems, there is virtually no room for conventional mathematical programming methods. Although much has been discussed about the benefits of genetic and evolutionary methods, there are a number of concerns that need to be addressed. Both genetic and evolutionary search methods require a number of user inputs that include crossover and mutation probabilities, type of crossover, population, and generation sizes to mention a few. As for constrained optimization problems, handling constraint via penalty function requires scaling factors and weights prior to aggregation. The performances of these algorithms are also dependent on the choice of the preceding inputs, which are not easy for a user to provide.

From an optimization point of view, there are significant differences in the solution requirements to a single-objective and a multiple-objective problem. In a single-objective problem, the aim is to find the best solution that maximizes or minimizes the objective, whereas in a multiobjective optimization problem (in absence of interobjective preference information) the goal is to arrive at a set of pareto optimal designs. In the context of multiobjective design, Messac[14,15] and Messac and Ismail-Yahaya[16] highlighted the drawbacks of using arbitrary weights for objective aggregation and proposed a method referred to as physical programming to deal with such problems. The method is flexible and provides the designer a scope to formulate the problem in a more natural form using different levels of desirability for the performance measures. A comprehensive discussion on various evolutionary algorithms for multiobjective problems appears in the book by Kalyanmoy.[17]

The presence of constraints is known to largely affect the performance of all optimization methods. Handling constraints is a nontrivial problem and more so in presence of nonlinear equalities and inequalities. There have been a number of approaches to handle constraints that include rejection of infeasible individuals, use of penalty functions and their variants, repair methods, use of decoders, separate treatment of constraints and objectives, and hybrid methods incorporating knowledge of constraint satisfaction. Penalty function is perhaps the most popular constraint handling scheme that evolved though the stages of static penalties and dynamic penalties to more recent adaptive penalties to better deal with inherent scaling and aggregation problems. Repair methods are based on additional

Received 23 September 2002; revision received 4 August 2003; accepted for publication 25 August 2003. Copyright © 2003 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0001-1452/04 $10.00 in correspondence with the CCC.

*Senior Research Scientist, Temasek Laboratories, 10 Kent Ridge Crescent; tsltray@nus.edu.sg.

†Principal Research Scientist, Temasek Laboratories, 10 Kent Ridge Crescent; tsltray@nus.edu.sg. Member AIAA.

function evaluations, whereas the decoders and special operators or constraint satisfaction methods are problem specific and cannot be used to model a generic constraint. A comprehensive review on various constraint handling schemes is presented by Michalewicz,[18] and an effective scheme to deal with equality constraints in the context of robust design has been proposed by Mattson and Messac.[19]

Although stochastic methods are an attractive choice for airfoil shape optimization problems, they are computationally expensive as the search is guided by zero-order information. Thus, for problems where the objective function is expensive to evaluate care should be taken before applying such algorithms, although the use of surrogate performance functions during the course of search could alleviate this problem.[4]

To solve various forms of the airfoil shape optimization problem effectively, the optimization algorithm should posses the following features:

1) The algorithm should be capable of handling single- and multiobjective forms of the airfoil design optimization problem.

2) It should be able to handle constraints without additional user inputs.

3) It should make use of minimal number of function evaluations (flow computations) because such computations are expensive.

4) The algorithm should be amenable to parallelization.

The purpose of this paper is to present a swarm algorithm that is effective and efficient in solving various forms of single- and multiobjective, unconstrained and constrained airfoil design optimization problems without additional user inputs. The number of function evaluations required by the swarm is comparable with existing genetic and evolutionary methods, and the algorithm is amenable to parallelization. These features make the swarm algorithm an attractive choice for airfoil design optimization problems. The details of the swarm algorithm are presented in the next section followed by a discussion on the PARSEC shape representation scheme and the flow solver used in this study.

Swarm Algorithm

Swarm algorithms are fairly recent additions to the existing list of optimization methods that are capable of handling single- and multiobjective, unconstrained and constrained optimization problems.[20,21] Unlike conventional evolutionary methods, the swarm strategy is based on simulation of social behavior where each individual in a swarm adjusts its flying according to its own flying experience and companions' flying experience. The key to the success of such a strategy in solving an optimization problem lies with the mechanism of effective information sharing across individuals.

A modified swarm algorithm[22] was proposed recently that mimics the movement of an individual from a point to another rather than updating an individual's velocity and acceleration along the trails. This allows a reduction in the number of user-defined inputs. The modified swarm algorithm has been used to solve single-objective constrained design optimization problems[23] and multiobjective design optimization problems.[24] The mathematical details of the swarm algorithm are presented in the light of a generic constrained multiobjective optimization problem:

Minimize

$$\boldsymbol{f} = [f_1(\boldsymbol{x}) \quad f_2(\boldsymbol{x}) \quad \ldots \quad f_k(\boldsymbol{x})] \tag{1}$$

Subject to

$$g_i(\boldsymbol{x}) \geq a_i, \qquad i = 1, 2, \ldots, q \tag{2}$$

$$h_j(\boldsymbol{x}) = b_j, \qquad j = 1, 2, \ldots, r \tag{3}$$

where $x = [x_1 \ x_2 \ \ldots \ x_n]$ is the vector of n design variables and there are q inequality and r equality constraints. It is a common practice to transform the equality constraints (with a tolerance δ) to a set of inequalities and use a unified formulation for all constraints, that is, $h_j(\boldsymbol{x}) \leq b_j + \delta$ and $h_j(\boldsymbol{x}) \geq b_j - \delta$. Thus r equality constraints will give rise to $2r$ inequalities, and the total number of inequalities for the problem is denoted by s, where $s = q + 2r$.

For each individual, $\boldsymbol{c}$ denotes the constraint satisfaction vector $\boldsymbol{c} = [c_1 \ c_2 \ \ldots c_s]$, where $c_i > 0$ indicates the violation of the constraint. The constraint matrix for a swarm of M individuals assumes the form

$$\text{Constraint} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1s} \\ c_{21} & c_{22} & \cdots & c_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ c_{M1} & c_{M2} & & c_{Ms} \end{bmatrix} \tag{4}$$

The objective matrix assumes the form

$$\text{Objective} = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1k} \\ f_{21} & f_{22} & \cdots & f_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ f_{M1} & f_{M2} & & f_{Mk} \end{bmatrix} \tag{5}$$

It is important to define the concept of pareto and nondominated solutions before proceeding any further. A vector of decision variables $\boldsymbol{x}^* \in \boldsymbol{F}$ is pareto optimal if there does not exist another $\boldsymbol{x} \in \boldsymbol{F}$, such that $\boldsymbol{f}_i(\boldsymbol{x}) \leq \boldsymbol{f}_i(\boldsymbol{x}^*)$ for all $i = 1, \ldots, k$ and $\boldsymbol{f}_j(\boldsymbol{x}) < \boldsymbol{f}_j(\boldsymbol{x}^*)$ for at least one j. Here $\boldsymbol{F}$ denotes the feasible region of the problem (i.e., where the constraints are satisfied). If the design space is limited to M solutions instead of the entire $\boldsymbol{F}$, such solutions are termed as nondominated solutions.

In a swarm of M individuals, all nondominated individuals are assigned a rank of 1. The rank 1 individuals are removed from the swarm, and the new set of nondominated individuals is assigned a rank of 2. The process is continued until every individual in the swarm is assigned a rank. This is the underlying concept of nondominated sorting. Rank = 1 in the objective or the constraint matrix indicates that the individual is nondominated. It can be observed from the constraint matrix that when all of the individuals in the swarm are infeasible the rank = 1 solutions are the best in terms of minimal constraint violation. Whenever there is one or more feasible individuals in the swarm, the feasible solutions assume the rank of 1. The initial swarm consists of a collection of random individuals. Over time, the individuals communicate with the better performers and derive information from them.

For a constrained problem, individuals with a constraint rank = 1 (nondominated based on constraint matrix) are the best performers based on constraint satisfaction. When there are no feasible individuals in the swarm, the set of leaders (SOL) consists of all nondominated individuals based on the constraint matrix. When there is one or more feasible individuals, the SOL will consist of all feasible individuals as they will have a constraint rank = 1. The size of SOL is expected to grow as more and more individuals become feasible. To maintain a selective pressure in such a feasible SOL, individuals are allowed to be members of SOL only if they are better than average performers based on objective ranks. This process ensures a pressure maintained among the feasible SOL members to improve their objective performance to remain as SOL members. The pseudocode of the algorithm is presented next.

Pseudocode

```
Initialize $M$ individuals in the Swarm;
*Unconstrained Problem Strategy*
Do {
  Compute Objective Values of each Individual in the Swarm;
  Compute Ranks based on the Objective Matrix;
  Compute the Average Rank based on the Objective Matrix;
  Count the Number of Rank 1 Individuals;
  If (Number of Rank 1 Individuals $\leq M/2$)
    Assign Individuals to SOL if their Rank $\leq$ Average Rank;
  If (Number of Rank 1 Individuals $> M/2$)
    Assign Rank 1 Individuals to SOL;
  For (Each Individual not in SOL) {
    Select a Leader from SOL to derive information;
    Acquire information from the Leader and move to a new point in the search space;
  }
} while (termination condition = False)
*Constrained Problem Strategy*
Do {
  Compute Objective values for each Individual in the Swarm;
  Compute Constraint values for each Individual in the Swarm;
  Use Non Dominated Sorting to Rank Individuals based on Objective Matrix;
  Use Non Dominated Sorting to Rank Individuals based on Constraint Matrix;
  Compute the Average Rank based on the Objective Matrix;
  Count the Number of Rank 1 Individuals based on the Objective Matrix;
  Compute the Average Rank based on the Constraint Matrix;
  Count the Number of Rank 1 Individuals based on the Constraint Matrix;
  Count the Number of Feasible Individuals;
  If Number of Individuals with Objective Rank = $1 > M/2$: Average Objective Rank = 1;
  If Number of Individuals with Constraint Rank = $1 > M/2$: Average Constraint Rank = 1;
  If Number of Feasible Individuals = 0{
    Assign Individuals to SOL with Constraint Rank $\leq$ Average Constraint Rank;
    Shrink the above SOL, to contain Individuals with Objective Rank $\leq$ Average Objective Rank;
  }
  If Number of Feasible Individuals > 0{
    Assign Feasible Individuals to SOL;
    Shrink the above SOL, to contain Individuals with Objective Rank $\leq$ Average Objective Rank;
  }
  For (Each Individual not in SOL) {
    Select a Leader from the SOL to derive information;
    Acquire information from the Leader and move to a new point in the search space;
  }
} while (termination condition = False)
```

Selection of a Leader from SOL

For a multiobjective problem, a leader that has the minimum number of neighbors in the parametric space is selected from SOL (to ensure spread along the nondominated front), whereas for a single-objective problem a leader is the one that is closest to the individual in the parametric space.

Acquiring Information

A simple generational operator is used in this study to acquire information from a leader. The operator can result in a variable value even if it does not exist in either the individual or its leader, which is useful to avoid premature convergence. The probability of a variable value generated between an individual and its leader is 50%. The probability of a variable value generated between the lower bound of the variable and the minimum among the individual and its leader or between the upper bound of the variable and the maximum among the individual and its leader is 25% each. In addition to the preceding process, a solution is regenerated if it is nonunique in the variable space. The given generational operator has been used by Ray and Saini.[22] Though the present examples have been solved using the preceding operator, other generational operators like the simulated binary crossover or blend crossover can well be used instead. The flowchart of the swarm algorithm is presented in Fig. 1. Details of the solution decoding and optimization problem formulation are presented in Fig. 2.

Shape Representation and the Flow Solver

A variety of airfoil shape representation schemes, which use a different number of variables and different mathematical functions linking the variables have been reported in the literature. An ideal shape representation scheme for an airfoil design optimization study should possess the following features:

1) It should have the flexibility to explore the entire search space including nonconventional airfoil shapes.

2) It should require minimum number of variables for shape definition as more number of variables means more variables for the optimization problem.

3) The shape representation scheme should avoid generation of sharp discontinuities in slope and curvature.

4) The variables should be related to the aerodynamic performance that is being optimized.

Once a shape representation scheme is adopted, the number of variables for the optimization problem is fixed. To use these variables effectively in any optimization algorithm, the upper and the lower bounds of the variables need to be assigned. An inverse shape fitting can be used to get an initial estimate of the variables, and an

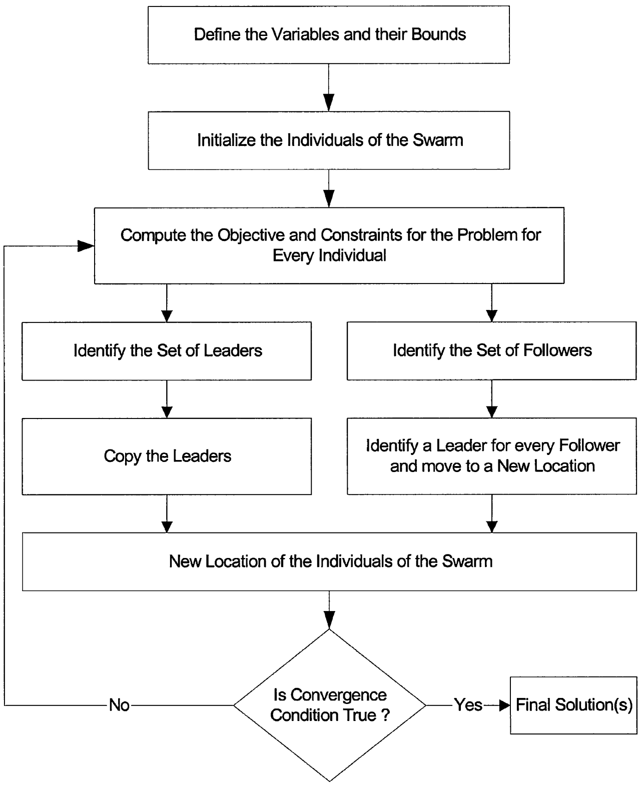
Define the Variables and their Bounds
Initialize the Individuals of the Swarm
Compute the Objective and Constraints for the Problem for Every Individual
Identify the Set of Leaders
Identify the Set of Followers
Copy the Leaders
Identify a Leader for every Follower and move to a New Location
New Location of the Individuals of the Swarm
Is Convergence Condition True ?
No
Yes
Final Solution(s)

Fig. 1 Flowchart of the swarm algorithm.

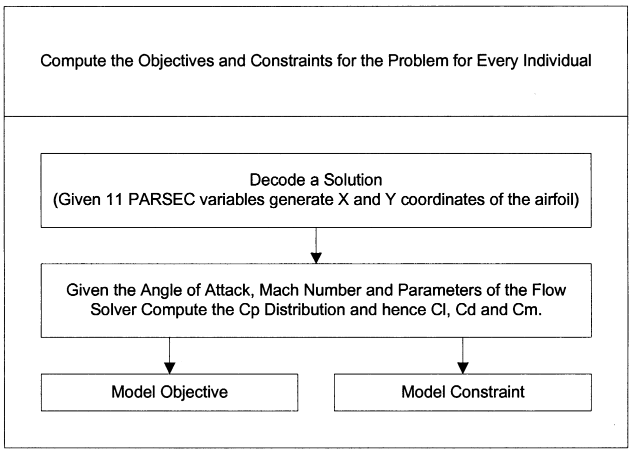
Compute the Objectives and Constraints for the Problem for Every Individual
Decode a Solution (Given 11 PARSEC variables generate X and Y coordinates of the airfoil)
Given the Angle of Attack, Mach Number and Parameters of the Flow Solver Compute the Cp Distribution and hence Cl, Cd and Cm.
Model Objective
Model Constraint

Fig. 2 Details of modeling a problem formulation.

appropriate percentage can be used to decide the variable bounds. A procedure for NURBS fitting is outlined in Ref. 25, which provides the initial estimates of the control points and their bounds for airfoil shape optimization problems.

In this study we used a PARSEC[3] representation scheme to define airfoil geometries because it involves 11 variables and allows the possibility of a better control over the shape of an airfoil during the course of optimization. The details of the PARSEC representation and the bounds for the PARSEC variables used in this study are presented in the following section.

PARSEC Representation

Unlike common engineering design optimization problems where the number and nature of the variables are fixed, the number of variables of an airfoil optimization problem are influenced by choice of the airfoil representation scheme. The PARSEC representation is particularly attractive as it uses a small number of design variables, all of which are related to some properties of the shape. It parameterizes the upper and the lower airfoil surfaces using polynomials

Table 1 Upper and lower bounds of the variables

Variable number	Variable name	Upper bound	Lower bound
1	X_{UP}	0.5	0.3
2	Z_{UP}	0.075	0.05
3	Z_{TE}	0	0
4	r_{le}	0.0085	0.0055
5	Z_{XXUP}	−0.4	−0.6
6	α_{TE}	−8	−12
7	X_{LO}	0.42	0.28
8	Z_{LO}	−0.050	−0.075
9	Z_{XXLO}	0.85	0.55
10	β_{TE}	−9.5	−14.5
11	ΔZ_{TE}	0	0

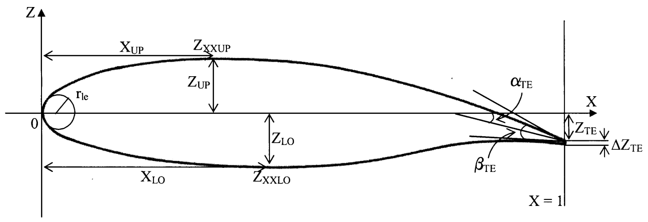

Fig. 3 Variables in the PARSEC representation scheme.

in coordinates X and Z as

$$Z = \sum_{n=1}^{6} a_n X^{n-\frac{1}{2}} \tag{6}$$

where a_n are real coefficients. The parameters of a PARSEC representation include the leading-edge radius r_{le}, upper and lower crest heights Z_{UP}, Z_{LO} and location X_{UP}, X_{LO}, curvatures at the upper and lower crests Z_{XXUP}, Z_{XXLO}, trailing-edge thickness ΔZ_{TE} and ordinate Z_{TE}, and direction and wedge angle α_{TE}, β_{TE}. The parameters are schematically shown in Fig. 3. In the examples, the trailing-edge thickness and the trailing-edge ordinate are set to 0. See Table 1.

Flow Solver

In this study we used an Euler code to compute the flow around the airfoil. The governing equations are solved using a finite volume formulation as proposed in Jameson et al.[26] Details of these standard procedures are outlined in Refs. 27 and 28. The field grid is generated algebraically using conformal mapping methods to create an O-grid size of 161 by 33 to represent the airfoil. In the interest of robustness in the multigrid computation, full coarsening up till a minimum of four cells was used with lower values of Courant–Friedrichs–Lewy numbers set at seven.

Numerical Examples

For the single-objective examples, we used a swarm size of 20 and allowed it to fly for 100 time steps. For the multiobjective problem, we used a swarm size of 100 and allowed it to fly for 50 time steps. The designs were evaluated at a Mach number of 0.73 and an angle of attack of 2 deg. A single-objective function evaluation using the preceding computational-fluid-dynamics solver takes around 25 s of CPU time.

Example 1

The first example deals with an inverse design of a RAE2822 airfoil. It is a single-objective unconstrained minimization problem. The objective is to minimize

$$f = \sum_{i=1}^{M} \left(C_{pi} - C_{pi}^{T}\right)^2$$

where C_{pi} is the pressure coefficient at the ith location of the airfoil, and C_{pi}^{T} is the target pressure at the ith location computed based on the Euler code for the RAE2822 airfoil. The initial, final, and the

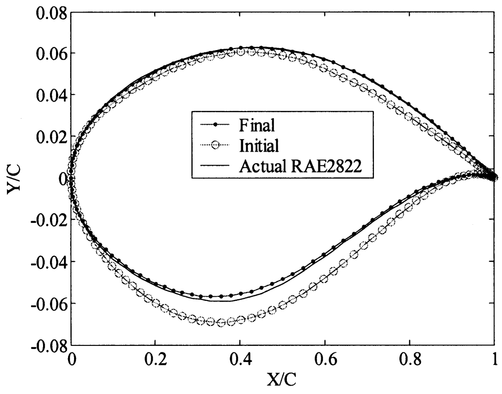

Fig. 4 Initial airfoil shape.

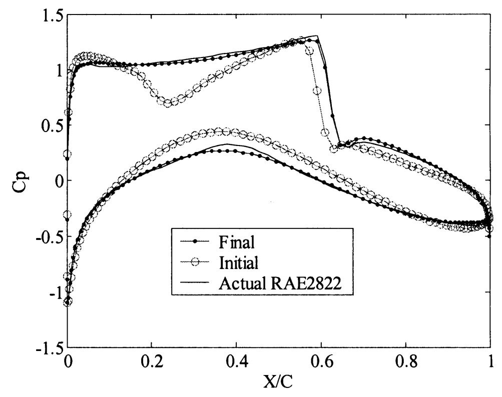

Fig. 5 Initial C_p distribution.

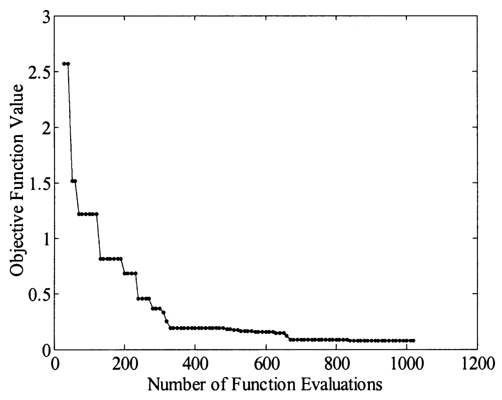

Fig. 6 Progress plot.

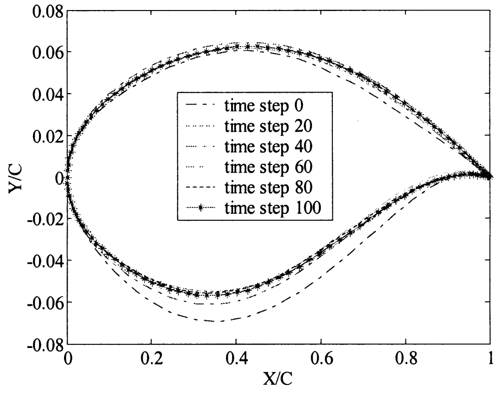

Fig. 7 Evolution of shape.

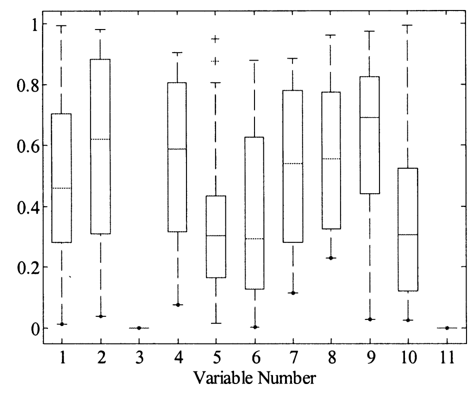

Fig. 8 Box plot of initial swarm.

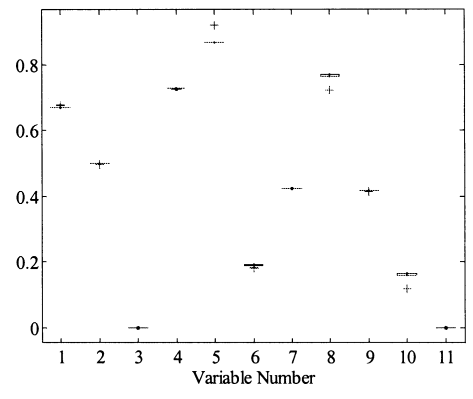

Fig. 9 Box plot of final swarm.

actual RAE2822 airfoil shapes are presented in Fig. 4, while their corresponding C_p distributions are presented in Fig. 5. The objective function progress plot in Fig. 6 shows a reduction in the objective function value from 2.567 to 0.0737 within 1020 function evaluations. The evolution of airfoil shapes is presented in Fig. 7. From Figs. 4 and 5, we can observe a reasonably close match between the shapes and their C_p distributions. A quantitative comparison between different methods is difficult as they use different shape representation schemes and different variable bounds. However, we observe a similar reduction in the objective function value for the same computational cost in Refs. 7 and 8, which use a GA and a PARSEC shape representation scheme to optimize airfoil shapes.

A box plot is used to show the convergence of our algorithm. A box plot typically shows the limits of the variables, their median, and the 25th and 75th percentile. The scaled (0-1) box plots of the initial and the final individuals of the swarm are presented in Figs. 8 and 9. Figure 8 shows that the variables are largely distributed over the entire variable range; however, Fig. 9 indicates that the variables have converged to specific values, and hence the limits, median, and the percentiles have also converged to a particular region of the variable space.

Example 2

The second example deals with a single-objective constrained optimization problem that aims to minimize the drag coefficient with a constraint on the minimum lift coefficient. The objective is to minimize $\boldsymbol{f} = \boldsymbol{C_d}$, subject to $\boldsymbol{C_l} \geq 0.824$. The final airfoil shape and the C_p distribution are presented in Figs. 10 and 11 for three independent runs. The progress of the optimization algorithm is also presented in Fig. 12 for three independent runs. One can observe that the independent runs consistently reached the same objective function value starting from different initial values. However, one can observe the changes in optimal shapes and their differences with the RAE2822 airfoil shape. This indicates the multimodal behavior of the objective function, that is, multiple airfoil shapes exist with almost same objective function value and a designer needs to consider additional performance criteria (e.g., performance at multiple operating conditions, etc.) to select one among them.

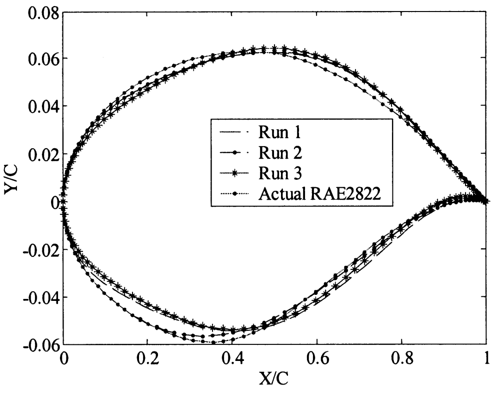

Fig. 10 Airfoil shapes (different runs).

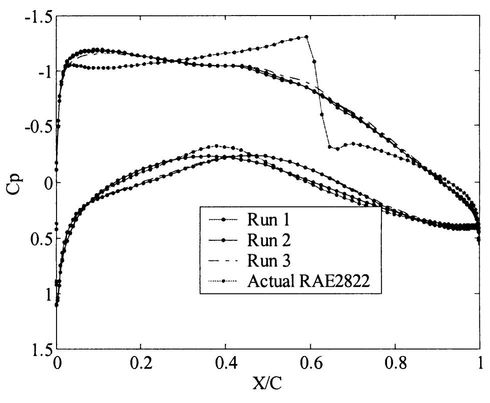

Fig. 11 C_p distributions.

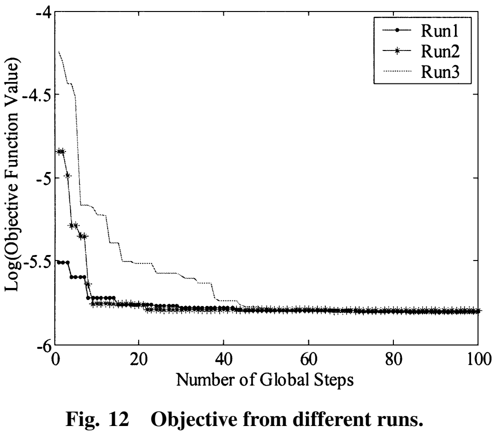

Fig. 12 Objective from different runs.

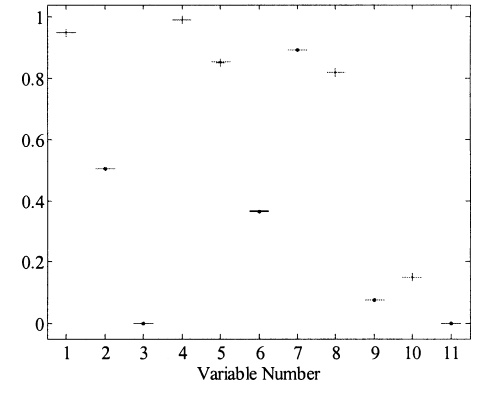

Fig. 13 Box plot of final swarm (run 1).

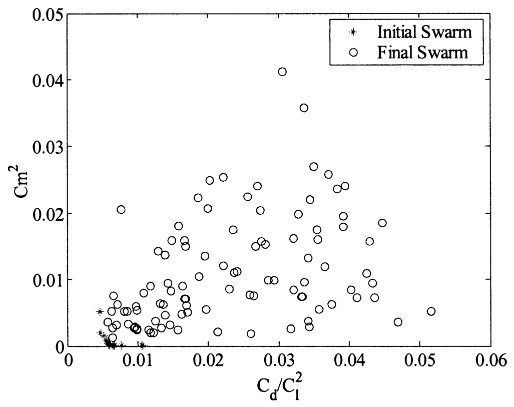

Fig. 14 Initial swarm.

The box plot of the final swarm from run 1 is presented in Fig. 13, which shows the convergence of the solution. The solutions converged to [$X_{\mathrm{UP}} = 0.459$, $Z_{\mathrm{UP}} = 0.0582$, $Z_{\mathrm{TE}} = 0.0$, $r_{\mathrm{le}} = 0.0080$, $Z_{\mathrm{XXUP}} = -0.469$, $\alpha_{\mathrm{TE}} = -8.35$, $X_{\mathrm{LO}} = 0.333$, $Z_{\mathrm{LO}} = -0.052$, $Z_{\mathrm{XXLO}} = 0.792$, $\beta_{\mathrm{TE}} = -11.18$, and $\Delta Z_{\mathrm{TE}} = 0.0$]. The C_l of the RAE2822 at an angle of attack of 2 deg and at Mach number of 0.73 is 0.824, and the C_d is 0.0073. The C_d reduced to 0.0031 while maintaining the C_l at 0.824 in our optimized design. This is the simple case of redesigning the airfoil to reduce drag. Notice that the shock present in the original RAE 2822 airfoil is no longer present as one expects.

Example 3

The third example deals with a multiobjective optimization formulation that aims to minimize $f_1 = C_d/C_l^2$ and $f_2 = C_m^2$ for an angle of attack of 2 deg and a Mach number of 0.73. The problem involves minimization of two objectives. This formulation appears in the works of Vicini and Quagliarella,[7] although in the context of the ONERA M6 wing section design. There are no constraints imposed on the lift coefficient in this study, but such a constraint can easily be imposed as highlighted in example 2.

Classical optimization methods are in general not efficient for multiobjective problems as they often lead to a single solution instead of a set of optimal solutions. Multiple runs of the same method cannot guarantee a different point on the nondominated front each time, and some methods cannot even handle problems with multiple optimal solutions.[4] Evolutionary or population-based methods maintain a set of solutions during its course of search and thus can result in a set of nondominated solutions in a single run. A widely differing set of nondominated solutions can be generated using a diversification strategy (e.g., niching) within such population-based algorithms.

In the present example, the initial and the final swarm are presented in Fig. 14. The migration of the individuals towards the nondominated front is apparent. The final swarm is plotted on an enlarged scale in Fig. 15. It shows that the individuals in the final swarm converged to the nondominated front with C_m^2 varying between $2.968850E-14$ and 0.005233231 and C_d/C_l^2 varying between 0.004584012 and 0.01065582 using 1774 function evaluations with 32 solutions along the nondominated front.

To assess the solutions along the nondominated front, they have been arbitrarily divided into two groups one with $C_d/C_l^2 < 0.0055$ and the other with $C_d/C_l^2 > =0.0055$. The box plots of the groups are presented in Figs. 16 and 17, while the two extreme designs in the current set, that is, the lowest C_d/C_l^2 design and the lowest C_m^2 design, are presented in Fig. 18. The corresponding pressure distribution of the airfoils is presented in Fig. 19.

The box plots indicate that for both low C_d/C_l^2 and low C_m^2 designs the leading-edge radius r_{le} tends to increase and shifts the pressure peaks towards the leading edge. To decrease drag, the Z_{UP} tends to decrease. The rate of change of gradient at the thickness region on the lower surface Z_{XXLO} also tends to decrease. For low C_d/C_l^2 designs, we also observe that α_{TE} tends to increase, that is, move towards the lower bound with X_{LO} also tending to increase,

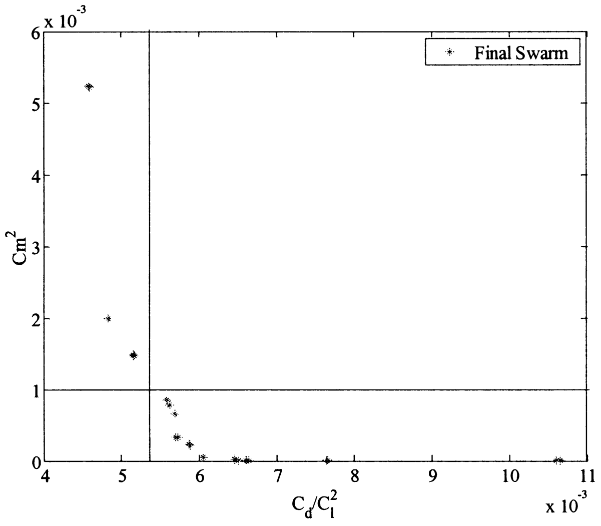

Fig. 15 Final swarm.

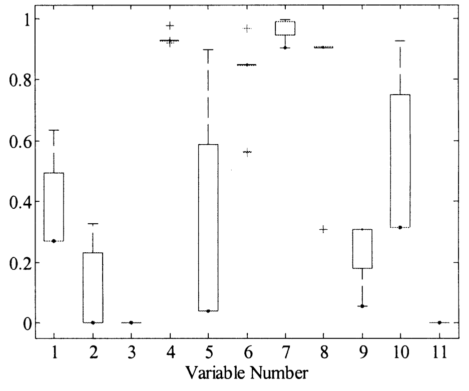

Fig. 16 Box plot for $C_d/C_l^2 < 0.0055$.

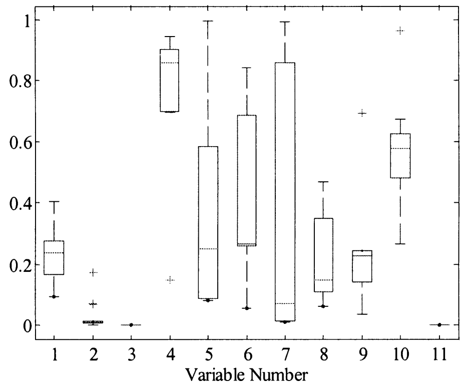

Fig. 17 Box plot for $C_d/C_l^2 >= 0.0055$.

that is, the thickness on the lower half of the airfoil spreads toward the trailing edge. For low C_m^2 designs, we observe that Z_{LO} tends to decrease. This makes the pressure peaks move toward the leading edge, which reduces the magnitude of the pitching moment. These observations are consistent with expected airfoil behaviors.

The process of selecting a single design from the set of nondominated designs is the next step in any multiobjective design process. One would usually list representative parametrically diverse solutions from the nondominated set as potential alternative designs that qualify for further consideration. Such designs are usually few (typically fewer than 10), and multiple-attribute decision-making methods[29] can be applied to select the best based on an additional set of criteria.

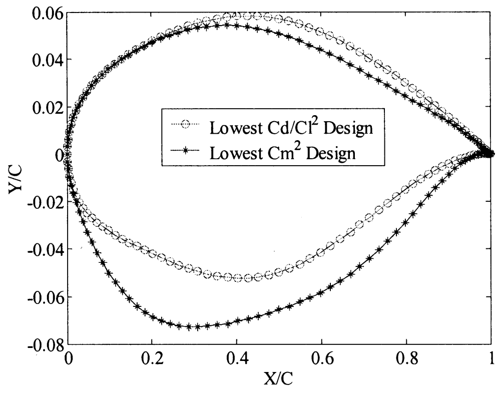

Fig. 18 Best C_d/C_l^2 and best C_m^2 shapes.

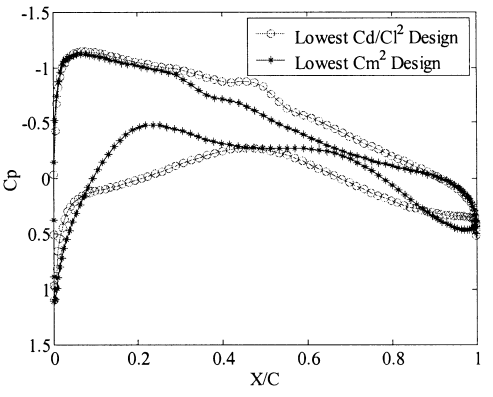

Fig. 19 Corresponding C_p.

Summary

In this paper, we identified the desired features of an airfoil shape optimization algorithm and introduced a swarm algorithm to effectively and efficiently solve such problems. The algorithm is particularly attractive as it does not require additional user inputs for scaling and aggregation of objectives and constraints and its computational efficiency is comparable with existing stochastic optimization methods. Although the use of nondominated sorting to handle constraints is an expensive operation, it is meaningful for problems where the objective and the constraint function evaluations are equally or even more expensive. The swarm attempts to improve the performance of all individuals including the bad ones through information acquisition from leaders, unlike mating in an evolutionary/genetic algorithm where only the good parents participate. Such a socialistic behavior on one hand leads to expensive evaluations, whereas on the other hand provides a scope for a wider exploration that is useful for problems that are highly nonlinear.

The numerical examples in the present study highlight the swarms' convergence capabilities and its flexibility to handle various formulations. For the inverse design problem, the swarm derived the final shape and the C_p distribution that is close to the actual RAE2822 airfoil using 1020 function evaluations, which compares favorably with existing evolutionary algorithms. As for the C_d minimization problem, good convergence was observed. Because the algorithm does not use penalty functions to handle constraints, no weights or scaling inputs are required from the user. For the multiobjective problem, the algorithm converged to the nondominated front with a mere 1774 function evaluations. Although swarm algorithm is attractive in its present form, improvements in efficiency and speed could be achieved via the development of meaningful information exchange operators and a parallel implementation of the algorithm.

References

[1]Hicks, R. M., and Henne, P. A., "Wing Design by Numerical Optimization," *Journal of Aircraft*, Vol. 15, 1978, pp. 407–412.

[2]LeGresley, P. A., and Alonso, J. J., “Airfoil Design Optimization Using Reduced Order Models Based on Proper Orthogonal Decomposition,” AIAA Paper 2000-2545, June 2000.

[3]Holst, T. L., and Pulliam, T. H., “Aerodynamic Shape Optimization Using a Real-Number-Encoded GA,” AIAA Paper 2001-2473, June 2001.

[4]Giannakoglou, K. C., “Design of Optimal Aerodynamic Shapes Using Stochastic Optimization Methods and Computational Intelligence,” *Progress in Aerospace Sciences*, Vol. 38, 2002, pp. 43–76.

[5]Marco, N., Desideri, J. A., and Lanteri, S., “Multiobjective Optimization in CFD by Genetic Algorithms,” Institut National de Recherche en Informatique et en Automatique, Rapport de recherché 3686, 1999.

[6]De Falco, I., Cioppa, A. D., Iazzetta, A., and Tarantino, E., “Evolutionary Algorithms for Aerofoil Design,” *International Journal of Computational Fluid Dynamics*, Vol. 11, 1998, pp. 51–77.

[7]Vicini, A., and Quagliarella, D., “Inverse and Direct Design Using a Multiobjective Genetic Algorithm,” *AIAA Journal*, Vol. 35, No. 9, 1997, pp. 1499–1505.

[8]Lepine, J., Guibault, F., and Trepanier, J. Y., “Optimized NURBS Geometrical Representation for Aerodynamic Design of Wings,” *AIAA Journal*, Vol. 39, No. 11, 2001, pp. 2033–2041,

[9]Quagliarella, D., and Vicini, A., “Combining Genetic Algorithms and Gradient Based Optimization Techniques,” *Genetic Algorithms in Engineering and Computer Science*, edited by G. Winter, J. Periaux, M. Galan, and P. Cuesta, Wiley, Chichester, England, U.K., 1995.

[10]Quagliarella, D., and Cioppa, A. D., “Genetic Algorithms Applied to the Aerodynamic Design of Transonic Airfoils,” *Journal of Aircraft*, Vol. 32, No. 4, 1995, pp. 889, 890.

[11]Obayashi, S., and Takanashi, S., “Genetic Optimization of Target Pressure Distribution for Inverse Design Methods,” *AIAA Journal*, Vol. 34, No. 5, 1996, pp. 881–886.

[12]Jameson, A., “Aerodynamic Design via Control Theory,” *Journal of Scientific Computing*, Vol. 3, 1988, pp. 233–260.

[13]Jameson, A., “Optimum Aerodynamic Design Using CFD and Control Theory,” AIAA Paper 95-1729, June 1995.

[14]Messac, A., “Physical Programming: Effective Optimization for Computational Design,” *AIAA Journal*, Vol. 34, No. 1, 1996, pp. 149–158.

[15]Messac, A., “From Dubious Construction of Objective Functions to the Application of Physical Programming,” *AIAA Journal*, Vol. 38, No. 1, 2000, pp. 155–163.

[16]Messac, A., and Ismail-Yahaya, A., “Multiobjective Robust Design Using Physical Programming,” *Structural and Multidisciplinary Optimization*, Vol. 23, No. 5, 2002, pp. 357–371.

[17]Kalyanmoy, Deb., *Multi-Objective Optimization Using Evolutionary Algorithms*, Wiley, New York, 2002.

[18]Michalewicz, Z., “A Survey of Constraint Handling Techniques in Evolutionary Computation Methods,” *Proceedings of the Fourth Annual Conference on Evolutionary Programming*, edited by J. McDonnell, R. Reynolds, and D. Fogel, MIT Press, Cambridge, MA, 1995, pp. 135–155.

[19]Mattson, C. A., and Messac, A., “Handling Equality Constraints in Robust Design Optimization,” AIAA Paper 2003-1780, April 2003.

[20]Kennedy, J., and Eberhart, R. C., “Particle Swarm Optimization,” *Proceedings of the 1995 IEEE International Conference on Neural Networks*, IEEE Service Center, Piscataway, NJ, 1995, pp. 1942–1948.

[21]Kennedy, J., and Eberhart, R. C., “A Discrete Binary Version of the Particle Swarm Algorithm,” *Proceedings of the 1997 IEEE Conference on Systems, Man and Cybernetics*, IEEE Service Center, Piscataway, NJ, 1997, pp. 4104–4109.

[22]Ray, T., and Saini, P., “Engineering Design Optimization Using a Swarm with an Intelligent Information Sharing Among Individuals,” *Engineering Optimization*, Vol. 33, No. 6, 2002, pp. 735–748.

[23]Ray, T., and Liew, K. M., “A Swarm with an Effective Information Sharing Mechanism for Unconstrained and Constrained Single Objective Optimization Problems,” *Proceedings of the Congress on Evolutionary Computation*, IEEE Service Center, Piscataway, NJ, 2001, pp. 75–80.

[24]Ray, T., and Liew, K. M., “A Swarm Metaphor for Multiobjective Design Optimization,” *Engineering Optimization*, Vol. 34, No. 2, 1992, pp. 141–153.

[25]Ray, T., and Tsai H. M., “Some Issues in NURBS Representation of Airfoil Shapes for Optimization,” *Proceedings of the 8th International Conference on Grid Generation in Computational Field Simulations*, June 2002.

[26]Jameson, A., Schmidt, W., and Turkel, E., “Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge–Kutta Time-Stepping Schemes,” AIAA Paper 81-1259, 1981.

[27]Jameson, A., “Time Dependent Calculations Using Multigrid with Applications to Unsteady Flows past Airfoils and Wings,” AIAA Paper 91-1596, 1991.

[28]Jameson, A., “Solution of the Euler Equations for Two Dimensional Transonic Flow by a Multigrid Method,” *Applied Mathematics and Computation*, Vol. 13, 1983, pp. 327–356.

[29]Hwang, C. L., and Yoon, K., “Multiple Attribute Decision Making,” *Lecture Notes in Economics and Mathematical Systems*, No. 186, Springer-Verlag, New York, 1981.

A. Messac
Associate Editor