A comparative evaluation of genetic and gradient-based algorithms applied to aerodynamic optimization

David W. Zingg* — Marian Nemec — Thomas H. Pulliam****

** Senior Canada Research Chair in Computational Aerodynamics*
University of Toronto Institute of Aerospace Studies
4925 Dufferin St., Toronto, Ontario Canada,
dwz@oddjob.utias.utoronto.ca

*** NASA Ames Research Center, Moffett Field, California, USA*
{Marian.Nemec,Thomas.H.Pulliam}@nasa.gov

ABSTRACT. *A genetic algorithm is compared with a gradient-based (adjoint) algorithm in the context of several aerodynamic shape optimization problems. The examples include singlepoint and multipoint optimization problems, as well as the computation of a Pareto front. The results demonstrate that both algorithms converge reliably to the same optimum. Depending on the nature of the problem, the number of design variables, and the degree of convergence, the genetic algorithm requires from 5 to 200 times as many function evaluations as the gradientbased algorithm.*

RÉSUMÉ. *La comparaison entre un algorithme génétique et un algorithme basé sur le calcul d'un gradient (méthode de l'adjoint) est proposée dans le cadre de différents problèmes d'optimisation de forme aérodynamique. Les exemples proposés comprennent des problèmes d'optimisation single-point et multipoint ainsi que le calcul de fronts Pareto. Les résultats obtenus démontrent que les deux algorithmes convergent vers la même solution. En fonction de la nature du problème, du nombre de variables de formes, et du degré de convergence, l'algorithme génétique nécessite de 5 à 200 fois plus d'évaluations de fonction coût que l'algorithme de type gradient.*

KEYWORDS: *aerodynamics, computational fluid dynamics, adjoint methods, genetic algorithms.*

MOTS-CLÉS : *aérodynamique, mécanique des fluides numérique, méthodes de l'adjoint, algorithmes génétiques.*

DOI:10.3166/REMN.17.103-126

1. Introduction

As computer speeds continue to obey Moore's law, numerical optimization techniques are being applied to increasingly complex and demanding problems. These problems are characterized by relatively expensive objective function evaluations, a large number of design variables, and a significant number of objectives and constraints. Optimization of an aerodynamic configuration, such as an aircraft wing, based on the numerical solution of the Reynolds-averaged Navier-Stokes equations is an example of such a problem. A typical objective in the design of a wing is to minimize the drag coefficient at a prescribed lift coefficient. Multiple competing objectives arise since the wing must be able to operate efficiently over a range of operating conditions. The need for satisfactory off-design performance and aircraft handling qualities introduces additional objectives or constraints, depending on how the problem is posed. Structural and manufacturing considerations lead to further constraints. Several hundred shape parameters may be needed to define a geometric parameterization of sufficient flexibility. Finally, the numerical solution of the Reynolds-averaged Navier-Stokes equations for turbulent flow over a complex geometry is itself a challenging and expensive task. These characteristics are typical of many partial-differential-equation (PDE)-based, or simulation-based, optimization problems.

There are several different issues involved in posing and solving such multiobjective optimization problems and numerous strategies for addressing these issues. One key consideration is whether to use a gradient-based or a gradient-free optimization algorithm. Each approach has advantages and disadvantages; these are discussed in the next section. The simplest method for computing gradients is the finite-difference method. If round-off error is a concern, then the complex-step method can be used (Anderson *et al.*, 2001). In either case, the cost of evaluating the gradient is proportional to the number of design variables. Since the number of iterations needed for convergence of a gradient-based algorithm is typically proportional to the number of design variables, the overall cost with a finite-difference approximation of the gradient is proportional to the square of the number of design variables. The adjoint approach (Jameson, 1988) provides a more computationally efficient alternative in which the gradient calculation is almost independent of the number of design variables, and hence the overall cost of the optimization is roughly linearly proportional to the number of design variables.

Since gradient-based and gradient-free optimization algorithms each have strengths and weaknesses, the choice is problem dependent. The objective of this paper is to provide data to help guide the selection of the appropriate approach for a specific task. Among gradient-based algorithms we consider only the adjoint approach; among gradient-free algorithms we consider only a genetic algorithm. Our emphasis is on a determination of the relative efficiency of the two algorithms for a representative class of optimization problems: aerodynamic shape optimization in two dimensions. Efficiency is one of several important considerations in choosing an approach. A key aspect of our comparisons is that the two algorithms are applied to

identical design spaces, including the geometry parameterization (design variables), objective function (flow solver), and constraints (geometric constraints representing structural and manufacturing considerations).

The potential benefits of aerodynamic optimization were recognized early on by Hicks *et al.* (1974) and Hicks *et al.* (1978). However, their work was constrained by the limitations of computers and algorithms at that time and furthermore by the use of finite differences for the gradient computation. Pironneau (1974) introduced the adjoint approach to optimization in fluid dynamics, but it was not until the work of Jameson (1988) that the approach became popular in aerodynamic optimization. There are two adjoint formulations currently in use, the *continuous* approach in which the adjoint equations are derived from the governing partial differential equations and then discretized and the *discrete* approach in which the discretized governing equations are differentiated to obtain the adjoint equations. Examples of the discrete approach can be found in (Elliott *et al.*, 1997; Nielsen *et al.*, 1999; Nemec *et al.*, 2002), while Reuther *et al.* (1999) provide an example of the continuous approach.

Genetic (or evolutionary) algorithms (Goldberg, 1989) have also become popular in aerodynamic optimization, especially for multiobjective optimization problems (Deb, 2002; Coello Coello, 2002). Such algorithms use a combination of *exploitation* of the positive characteristics of the existing set of solutions and *exploration* of other areas of the design space to find an optimum. Examples of genetic algorithms applied to aerodynamic optimization can be found in (Obayashi, 1997; Vicini *et al.*, 1997; Marco *et al.*, 1999; Oyama, 2000; Holst *et al.*, 2001; Giannakoglou, 2002).

One means of dealing with multiple competing objectives is through the concepts of dominance and Pareto optimality. A given solution is said to *dominate* another if it is not inferior with respect to any objective and is superior with respect to at least one. The Pareto optimal set is the set of solutions that are non-dominated by any other member of the total set of feasible solutions. The image of the Pareto optimal set in the objective function space is known as the *Pareto front*. The Pareto front provides the set of solutions of interest to the designer and reveals tradeoffs between objectives. A simple means of computing a Pareto front is the weighted-sum method, in which a single objective is formed by a weighted sum of the competing objectives. By varying the weights, the Pareto front can be generated. This approach fails if the Pareto front is nonconvex or poorly scaled. For our present application, the Pareto front is convex and well scaled, so the weighted-sum method is effective. An alternative approach, the *dominance Pareto front* technique (Goldberg, 1989; Deb, 2002; Coello Coello, 2002), is often applied in conjunction with evolutionary algorithms. This technique uses non-dominated sorting and selection coupled with a genetic algorithm to move a population toward the Pareto front and does not suffer from convexity or scaling problems.

In this paper, we compare the performance of the gradient-based discrete-adjoint optimization algorithm of Nemec *et al.* (2004) with that of the genetic algorithm of Holst *et al.* (2001) for several single and multiobjective aerodynamic shape optimization problems. Of particular interest is the dependence of the relative cost of the two algorithms on the number of design variables and the degree of convergence needed.

2. Genetic and gradient-based algorithms for optimization

In this section, we give a brief overview of the strengths and weaknesses of genetic and gradient-based algorithms in the context of optimization problems where each function evaluation is relatively expensive. We focus on problems in which the number of design variables is significantly greater than the number of objectives and constraints. In this context, the adjoint method is the most efficient approach for gradient-based optimization; hence other approaches, such as the flow sensitivity approach, are not discussed.

Once the gradient is computed, there are several options for finding a minimum. For constrained problems, sequential quadratic programming methods are popular (Gill *et al.*, 2002). Alternatively, an unconstrained formulation can be obtained by adding the constraints as a penalty term to the objective function. For unconstrained problems, quasi-Newton methods are effective, typically in conjunction with a line search procedure (Dennis *et al.*, 1983). In either case, one can expect the cost of the optimization to be roughly proportional to the number of design variables. Rapid convergence is the primary advantage of a gradient-based method. Clearly, proper exploitation of gradient information can significantly enhance the speed of convergence in comparison with a method that does not compute gradients. Another feature of gradient-based methods is that they provide a clear convergence criterion. If the gradient is reduced by many orders of magnitude, one can be confident that at least a local optimum has been reached.

One of the key disadvantages of gradient-based methods using adjoints is the development cost. Whether the linearization is performed by hand or using automatic differentiation, with a complex code this can be time-consuming. Furthermore, each time the code is altered, the adjoint computation may need to be modified. Another potential weakness of gradient-based methods is that they are relatively intolerant of difficulties such as noisy objective function spaces, inaccurate gradients, categorical variables, and topology optimization. Another oft-mentioned disadvantage of gradient-based methods is that they find a local rather than a global optimum. However, in many engineering design contexts this is unlikely to be an issue, since the highly constrained nature of the design problem inhibits multimodality.

The key disadvantages of gradient-based methods are precisely the strengths of genetic algorithms. First, genetic algorithms treat the function evaluation as a "black box." Consequently, development cost is minimal. Second, they are tolerant of noise in the objective function and have no difficulty with categorical variables or topology changes. Furthermore, in principle, genetic algorithms find a global optimum. The key disadvantage associated with genetic algorithms is that they can converge very slowly, especially near an optimum. A second weakness is that determining a termination criterion is not straightforward.

Of course, research is underway to address the weaknesses of both genetic and gradient-based algorithms. For example, the cost of genetic algorithms can be reduced through the use of surrogates or response surfaces. Various hybrid algorithms are also

under development (Vicini *et al.*, 1999; Poloni *et al.*, 2000). Furthermore, there exist techniques, such as variable-fidelity optimization, that can enhance the performance of both algorithm classes.

The issue of noisy objective function spaces remains open. In aerodynamic shape optimization, such behavior can be encountered as a result of poor flow solver convergence in some regions of the design space, usually associated with flow separation and unsteadiness. One does not wish to resolve the noise but to determine the optimal design with respect to meaningful design variable scales. It is not yet well understood how well either a gradient-based[1] or a genetic algorithm can deal with such design problems. Although a genetic algorithm can proceed in principle, insufficiently converged solutions can produce misleading results that can lead the genetic algorithm into nonoptimal areas of the design space.

The above strengths and weaknesses must be considered in choosing an optimization algorithm for a specific problem class. A key tradeoff is between the relatively high development cost of the gradient-based method using an adjoint formulation and the relatively high computational cost of the genetic algorithm. The more frequently the algorithm is to be used, the more beneficial the gradient-based algorithm becomes. Clearly, a quantitative assessment of the computational cost of the two algorithms is needed to make an intelligent choice for a given class of problems.

3. Optimization process

We consider shape optimization of two-dimensional airfoil sections, including single-point, multipoint, and multiobjective optimization. The flow is governed by the compressible thin-layer Reynolds-averaged Navier-Stokes equations along with the one-equation Spalart-Allmaras turbulence model. The optimization process begins with an initial airfoil around which a mesh is generated using an elliptic mesh generator. A B-spline curve is fit to the initial airfoil surface using a prescribed number of control points, leading to a slightly modified airfoil. The mesh is perturbed to conform to the new airfoil using a simple algebraic procedure. Thus we have an analytically defined airfoil whose shape is controlled by a relatively small number of parameters, the locations of the B-spline control points. Details of the geometry parameterization and the algebraic mesh modification technique can be found in (Nemec, 2002).

The optimization problem is posed in terms of operating conditions, design variables, objective functions, and constraints. The similarity parameters, freestream Mach and Reynolds numbers, provide the necessary operating conditions. The design variables are a specified subset of the B-spline control point locations. In addition, in all cases studied here, the angle of attack is also a design variable. Objective functions for each case are defined below. Geometric constraints, which represent structural and manufacturing considerations, are imposed by requiring the airfoil thickness to be

1. Adjoint algorithms have been developed specifically for unsteady flows; these are not addressed in this paper.

greater than or equal to a specified value at prescribed chordwise locations. These are added to the objective function as penalty terms.

When the values of the design variables corresponding to the locations of B-spline control points are changed, a new airfoil shape is produced. The algebraic mesh movement algorithm is used to perturb the original mesh to conform to the altered geometry. The goal of the optimization algorithm is to compute the values of the design variables that minimize the objective function, which includes the constraints. The genetic optimization algorithm utilizes the flow solver ARC2D (Pulliam, 1985) to perform the objective function evaluation, *i.e.* to solve the governing equations, for a given geometry. The spatial derivatives in the governing equations are discretized using second-order centered finite-difference approximations together with scalar artificial dissipation through a generalized curvilinear coordinate transformation. Iteration to a steady state is achieved by an implicit approximate factorization algorithm. The gradient-based optimization algorithm uses the same spatial discretization together with a Newton-Krylov algorithm to iterate to a steady state (Pueyo *et al.*, 1998). Identical parameters are used in the spatial discretizations, and consistency between the two solvers was verified. For a given set of design variables (hence identical meshes) and operating conditions, the two solvers produce identical solutions. The Newton-Krylov solver converges somewhat more rapidly than the approximately-factored solver. This does not affect our cost estimates, however, since we measure cost in terms of function evaluations, so the actual expense of a function evaluation is not considered.

4. Gradient-based algorithm using the discrete-adjoint formulation

The present algorithm is described in detail in (Nemec *et al.*, 2002; Nemec *et al.*, 2004). The compressible Navier-Stokes equations are solved with a Newton-Krylov method in which the linear system at each Newton iteration is solved using the generalized minimal residual method (GMRES) preconditioned with an incomplete lower-upper (ILU) factorization with limited fill. The gradient is computed using the discrete-adjoint method. The discretized flow and turbulence model equations have been differentiated by hand to obtain the Jacobian. The adjoint equation is solved using the same preconditioned Krylov solver used by the flow solver. A new set of design variables is found using a quasi-Newton optimizer in which an estimate of the inverse Hessian based on the BFGS (Broyden-Fletcher-Goldfarb-Shanno) rank-two update formula (Nocedal *et al.*, 1999) is used to compute a search direction. If the initial step does not produce sufficient progress toward the minimum, the step size is determined using a cubic line search, which terminates when the strong Wolfe conditions (Nocedal *et al.*, 1999) are satisfied. A detailed evaluation of the algorithm was performed with emphasis on the accuracy and efficiency of the gradient computation and the efficiency of the flow solver (Nemec, 2002). The resulting algorithm provides a highly efficient approach for aerodynamic design problems governed by the Navier–Stokes equations. The preconditioning strategy, in particular, has been optimized for both the flow solution and the gradient evaluation. Nemec (2002) provides a detailed

development and validation of the algorithm for single and multipoint airfoil shape optimization.

5. Genetic algorithm

A generational genetic algorithm (Holst *et al.*, 2001) is used. It combines a number of ranking and selection techniques, mutations, and perturbations performed on the exploited *chromosomes* to produce the exploration set for the next generation. Briefly, objective function (or *fitness*) values are computed using ARC2D for a set of chromosomes (or a *population*) consisting of various sets of values of the design variables (or *genes*). The chromosomes are then ranked according to their fitness values, and a new intermediate generation is selected from the current population using a number of selection algorithms (e.g. tournament selection, where 3 randomly chosen *parents* compete based on ranking, the winners surviving to form the intermediate generation). The intermediate generation is then processed by cross-over and mutation strategies to produce the new generation of chromosomes. The genetic algorithm process is repeated until convergence, which is difficult to define for algorithms of this type. Typically, convergence of a genetic algorithm is assumed when the objective function ceases to improve, although this is not always a good measure, since the genetic algorithm can wander around for many generations until a new region of convergence is found. There are a number of genetic algorithm parameters involved, such as the number of chromosomes in a generation, the probability of selection, mutation, and cross-over, and the convergence criterion. These were chosen to optimize the convergence speed of the algorithm (Holst *et al.*, 2001). Efficiency is obtained on a parallel computing system by parallel evaluation of each chromosome's fitness value. Most of the results shown for the genetic algorithm were performed on 32 processors, where the number of processors chosen was equal to the population size of the optimization (which is 32 unless otherwise stated).

With a genetic algorithm, a Pareto front can be found using either the weighted-sum or the dominance Pareto front technique. In the latter case, a chromosome archival strategy is used, where new points found on the Pareto front are stored in an accumulation file, producing a well defined front. The archival file can be used in the ranking process and also as part of the selection pool.

6. Optimization problems

Details of the five optimization problems studied are provided in this section. The problems are referenced A through E and have been chosen to be representative of realistic optimization problems. However, design of a practical airfoil requires more formal definition and treatment of structural and manufacturing constraints, operating conditions, and off-design performance requirements. Therefore, the optimized airfoils produced are not suitable for practical use. Problems A through C demonstrate the effect of the number of design variables on the convergence of the two algorithms.

Table 1. *Thickness constraints for optimization problems A through C*

constraint n°	1	2	3	4	5	6
location $(\%c)$	5.0	35.0	65.0	85.0	95.0	99.0
thickness $(\%c)$	4.0	11.0	4.0	2.6	1.2	0.2

Table 2. *B-spline control points and design variables for problems A through C*

problem	n° of control points	n° of design variables (includes angle of attack)
A	15	9
B	25	19
C	45	35

Problem D permits a comparison of the two algorithms in the context of a multipoint optimization problem, while Problem E is a two-objective example.

6.1. *Problems A-C: lift-to-drag ratio maximization*

For this problem the objective is to maximize the lift-to-drag ratio of a single-element airfoil at a Mach number of 0.25 and a Reynolds number of 2.88 million in a fully turbulent flow. Hence the objective function to be minimized is

$$J = C_{\mathrm{d}}/C_{\mathrm{l}} + \omega_{\mathrm{T}} T \qquad [1]$$

where C_{d} and C_{l} are the drag and lift coefficients, respectively, T is the penalty term from the thickness constraints, and ω_{T} is a weight applied to the penalty term equal to 0.05 for this case. The thickness constraints are given in Table 1 expressed as a percentage of the airfoil chord c.

The initial shape for the gradient-based algorithm is the NACA 0012 airfoil at an angle of attack of 9 degrees. The mesh has a "C" topology with 201 points in the streamwise direction and 45 in the normal direction. Although a finer mesh is required if high accuracy is needed, this mesh density is adequate for the current purpose. For Problem A, the geometry is parameterized with 15 B-spline control points, as shown in Figure 1. Three control points are frozen at the leading edge and four at the trailing edge. Therefore, four control points on each surface serve as the shape design variables. Combined with the angle of attack, there are thus nine design variables. For the genetic algorithm, the initial population is based on a random sampling between minimum and maximum values for all of the genes, *i.e.* design variables.

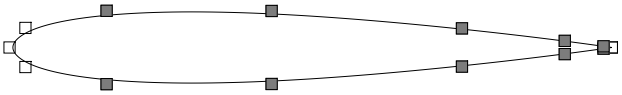


Figure 1. *B-spline control points for optimization problems A, D, and E. The shaded points are the design variables for problem E*

Table 3. *Thickness constraints for optimization problem D*

constraint n°	1	2	3	4
location $(\%c)$	35.0	85.0	95.0	99.0
thickness $(\%c)$	12.04	2.6	1.2	0.2

For Problems B and C, the geometry is parameterized with a larger number of B-spline control points resulting in a greater number of design variables, as summarized in Table 2. All other parameters are identical to Problem A.

6.2. *Problem D: multipoint example*

The objective is to minimize the drag coefficient at a fixed lift coefficient over a range of Mach numbers from 0.68 to 0.76. The Reynolds number is 2.7 million. By lifting the lift constraint into the objective function as a quadratic penalty term, the following objective function is obtained:

$$J = \begin{cases} \omega_{\mathrm{l}} \left(1 - \frac{C_{\mathrm{l}}}{C_{\mathrm{l}}^*}\right)^2 + \omega_{\mathrm{d}} \left(1 - \frac{C_{\mathrm{d}}}{C_{\mathrm{d}}^*}\right)^2 + \omega_{\mathrm{T}} T & \text{if } C_{\mathrm{d}} > C_{\mathrm{d}}^* \\ \omega_{\mathrm{l}} \left(1 - \frac{C_{\mathrm{l}}}{C_{\mathrm{l}}^*}\right)^2 + \omega_{\mathrm{T}} T & \text{otherwise} \end{cases} \quad [2]$$

The target lift coefficient, C_{l}^*, is 0.733; the target drag coefficient, C_{d}^*, which is assumed to be unattainable within the specified constraints, is 0.013. The weights are $\omega_{\mathrm{l}} = 1.0$, $\omega_{\mathrm{d}} = 0.1$, and $\omega_{\mathrm{T}} = 1.0$. The thickness constraints are given in Table 3. The initial shape, the geometry parameterization, the design variables, and the mesh are all the same as in Problem A.

A composite objective function is formed with four operating points distributed over the specified range of Mach numbers. The composite objective function is

$$J_{\mathrm{comp}} = \omega_1 J_{M=0.68} + \omega_2 J_{M=0.71} + \omega_3 J_{M=0.74} + \omega_4 J_{M=0.76} \quad [3]$$

where, for example, $J_{M=0.68}$ denotes the objective function given in (2) evaluated at a Mach number of 0.68. The four weights are $\omega_1 = 1/7$, $\omega_2 = 1/7$, $\omega_3 = 2/7$, and

Table 4. *Thickness constraints for optimization problem E*

constraint n°	1	2	3	4
location ($\%c$)	2.0	25.0	92.0	99.0
thickness ($\%c$)	4.0	11.8	0.9	0.2

$\omega_4 = 3/7$. Since it becomes more difficult to reduce drag as the Mach number is increased, the higher Mach numbers are given increased weight in order to achieve a reasonably constant drag coefficient over the specified range of Mach numbers.[2]

6.3. *Problem E: multiobjective example*

In this multiobjective optimization problem the two competing objectives are given by

$$J_{\rm l} = \left(1 - \frac{C_{\rm l}}{C_{\rm l}^*}\right)^2 + \omega_{\rm T} T \quad \text{and} \quad J_{\rm d} = \left(1 - \frac{C_{\rm d}}{C_{\rm d}^*}\right)^2 + \omega_{\rm T} T \qquad [4]$$

These two objectives are in competition because it is generally easier to produce low drag at a low lift coefficient value. The target lift coefficient, $C_{\rm l}^*$, is 0.55; the target drag coefficient, $C_{\rm d}^*$, which is again unattainable within the specified constraints, is 0.0095. The Mach number is 0.7, the Reynolds number, nine million. The weight on the thickness constraints, which are listed in Table 4, is $\omega_{\rm T} = 1.0$.

In the weighted-sum approach, the two objectives are combined as follows:

$$J = \omega_{\rm l} J_{\rm l} + (1 - \omega_{\rm l}) J_{\rm d} \qquad [5]$$

where the relative weight of the two objectives is controlled by $\omega_{\rm l}$. If $\omega_{\rm l} = 1$, then the solution is nonunique, since there are many airfoil shapes, within the specified constraints, that can produce a lift coefficient of 0.55. However, even a small weight on $J_{\rm d}$ eliminates this nonuniqueness. Therefore, one end of the Pareto front is defined by the limit as ω_1 approaches unity. The other end of the front is obtained when $\omega_1 = 0$, which leads to a pure drag minimization problem subject to the specified thickness constraints.

The initial airfoil and grid are as described in Problem A. The geometry parameterization is also identical, but two additional B-spline control points near the trailing edge are used as design variables, as shown in Figure 1. Thus there are eleven design variables, including the angle of attack. Note that, as a result of the need to minimize the strength of shock waves, aerodynamic optimization problems at transonic speeds

2. See Zingg *et al.* (2006) for an automated method for determination of the weights in multipoint optimization.

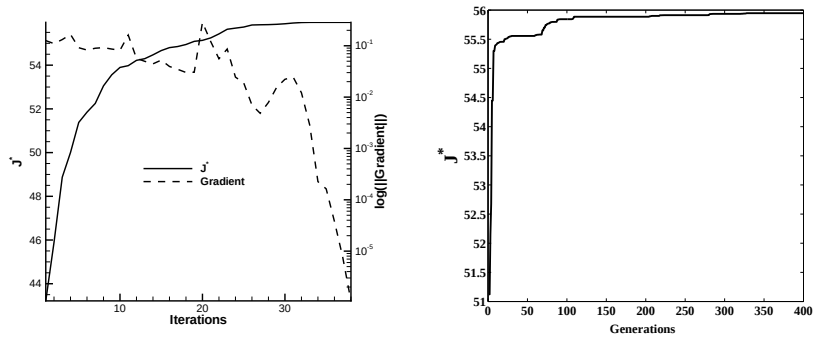
(*e.g.* Problems D and E) typically lead to smoother airfoils than problems at subsonic speeds (*e.g.* Problems A to C) and consequently require fewer geometric constraints.

7. Results and discussion

Our goal is to compare the efficiency of the two algorithms for the five optimization problems defined above. We measure cost in terms of the number of equivalent function evaluations (or flow solutions) needed to achieve a given level of convergence. In the genetic algorithm, the processing time needed to compute the required flow solutions dominates the overall processing time. The total number of flow solutions is roughly equal to the population size times the number of generations needed to reach a prescribed value of the objective function. It is actually less than this product because ten percent of the population survives from one generation to the next and hence must not be recomputed. Furthermore, a few members of a population at a given generation may be unacceptable (for example, the airfoil shape may be crossed over) and hence the flow is not computed for those airfoil shapes. In the gradient-based algorithm, the cost of computing the gradient is also significant, typically 20 to 50 percent of the cost of a flow solution. Consequently, when calculating the cost of the gradient-based algorithm we count a gradient computation as equivalent to a flow solution, which is typical of adjoint methods. Since each step of the gradient-based algorithm, whether calculating a new search direction or within a line search, involves one flow solution and one gradient computation, the total number of equivalent flow solutions is determined simply by multiplying the total number of steps by two.

Each algorithm has several parameters that can affect its efficiency. For example, within the gradient-based algorithm there must be a convergence criterion for both the flow solver and the gradient computation. The genetic algorithm also requires a convergence criterion for the flow solver. Another key parameter in the genetic algorithm is the population size. In our comparisons we use the default set of parameters for both algorithms. These parameter choices have been found to be efficient for the class of problems under study.

There are several strategies possible for reducing the cost of both algorithms. For example, one could consider beginning the optimization using a coarser grid, fewer design variables, or a simpler set of equations such as the Euler equations (as in variable fidelity optimization). Furthermore, both the genetic and the gradient-based algorithm can be parallelized. The former is particularly amenable to parallelization. These strategies are not considered in our current comparisons, primarily because in principle they can benefit both algorithms. Therefore, our present comparison of efficient implementations of the two algorithms in their basic form provides an important baseline comparison. Given the approximations made in defining the cost of the two algorithms and the variability associated with parameters and implementation, the resulting comparisons should not be considered precise. Rather they provide a reasonable estimate of the relative cost of the two algorithms to be expected when they are applied to optimization problems with characteristics similar to our five examples.



(a) Gradient-based algorithm (b) Genetic algorithm

Figure 2. *Convergence histories for problem A*

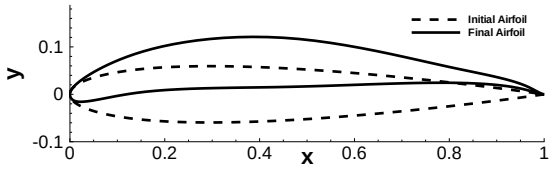


Figure 3. *Initial and final airfoils from the gradient-based algorithm, problem A*

7.1. *Problems A-C: lift-to-drag ratio maximization*

7.1.1. *Problem A: 9 design variables*

The convergence history of the gradient-based algorithm is plotted in Figure 2(a). The objective function plotted is $J^* = 1/J$, where J is defined in (1). Hence J^* is roughly equal to the lift-to-drag ratio, but slightly lower due to the contribution from the thickness constraints. J^* is increased from 43.22 to 55.96 after 38 iterations, or 76 equivalent function evaluations. The gradient has been reduced by roughly five orders of magnitude, indicating that the solution is well converged. The final airfoil shape is shown in Figure 3. The corresponding convergence history of the genetic algorithm is plotted in Figure 2(b) with respect to generations, where the population size is 32, *i.e.* there are 32 chromosomes per generation. The genetic algorithm has produced an airfoil with J^* equal to 55.95 after 400 generations (6317 function evaluations).

Table 5 compares the cost of the two algorithms for various levels of convergence, measured as a percentage of the change in J^* achieved by the fully-converged gradient-based solution. For example, 90% convergence corresponds to a J^* value of 54.68. For this degree of convergence, the genetic algorithm requires 7 times as

Table 5. *Cost to achieve various levels of convergence for problems A to C in terms of equivalent function evaluations. The column labelled % gives the percentage of the change in J^* relative to the fully converged solution. The column labelled ratio gives the cost of the genetic algorithm divided by the cost of the gradient-based algorithm*

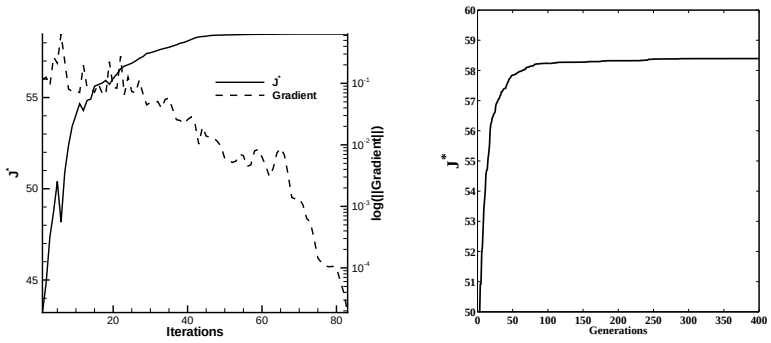
%	gradient-based algorithm			genetic algorithm			ratio		
	n° of variables			n° of variables			n° of variables		
	9	19	35	9	19	35	9	19	35
90	32	52	100	215	899	7,296	7	17	73
95	44	70	150	278	1,432	11,767	6	20	78
98	50	82	202	2,278	2,572	20,727	46	31	103
99	52	88	246	2,745	7,357	46,036	53	84	187

many equivalent function evaluations as the gradient-based algorithm. As the degree of convergence is increased, this ratio increases dramatically. For 99% convergence, the genetic algorithm is 53 times more expensive. This reflects the key shortcoming of genetic algorithms: although the general vicinity of the optimum is found reasonably quickly, convergence is slow near the optimum.

7.1.2. *Problem B: 19 design variables*

Figure 4(a) displays the convergence history of the gradient-based algorithm. The gradient is again reduced several orders of magnitude. After 83 iterations, or 166 equivalent function evaluations, J^* is increased to 58.48. As shown in Figure 5, the additional design variables have been exploited to produce two concave regions on the lower surface fore and aft of the primary thickness constraint at 35% chord. The resulting airfoil shape reinforces the fact that this is not a practical example from either a structural or off-design performance perspective, but it does provide a good test for an optimization algorithm. The shape of the lower surface could be improved through an alternative geometric constraint, such as a cross-sectional area constraint. The corresponding convergence history of the genetic algorithm is plotted in Figure 4(b) with respect to generations, where the population size is again 32. The genetic algorithm has produced an airfoil with $J^* = 58.41$ after 800 generations (25,319 function evaluations).

Table 5 again compares the cost of the two algorithms for various levels of convergence. The gradient-based algorithm requires on the order of twice as many function evaluations as for Problem A. Given that this case has roughly twice as many design variables, this is consistent with a linear dependence on the number of design variables. For 90% convergence, the genetic algorithm requires 17 times as many equivalent function evaluations as the gradient-based algorithm. For 99% convergence, the ratio is 84. The genetic algorithm can converge quite quickly, requiring hundreds of



(a) Gradient-based algorithm (b) Genetic algorithm

Figure 4. *Convergence histories for problem B*

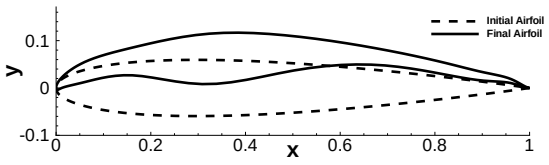
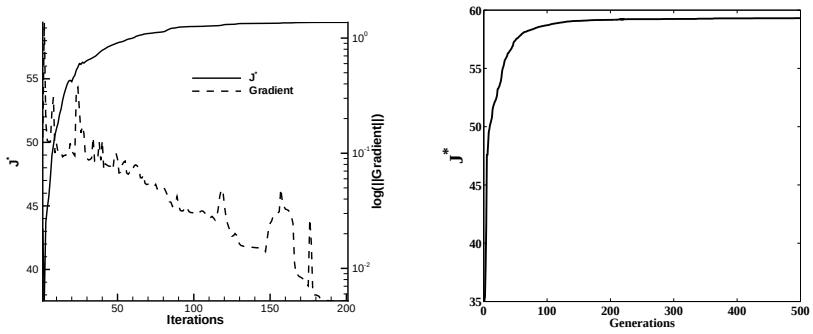


Figure 5. *Initial and final airfoils from the gradient-based algorithm, problem B*

function evaluations, when the number of design variables is small and the convergence tolerance is not too stringent. However, if the number of design variables is increased or the convergence tolerance is more stringent, then several thousand function evaluations are needed.

7.1.3. *Problem C: 35 design variables*

Results for Problem C are shown in Figures 6 and 7, and Table 5. The final J^* value achieved by the gradient-based algorithm is 59.44, about 2% greater than that achieved with 19 design variables. The line search stalls when the gradient is reduced only to roughly 0.005. Further study is needed to determine whether this is associated with increased stiffness caused by the increase in the number of design variables. The gradient-based convergence results are again consistent with a roughly linear dependence on the number of design variables. A population of 128 chromosomes was used with the genetic algorithm for this case, although a population size of 32 converged only slightly more slowly in terms of function evaluations. After 500 generations (63,216 function evaluations), the genetic algorithm produced an airfoil shape with $J^* = 59.31$, very close to the converged value from the gradient-based al-



(a) Gradient-based algorithm (b) Genetic algorithm

Figure 6. *Convergence histories for problem C*

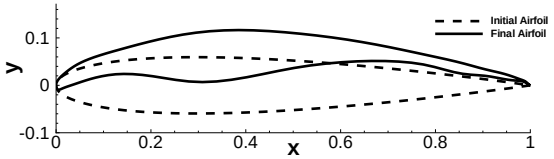
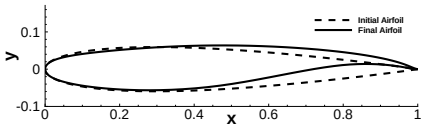


Figure 7. *Initial and final airfoils from the gradient-based algorithm, problem C*

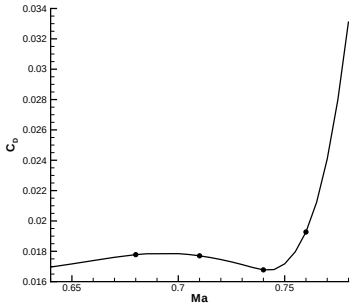
gorithm. The cost of the genetic algorithm increases more rapidly with the number of design variables than the roughly linear dependence displayed by the gradient-based algorithm. As a result, the genetic algorithm requires 73 times as many function evaluations for 90% convergence and 187 times as many for 99% convergence. The relative performance of the two algorithms reinforces the observations made for Problems A and B.

7.2. *Problem D: multipoint example*

In this example, the gradient-based algorithm reduces the objective function from 0.314 to 0.0123 in 104 iterations. Since each iteration involves four flow solutions, one at each Mach number, this corresponds to 416 flow solutions or 832 equivalent function evaluations (including the cost of the gradient computation). The gradient is reduced by over five orders of magnitude, indicating that an optimum has been found. Figure 8 shows the initial and final airfoils as well as the drag coefficient versus Mach number at $C_l = 0.733$. Note that during the optimization the target lift coefficient is not precisely achieved, as a result of the penalty approach to the imposition of the lift



(a) Initial and final airfoils from the gradient-based algorithm



(b) Drag coefficient vs. Mach number for $C_l = 0.733$

Figure 8. *Results for problem D*

constraint.[3] However, for the data plotted in Figure 8 the angles of incidence have been adjusted such that the lift coefficient is exactly 0.733. The drag coefficient is reasonably constant over the range of Mach numbers used for the optimization. After 18 iterations (144 function evaluations), the objective function is reduced to 0.0268, and the drag coefficient at $M = 0.76$ is within five counts (0.0005) of the converged value.

The genetic algorithm reduces the objective function to 0.0125 after 800 generations (101,424 function evaluations). A similar value of the objective function is attained by the gradient-based algorithm after 60 iterations (480 function evaluations), or 211 times faster. The lift and drag coefficients produced by the two algorithms at the four operating points are almost identical. After 600 generations (76,044 function evaluations), the genetic algorithm reduces the objective function to 0.0129; the gradient-based algorithm achieves this value 179 times faster. The genetic algorithm requires 3,520 function evaluations to reduce the objective function to 0.267, which is 24 times slower than the gradient-based algorithm.

When a multipoint optimization problem is solved with a fixed set of weights, as we have done here, it is equivalent to obtaining one point on a Pareto front[4] using the weighted-sum approach. Using the genetic algorithm, one can also treat this as a four-objective problem and apply the dominance Pareto front technique. Figure 9 shows the results from the archival file. Since a four-dimensional front cannot be easily plotted, the individual fitness values for each individual Mach number are shown. The chromosomes are ordered in terms of the fitness function computed at the lowest Mach number. Of course, it is expensive to generate such data, but they can be helpful in understanding and subsequently modifying the optimization problem to achieve

3. Zingg *et al.* (2007) present an alternative approach in which the lift constraint is strictly enforced.
4. In this case, the front is four-dimensional.

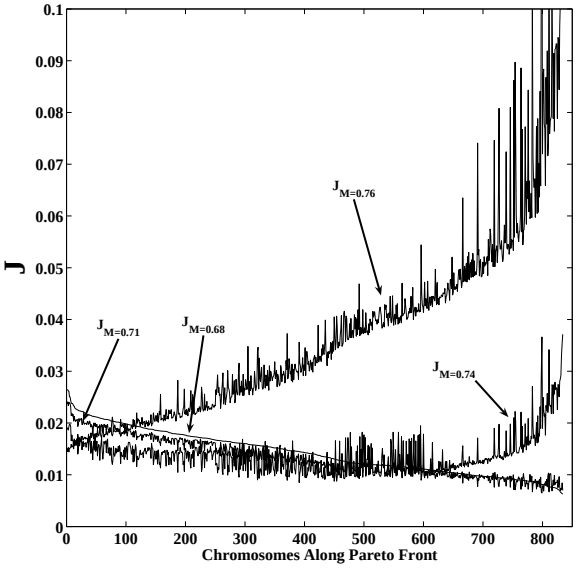


Figure 9. *Results of dominance Pareto front approach, problem D*

a specific goal. The fitness functions corresponding to Mach numbers of 0.68 and 0.71 behave similarly. The fitness at the highest Mach number is almost completely opposed, *i.e.* an airfoil shape that produces a low fitness function value at low Mach number produces a high value at high Mach number and vice-versa. The third objective function lies in between. With the weighted objective function, the optimum is found for a specific set of weights. With the data shown in the figure, the optimum can be found for any combination of weights without further optimization. Therefore, it can be advantageous to take a multiobjective approach to a multipoint problem. The cost of producing the data plotted in Figure 9 using the genetic algorithm is 38,912 function evaluations. With the equivalent expense, approximately 48 different sets of weights can be evaluated using the gradient-based algorithm.

7.3. *Problem E: multiobjective example*

Here we compare the cost of producing a two-dimensional Pareto front by the gradient-based technique using the weighted-sum approach and the genetic algorithm using both the weighted-sum and the dominance Pareto front approaches. For the weighted-sum approach, we use fifteen different values of ω_1 ranging from 0.01 to 0.99, as shown in Tables 6 and 7.

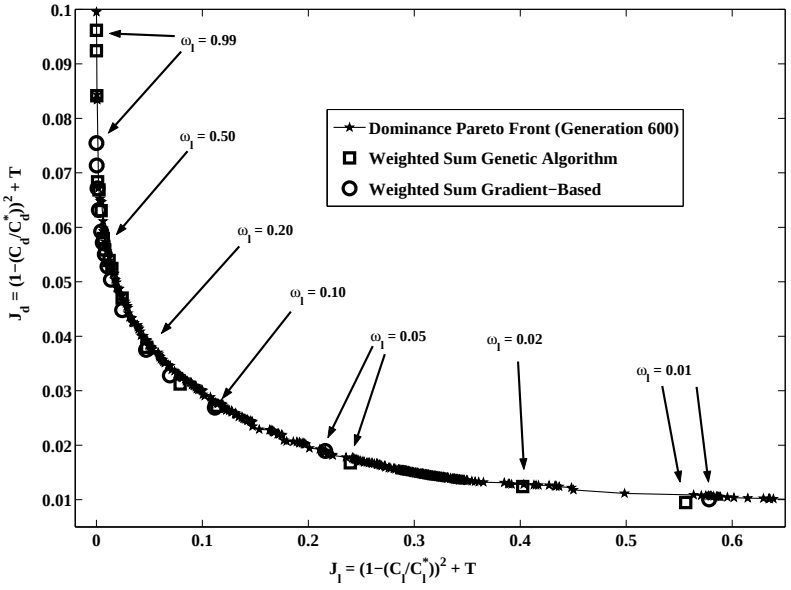


Figure 10. *Pareto front calculation*

The fronts generated are plotted in Figure 10. The drag minimization objective, J_{d}, is plotted on the y-axis, the lift objective, J_{l}, on the x-axis. The front generated using the gradient-based algorithm can be considered the true Pareto front, since there is a substantial reduction in the gradient for each point. Furthermore, the front is convex, so the weighted-sum approach is effective. With $\omega_{\mathrm{l}} = 0.99$, the lift objective is close to zero, since the target lift coefficient is achievable. The drag objective does not approach zero, even with $\omega_{\mathrm{l}} = 0.01$, since the target drag is unattainable. The three methods produce the same front with small differences related to convergence. This indicates that the property of the gradient-based algorithm of finding a local minimum is not an issue for this optimization problem.

The trade-off between lift and drag is shown more clearly by plotting the front in terms of the lift and drag coefficients rather than the objective functions, as in Figure 11.[5] With $\omega_{\mathrm{l}} = 0.99$, the gradient-based algorithm produces an airfoil with a lift coefficient of 0.549 and a drag coefficient of 0.0121. With $\omega_{\mathrm{l}} = 0.01$, the values are 0.132 and 0.0104. Tables 6 and 7 give detailed data for the airfoils generated by the two algorithms using the weighted-sum approach. Lack of convergence of the genetic algorithm is particularly noticeable in the angle of attack (α), but this design variable tends to vary in conjunction with the trailing edge angle, and the net effect on the objective function is small. The anomalous results showing the genetic algorithm

5. Note that this obscures the contribution of the thickness constraints.

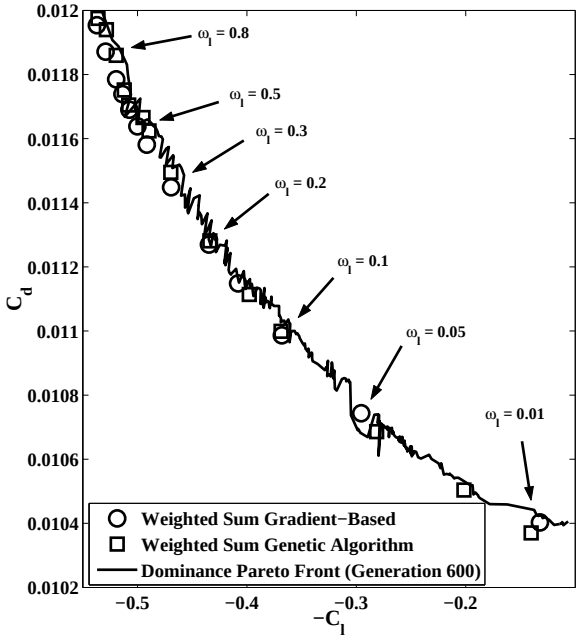


Figure 11. *Pareto fronts showing lift and drag coefficients*

result to be superior (*e.g.* $\omega_1 = 0.01$) are caused by incomplete flow solver convergence in the genetic algorithm result.

Figure 12 shows the intermediate results of the dominance Pareto front approach. After 600 generations, the front lies very close to that computed using the gradient-based algorithm. The costs are as follows:

– *gradient-based algorithm:* the number of iterations needed to compute well converged solutions for each of the fifteen points with the gradient-based algorithm ranges from 49 to 151. A total of 1,219 iterations are required; hence the cost of computing the Pareto front shown in Figure 10 is 2,438 equivalent function evaluations;

– *genetic algorithm, weighted-sum approach:* well converged solutions for each point are obtained using the genetic algorithm with roughly 4,700 function evaluations. This is consistent with the cost of a convergence level between 98 and 99% for Problem A, which has a similar number of design variables. The total cost of computing the front using the weighted-sum approach is thus roughly 70,000 function evaluations, which is 28 times the cost of the gradient-based approach;

– *genetic algorithm, dominance Pareto front approach:* as shown in Figure 12, the front produced using the dominance Pareto front approach after only five generations

Table 6. *Gradient-based algorithm results, problem E*

ω_l	C_l	C_d	α	J_l	J_d
0.99	0.5494	0.01211	0.286	0.0000	0.0754
0.90	0.5439	0.01203	0.264	0.0002	0.0713
0.80	0.5371	0.01195	0.243	0.0009	0.0671
0.70	0.5291	0.01187	0.221	0.0023	0.0631
0.60	0.5194	0.01178	0.193	0.0044	0.0592
0.55	0.5137	0.01174	0.180	0.0059	0.0571
0.50	0.5073	0.01169	0.166	0.0079	0.0550
0.45	0.5000	0.01164	0.152	0.0104	0.0527
0.40	0.4915	0.01158	0.135	0.0136	0.0503
0.30	0.4693	0.01145	0.090	0.0242	0.0447
0.20	0.4346	0.01127	0.023	0.0467	0.0375
0.15	0.4080	0.01115	-0.007	0.0693	0.0328
0.10	0.3680	0.01099	-0.055	0.1117	0.0268
0.05	0.2955	0.01074	-0.140	0.2159	0.0189
0.01	0.1321	0.01040	-0.261	0.5781	0.0100

Table 7. *Genetic algorithm results with weighted-sum approach, problem E*

ω_l	C_l	C_d	α	J_l	J_d
0.99	0.5491	0.01245	2.123	0.0000	0.0961
0.90	0.5437	0.01221	1.884	0.0004	0.0814
0.80	0.5361	0.01197	0.406	0.0011	0.0683
0.70	0.5284	0.01194	1.333	0.0024	0.0668
0.60	0.5194	0.01186	1.309	0.0044	0.0630
0.55	0.5120	0.01175	0.712	0.0064	0.0579
0.50	0.5079	0.01170	0.434	0.0078	0.0558
0.45	0.4949	0.01167	1.000	0.0119	0.0539
0.40	0.4894	0.01162	1.063	0.0145	0.0524
0.30	0.4695	0.01149	0.984	0.0242	0.0470
0.20	0.4338	0.01128	0.420	0.0474	0.0379
0.15	0.3978	0.01111	0.192	0.0789	0.0312
0.10	0.3687	0.01010	-0.003	0.1109	0.0272
0.05	0.2814	0.01069	-0.115	0.2395	0.0167
0.01	0.1402	0.01037	-0.275	0.5561	0.0094

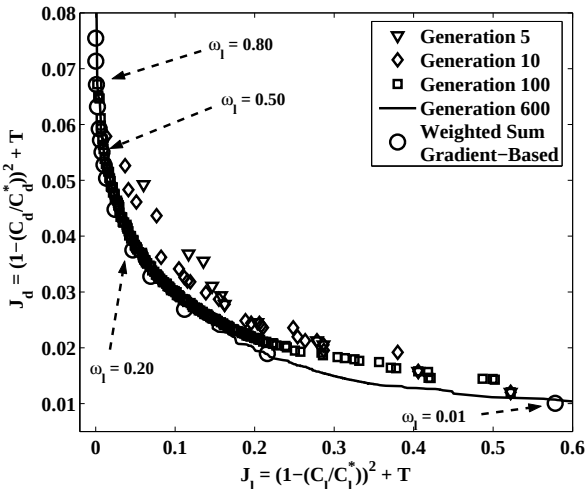


Figure 12. *Pareto front comparison*

(105 function evaluations) already displays the basic trade-off between the two competing objectives. Note that for this optimization forty percent of the population is passed through without alteration from one generation to the next, so the number of function evaluations is substantially less than the product of the number of generations and the population size. After ten generations (205 function evaluations), the front is marginally improved. After 100 generations, the front is well converged except for the region corresponding to small values of ω_1. For 100 generations, 1,999 function evaluations are needed, which is similar to the expense of the gradient-based method to achieve a well converged front. Finally, 11,937 function evaluations are needed to compute 600 generations. The cost using the genetic algorithm with the dominance Pareto front approach is thus on the order of five times that of the gradient-based algorithm.

8. Summary and conclusions

A comparative evaluation of a genetic algorithm and a gradient-based algorithm using the discrete-adjoint approach has been presented in the context of a typical PDE-constrained optimization problem, namely aerodynamic shape optimization. Single-point optimization problems with varying numbers of design variables, a multipoint problem, and a multiobjective problem have been examined. Aspects of the computations other than the optimization algorithms under study, including the geometry parameterization and the flow solution are identical, thus permitting an accurate comparison between the two algorithms. The objective is to determine a rough estimate of the relative computing expense of the two algorithms for a representative optimiza-

tion problem. Given that genetic and gradient-based algorithms have various strengths and weaknesses, as discussed earlier, their relative cost is a key factor in choosing an algorithm for a specific class of optimization problems.

The algorithms' cost is measured in terms of the number of flow solutions required to obtain specified degrees of convergence. There are several parameters affecting cost and some approximations made in calculating it, so these numbers should be considered only as a rough indication of the relative cost. For the single-point optimization problems, the genetic algorithm varies from 6 to 187 times more expensive than the gradient-based algorithm, depending on the number of design variables and the degree of convergence. For the multipoint problem, the genetic algorithm requires roughly 24 to 200 times as many function evaluations. For the multiobjective problem, the Pareto front can be computed using the weighted-sum approach with either algorithm. Alternatively, with the genetic algorithm the dominance Pareto front approach can be used. With the dominance Pareto front approach the genetic algorithm produces a well converged front three to six times more rapidly than with the weighted-sum approach, but five times slower than the gradient-based algorithm.

Based on our data the following conclusions can be drawn:

– the gradient-based algorithm using the adjoint approach scales roughly linearly with the number of design variables, while the genetic algorithm's cost increases more rapidly as the number of design variables is increased;

– the cost of the genetic algorithm relative to the gradient-based algorithm increases dramatically with tighter convergence requirements;

– the two algorithms produce essentially identical Pareto fronts;

– using the genetic algorithm, the dominance approach produces a Pareto front more efficiently than the weighted-sum approach;

– using the weighted-sum approach, the gradient-based algorithm can compute a Pareto front on the order of five times more rapidly than the genetic algorithm with the dominance approach.

Although relative cost is only one of several important considerations in choosing an optimization algorithm, our results suggest the following important overall conclusion. The genetic algorithm may be more suited to *preliminary design* where low-fidelity models are typically used (hence reduced function evaluation cost), lower convergence tolerances are required, and understanding trade-offs is extremely important. The gradient-based algorithm may be more appropriate for *detailed design*, where high-fidelity simulations are used, tighter convergence tolerances are needed, and although a good understanding of trade-offs remains important in posing the optimization problem, the range of possible designs under consideration is much narrower.

9. References

Anderson W., Newman J., Whitfield D., Nielsen E., " Sensitivity analysis for the Navier-Stokes equations on unstructured meshes using complex variables", *AIAA J.*, vol. 39, n° 1, p. 56-63, 2001.

Coello Coello C. A., " Evolutionary Multi-Objective Optimization: A Critical Review", *Evolutionary Optimization*, Kluwer Academic, New York, p. 117-146, 2002.

Deb K., *Multi-Objective Optimization Using Evolutionary Algorithms*, Chichester, New York, 2002.

Dennis J., Schnabel R., *Numerical Methods for Unconstrained Optimization*, Prentice-Hall, Englewood Cliffs, N.J., 1983.

Elliott J., Peraire J., " Practical 3D aerodynamic design and optimization using unstructured meshes", *AIAA J.*, vol. 35, p. 1479-1485, 1997.

Giannakoglou K., " Design of optimal aerodynamic shapes using stochastic optimization methods and computational intelligence", *Progress in Aerospace Sciences*, vol. 38, p. 43-76, 2002.

Gill P., Murray W., Saunders M., "SNOPT: An SQP algorithm for large-scale constrained optimization", *SIAM J. Optim.*, vol. 12, p. 979-1006, 2002.

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

Hicks R., Henne P., " Wing design by numerical optimization", *J. of Aircraft*, vol. 3, p. 407-412, 1978.

Hicks R., Murman E., Vanderplaats G., An assessment of airfoil design by numerical optimization, TM n° X-3092, NASA, July, 1974.

Holst T. L., Pulliam T. H., Aerodynamic Shape Optimization Using A Real-Number-Encoded Genetic Algorithm, Technical Report n° 2001-2473, AIAA, June, 2001.

Jameson A., " Aerodynamic design *via* control theory", *J. of Scientific Computing*, vol. 3, n° 3, p. 233-260, 1988.

Marco N., Désidéri J.-A., Lantéri S., Multi-objective optimization in CFD by genetic algorithms, Technical report n° 3686, Institut National de Recherche en Informatique et en Automatique (INRIA), France, April, 1999.

Nemec M., Optimal Shape Design of Aerodynamic Configurations: A Newton-Krylov Approach, Ph.d. Thesis, University of Toronto, Canada, 2002.

Nemec M., Zingg D., " Newton-Krylov algorithm for aerodynamic design using the Navier-Stokes equations", *AIAA J.*, vol. 40, p. 1146-1154, 2002.

Nemec M., Zingg D., Pulliam T., " Multipoint and Multi-Objective Aerodynamic Shape Optimization", *AIAA J.*, vol. 42, p. 1057-1065, 2004.

Nielsen E., Anderson W., " Aerodynamic design optimization on unstructured meshes using the Navier-Stokes equations", *AIAA J.*, vol. 37, p. 1411-1419, 1999.

Nocedal J., Wright S., *Numerical Optimization*, Springer-Verlag, New York, 1999.

Obayashi S., " Aerodynamic optimization with evolutionary algorithms", *Inverse Design and Optimization Methods, Lecture Series 1997-05*, von Karman Institute for Fluid Dynamics, Brussels, Belgium, 1997.

Oyama A., Wing design using evolutionary algorithms, Ph.d. Thesis, Tohoku University, Japan, 2000.

Pironneau O., " On optimum design in fluid mechanics", *J. Fluid Mech.*, vol. 64, p. 97-110, 1974.

Poloni C., Giurgevich A., Onesti L., Pediroda V., " Hybridization of a Multi-Objective Genetic Algorithm, a Neural Network and a Classical Optimizer for Complex Design in Fluid Dynamics", *Comput. Methods Appl. Mech. Eng.*, vol. 18, p. 403-420, 2000.

Pueyo A., Zingg D., " Efficient Newton-Krylov Solver for Aerodynamic Computations", *AIAA J.*, vol. 36, p. 1991-1997, 1998.

Pulliam T., " Efficient Solution Methods for The Navier-Stokes Equations", *Numerical Techniques for Viscous Flow Computation In Turbomachinery Bladings*, von Karman Institute for Fluid Dynamics, Brussels, Belgium, 1985.

Reuther J., Jameson A., Alonso J., Rimlinger M., Saunders D., " Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, part 1", *J. of Aircraft*, vol. 36, p. 51-60, 1999.

Vicini A., Quagliarella D., " Inverse and direct airfoil design using a multiobjective genetic algorithm", *AIAA J.*, vol. 35, p. 1499-1505, 1997.

Vicini A., Quagliarella D., " Airfoil and Wing Design through Hybrid Optimization Strategies", *AIAA J.*, vol. 37, p. 634-642, 1999.

Zingg D., Billing L., Toward Practical Aerodynamic Design Through Numerical Optimization, Technical report n° 2007-3950, AIAA, June, 2007.

Zingg D., Elias S., On Aerodynamic Optimization Under a Range of Operating Conditions, Technical Report n° 2006-1053, AIAA, Jan., 2006.