Comparison of Optimization Algorithms Applied to Aerodynamic Design

Po-Yang Jay Liu * and D.W. Zingg †
Institute for Aerospace Studies
University of Toronto
4925 Dufferin St., Toronto
M3H 5T6, Canada

This paper investigates the performance of two gradient-based optimizers, a quasi-Newton algorithm that uses the quadratic penalty approach to handle constraints, and a sequential quadratic programming (SQP) algorithm that uses an augmented Lagrangian merit function (SNOPT). They are applied to the two-dimensional Navier-Stokes aerodynamic shape optimization problem. The aerodynamic analysis is performed using a Newton-Krylov algorithm consisting of an inexact-Newton method and a preconditioned Krylov solver. The gradient is computed through the discrete adjoint technique, and the discrete adjoint problem is solved with a preconditioned Krylov algorithm. The performance of the optimizers is demonstrated for several design examples, including inverse design, maximization of lift-to-drag ratio, maximization of endurance factor, lift-constrained drag minimization, and multi-point optimization. The SQP optimizer with an augmented merit function is more robust and efficient than the quasi-Newton optimizer with quadratic penalty terms for the majority of the test cases presented.

Introduction

Engineering design problems often involve a large number of design parameters and a complex analysis problem. For example, the aerodynamic design of a wing requires a geometry parameterization with enough variables to define a suitably flexible design space and a means for determining the performance of a given wing. Numerical solution techniques for the Reynolds-averaged Navier-Stokes equations have greatly reduced the reliance on wind-tunnel testing of prospective configurations. With a sufficiently accurate numerical analysis technique, a numerical optimization algorithm can be used to determine the aerodynamic shape which minimizes a specified objective while satisfying certain constraints. This powerful approach frees the designer from the tedious cut-and-try process and permits more emphasis to be placed on careful selection of objectives and constraints.

There are many different techniques available for aerodynamic optimization, and a clear consensus has yet to emerge. The two most popular current techniques are evolutionary or genetic algorithms[1] and gradient-based algorithms based on either the discrete or continuous adjoint approaches.[2,3] Gradient-based algorithms are typically much faster, while genetic algorithms can be more robust and more generally applicable. Pulliam *et al.*[4] provided a detailed discussion of the trade-offs and an example of a Pareto front computation in which a gradient-based algorithm converged about thirty times faster than a genetic algorithm. Other studies characterizing genetic and gradient-based algorithms were performed by Weinerfelt[5] and Obayashi.[6] The challenge is to improve the speed of genetic algorithms and the robustness of gradient-based algorithms, or perhaps to combine the two in a hybrid algorithm with the best characteristics of both.

Given efficient algorithms for solving the analysis problem and calculating the gradient, the overall efficiency of a gradient-based algorithm is determined by the effectiveness of the optimizer in determining the local optimum while satisfying the constraints. If the gradient can be calculated accurately, quasi-Newton methods are generally preferred for unconstrained optimization problems. Such methods apply an updating formula, such as the Davidon-Fletcher-Powell (DFP) and Broyden-Fanno-Goldfarb-Shannon (BFGS) formulas, to provide an approximation to the Hessian or its inverse which can become exact under certain conditions.[7,8] These formulas are typically used in conjunction with an inexact line search. Jameson and Vassberg[9] tested several such methods in the context of a model optimization problem. Quasi-Newton algorithms can be applied to constrained problems by incorporating the constraints into the objective function through a quadratic penalty function[10,11] or the Kreisselmeier–Steinhauser (KS) function.[12–15]

Alternatively, constrained optimization problems can be solved using sequential quadratic programming (SQP) methods.[16–18] For example, SNOPT[18] (which stands for sparse nonlinear optimizer) uses an

[*] M.A.Sc. Graduate, poyang.liu@utoronto.ca.

[†] Professor, Senior Canada Research Chair in Computational Aerodynamics, Senior AIAA Member, dwz@oddjob.utias.utoronto.ca.

augmented Lagrangian merit function and a limited-memory quasi-Newton approximation to the Hessian of the Lagrangian. SQP algorithms were used by Willcox and Wakayama[19] to design wings for multiple aircraft, by Soemarwoto and Labrujère[20] to reduce pressure drag over a range of design Mach numbers, by Li *et al.*[21] to solve multi-point lift-constrained drag minimization problems, and by Melvin *et al.*[22] for multi-point inverse design.

In this paper, we compare two optimizers, SNOPT and a quasi-Newton method based on the BFGS update formula with an inexact line search and a quadratic penalty formulation for the constraints.[11] The two optimizers are applied to the design of airfoils with the flow governed by the Reynolds-averaged Navier-Stokes equations using the Newton-Krylov algorithm of Nemec and Zingg[11] in which the gradient is calculated through the discrete adjoint approach. This algorithm has been successfully applied to numerous optimization problems, including high-lift, multi-point, and multi-objective problems.[23,24] The objective here is to determine whether the SNOPT algorithm is more effective than the quasi-Newton algorithm used by Nemec and Zingg. A key distinction in the two algorithms lies in the treatment of constraints. SNOPT is designed for constrained optimization, while the quasi-Newton method incorporates the constraints through a penalty formulation.

Problem Formulation

The aerodynamic shape optimization problem consists of determining values of design variables x such that the specified objective function $\mathcal{J}$ is minimized subject to constraint equations C_j:

$$\begin{aligned} \min_{x} \quad & \mathcal{J}\left[x, Q(x)\right] \\ \text{s.t.} \quad & C_j\left[x, Q(x)\right] \le 0 \quad j = 1, \ldots, N_{\mathrm{C}} \end{aligned} \tag{1}$$

where Q, the flow variables, satisfy the governing flow equations, and N_{C} is the number of constraints. In this work, the constraint equations represent airfoil thickness constraints that are a function of the design variables only, that is $C_j(x) \le 0$.

Design Variables

The geometry of the airfoil is described with fourth-order B-spline curves. The coordinates of the B-spline control-points are used as design variables. We only allow displacements in the vertical direction for the B-spline control points. The control points associated with the leading and trailing edges remain fixed in order to provide a round leading edge and a sharp trailing edge. The design variables may also include the angle-of-attack.

By increasing the number of control points, the flexibility of the B-spline curve is improved. Figs. 1 to 3 show the locations of control points that are used to approximate the RAE 2822 airfoil. The grey control points are used as design variables for optimization.

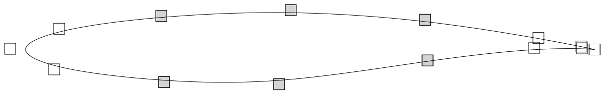

Fig. 1 RAE 2822 airfoil with 6 out of 15 control control points as design variables

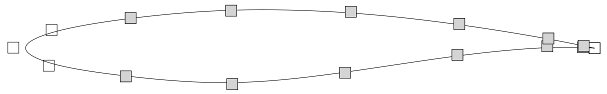

Fig. 2 RAE 2822 airfoil with 12 out of 17 control control points as design variables

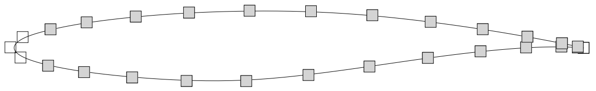

Fig. 3 RAE 2822 airfoil with 24 out of 29 control control points as design variables

Scaling Methods

For most of the test cases, the angle-of-attack is one of the design variables. It is of a different order of magnitude than the control-point design variables. One way to ensure that the problem is well-scaled is to devise a linear transformation that brings the diagonal elements of the Hessian of the transformed problem as closed to unity as possible. The purpose of scaling is to make all the variables of a similar order of magnitude with the aim of causing each variable to be of similar weight during the optimization. Gill *et al.*[16] provided a good discussion on the pros and cons of scaling. We consider a simple linear transformation of the variables with the form

$$x = Dy \tag{2}$$

where x is the vector of design variables, y contains the transformed variables, and D is a constant diagonal matrix. The design variables are scaled before an optimization iteration starts and transformed back to the original scaling before a flow solve.

A natural approach to scaling is to choose d_i, the i-th diagonal entry in D, as the i-th design variable at the first iteration of the optimization:

$$\text{scaling option 1}: \quad d_i = \begin{cases} x_i & \text{control point design variables} \\ AOA & \text{angle-of-attack design variable} \end{cases} \tag{3}$$

However, this scaling method depends on the initial control point values. It gives more weight to the control points closer to midchord with larger magnitude. It also gives more weight to the cases where the angle-of-attack is larger. An alternative scaling strategy is:

$$\text{scaling option 2}: \quad d_i = \begin{cases} 0.03 & \text{control point design variables} \\ 1.0 & \text{angle-of-attack design variable} \end{cases} \tag{4}$$

The scaling factor 0.03 is chosen based on the initial design variables for our test cases, which vary from 0.002 to 0.7.

Thickness Constraints

The airfoil thickness constraints are cast as inequality constraints given by the following term

$$C_j = \begin{cases} \left(1 - \dfrac{t_j(x)}{t_j^*}\right)^2 & \text{if } t_j(x) < t_j^* \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where $t^*(x)$ is the minimum allowable airfoil thickness at some location x. If the airfoil thickness is below the target value specified at the given chordwise location, then a penalty is imposed on the objective function. If the thickness to chord ratio is equal to or greater than that specified, then the objective function is unaffected, and the constraint is described as inactive. Some thickness constraints are imposed to prevent crossover during the optimization. These are typically inactive at convergence.

Numerical Method

Flow Solution

The spatial discretization of the flow equations is the same as that used in ARC2D.[25] The discretization consists of second-order centered-difference operators with second- and fourth-difference scalar artificial dissipation. The Spalart-Allmaras turbulence model is discretized as described in Ref. 26. Overall, the spatial discretization leads to a nonlinear system of equations

$$R(\widehat{Q}, x) = 0 \tag{6}$$

where $\widehat{Q}_i = J^{-1}Q_i = J^{-1}[\rho_i, (\rho u)_i, (\rho v)_i, e_i, \tilde{\nu}_i]^T$ is the vector of conservative dependent state variables, and J^{-1} denotes the grid-metric Jacobian. Note that structured C-topology grids are used.

Eq. 6 is solved in a fully-coupled manner, where convergence to steady state is achieved using the preconditioned GMRES algorithm in conjunction with an inexact-Newton strategy based on Pueyo and Zingg.[27] Details of the algorithm are provided in Refs. 11 and 28.

Gradient Evaluation

The adjoint method for calculating the gradient of the objective function has been applied successfully by Jameson *et al.*,[29] Nemec and Zingg,[11] and several others. It is extremely efficient, since the computational expense of the gradient evaluation is effectively independent of the number of design variables.

Using the discrete adjoint method, the gradient of the objective function is given by

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}x} = \frac{\partial \mathcal{J}}{\partial x} - \psi^{\mathrm{T}} \frac{\partial R}{\partial x} \tag{7}$$

where we reduce the vector of design variables x to a scalar in order to clearly distinguish between partial and total derivatives. The vector ψ represents the adjoint variables, which are determined from the adjoint equation:

$$\frac{\partial R}{\partial Q}^{\mathrm{T}} \psi = \frac{\partial \mathcal{J}}{\partial Q}^{\mathrm{T}} \tag{8}$$

We adopt the preconditioned GMRES strategy from the flow solver to solve the adjoint equation.[28] With this approach, the computing time required for the gradient evaluation is typically about one-fifth of that required for a flow solve.[11]

Optimizers

BFGS Algorithm with Penalty Terms

We use the BFGS quasi-Newton algorithm[7] to solve the unconstrained optimization problem. A quadratic penalty method can be used to cast a constrained problem as an unconstrained problem by incorporating the constraints in a composite objective function:

$$\mathcal{J} = \mathcal{J}_o + \mathcal{J}_{\mathrm{T}} \tag{9}$$

where $\mathcal{J}_o$ denotes the original objective function. $\mathcal{J}_{\mathrm{T}}$ is the quadratic penalty term defined as:

$$\mathcal{J}_{\mathrm{T}} = \omega_{\mathrm{T}} \sum_{j=1}^{N_{\mathrm{T}}} C_j \tag{10}$$

where N_{T} is the number of thickness constraints, ω_{T} is a user-specified constant, and the C_j are the constraints defined as in Eq. 5. An unconstrained optimization algorithm is then used to find the minimum of the composite objective function.

A quadratic model of the composite objective function is formed at each iterate x_k:

$$m_k(p) = \mathcal{J}_k + \nabla \mathcal{J}_k^T p + \frac{1}{2} p^T B_k p \tag{11}$$

where B_k is a symmetric positive definite matrix which holds the curvature information. For Newton's method, B_k is the Hessian matrix. However, the Hessian matrix is often difficult, if not impossible, to obtain. The quasi-Newton methods are based on the idea of building up curvature information as the iterations proceed, using the observed behaviour of the objective function and its gradient. The approximation to the curvature of a nonlinear function can be computed without explicitly forming the Hessian matrix. For the quasi-Newton method, B_k is an approximation of the Hessian matrix that is updated at every iteration.

The minimizer p_k of this model, which we can write explicitly as

$$p_k = -B^{-1}\nabla\mathcal{J}_k \tag{12}$$

is used as the search direction, and the new iterate is

$$x_{k+1} = x_k + \alpha_k p_k \tag{13}$$

where α_k is the step length determined by a line search. At each step of the line search, the objective function value and the gradient value are used to construct a local cubic interpolant.

SNOPT Algorithm

Transformation to an unconstrained problem by incorporating the constraints into the objective function can be an effective approach. However, with each additional constraint, the approximate Hessian matrix that holds the curvature information may become more ill conditioned near the minimizer.[8] This condition can make unconstrained optimization algorithms, such as the quasi-Newton method, perform poorly. The scaling of the problem may also be adversely affected if the constraints are not formulated carefully. One alternative optimization method is the SNOPT algorithm based on sequential quadratic programming which has been designed for problems with a large number of variables and constraints.

SNOPT generates a search direction from the solution of a QP subproblem which minimizes a quadratic model subject to linearized constraints. A smooth augmented Lagrangian merit function is used to ensure convergence from an arbitrary starting point.[18] The merit function plays the role of the objective function in unconstrained optimization; however, the constraints are treated separately. For a detailed description of the SNOPT algorithm, see Gill, Murray and Saunders.[18] For an overview of SQP methods, see, for example, Gill, Murray, and Wright,[16] Murray,[17] and Nocedal and Wright.[8]

The SNOPT algorithm is applicable to both constrained and unconstrained problems. In order to clearly delineate the effect of the approach to constraint handling, we also apply SNOPT to the unconstrained problem with the composite objective function incorporating the quadratic penalty terms.

Results and Discussion

We examine the performance of the following optimization algorithms:

- BFGS with the quadratic penalty approach (BFGS)
- SNOPT with the quadratic penalty approach (SNOPT-Q)
- SNOPT with constraints handled explicitly (SNOPT-A)

Inverse design, maximization of lift-to-drag ratio, maximization of endurance factor, lift-constrained drag-minimization, and multi-point optimization problems are considered. In all cases, the maximum number of objective function and gradient evaluations is restricted to 150. In some cases, the line search stalled before 150 iterations were completed. These cases can be restarted with the approximate Hessian inverse in the BFGS algorithm reset to the identity matrix. If the gradient was already reduced by several orders of magnitude, then no restart was performed. If the stalling occurred after at least 140 iterations, a restart was also not performed.

Inverse Design

For the inverse design problem, the objective function is given by

$$\mathcal{J} = \frac{1}{2}\sum_{j=1}^{N_\mathrm{A}}(C_{\mathrm{p}_j} - C^*_{\mathrm{p}_j})^2 \tag{14}$$

where C^*_p represents the target pressure distribution, and N_A denotes the number of nodes on the airfoil. For our tests we use a pressure distribution obtained from a B-spline representation of the RAE 2822 airfoil. Thus we are assured that there is a solution in the design space for which the objective function is equal to machine zero. Flow conditions are $M_\infty = 0.2$ and $Re = 2 \times 10^6$, with the angle-of-attack fixed throughout the optimization. The initial shape is a B-spline representation of the NACA 0012 airfoil. No thickness constraints are imposed (so there is no distinction between SNOPT-A and SNOPT-Q).

Figs. 4 and 5 show the convergence of the objective function with 6, 12, and 24 design variables (denoted by 6dv, 12dv, and 24dv, respectively). Initially the first scaling option (Eq. 3) is used for all test cases. Results for the second scaling technique are presented subsequently. The horizontal axis shows the number of objective function and gradient evaluations. All cases converge well with the two optimizers performing similarly. For both optimizers, the number of iterations needed for convergence increases as the number of degrees of freedom increases. This is typical of quasi-Newton methods. Fig. 6 shows the initial, final, and target pressure distributions using 24 out of 29 control points as design variables. Also shown is the SNOPT solution after 15 design iterations, which is partially converged.

Lift-to-Drag Maximization

For the maximization of the lift-to-drag ratio, the objective function is given by

$$\mathcal{J} = \frac{C_\mathrm{D}}{C_\mathrm{L}} \tag{15}$$

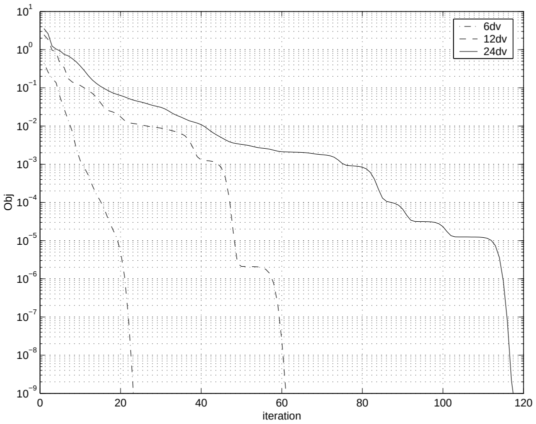

Fig. 4 Inverse design : BFGS

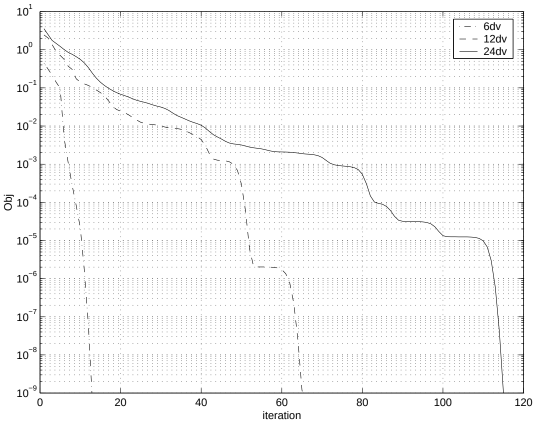

Fig. 5 Inverse design : SNOPT

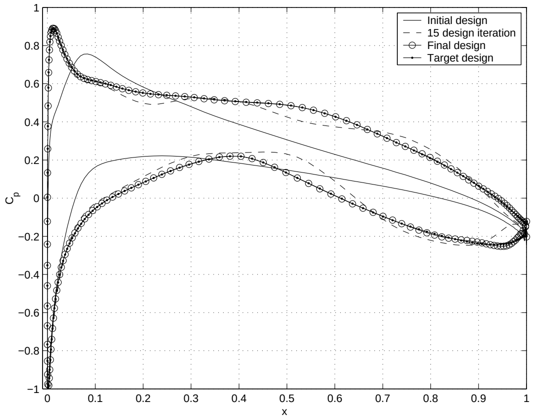

Fig. 6 C_p distribution comparison for the inverse design problem (for 24dv case)

Constraint #	1	2	3	4	5
x/c	15.0	35.0	60.0	92.0	99.0
t/c	1.0	16.4	7.0	1.0	0.1

Table 1 Thickness constraints for the maximization of lift-to-drag ratio and endurance factor

Flow conditions are again $M_\infty = 0.2$ and $Re = 2 \times 10^6$. The RAE 2822 airfoil is used as the starting airfoil, and the angle-of-attack is included as a design variable.[1] In order to prevent the lower and upper airfoil surfaces from crossing during the optimization process and to ensure that an airfoil of reasonable thickness is obtained, the thickness constraints given in Table 1 are imposed. The table shows a set of five thickness constraints and their chord locations.

Figs. 7 to 9 compare the results for lift-to-drag maximization. The lift-to-drag ratio is greatly increased, with the increased flexibility associated with a larger number of design variables leading to greater increases. A significant portion of the increases obtained is achieved within the first few iterations. Further iterations are needed in order to reduce the gradient sufficiently that one can be confident that a local minimum has been found. Comparing BFGS and SNOPT-Q, we see that the latter is somewhat faster. This shows that SNOPT offers an advantage even when constraints are imposed using the penalty approach. A much greater advantage is obtained using SNOPT-A. For this problem, the explicit handling of constraints leads to much faster convergence. SNOPT-A converges roughly twice as fast as SNOPT-Q for the 6 and 24 design variable cases. All of the methods converge to the same value of the lift-to-drag ratio and the design variables. The only exception is the 24 design variable case for which BFGS is not converged after 150 iterations.

Endurance Factor Maximization

The third design problem is the maximization of the endurance factor with the same flow conditions, initial airfoil, and thickness constraints as the previous problem. The objective function, which is the inverse of the endurance factor, is given by[30]

$$\mathcal{J} = \frac{C_{\mathrm{D}}}{C_{\mathrm{L}}^{3/2}} \qquad (16)$$

Figs. 10 to 12 show the results for this design problem. The convergence patterns are similar to the previous case. The endurance factor is greatly increased, more than tripled with 12 and 24 design variables, with a large fraction of the improvement occurring in the first ten iterations. The SNOPT-Q algorithm again converges more rapidly than the BFGS algorithm; the latter fails to converge within 150 iterations for the

[1] Note that there is thus actually one more design variable than indicated in the labelling in the figures. "6dv" means that there are six control-point design variables, seven in total.

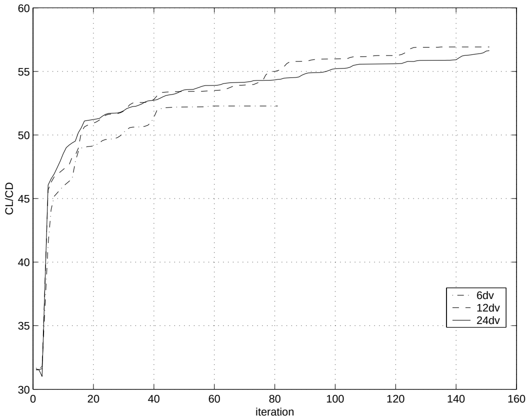

Fig. 7 Lift-to-drag ratio: BFGS

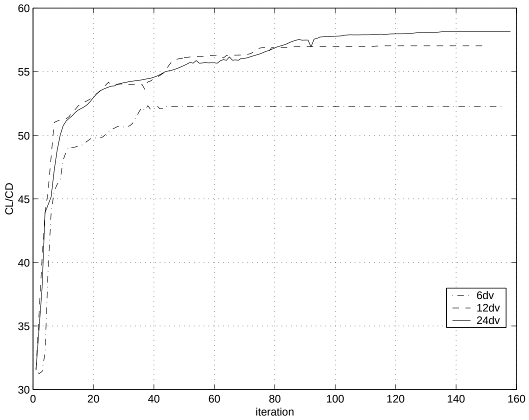

Fig. 8 Lift-to-drag ratio: SNOPT-Q

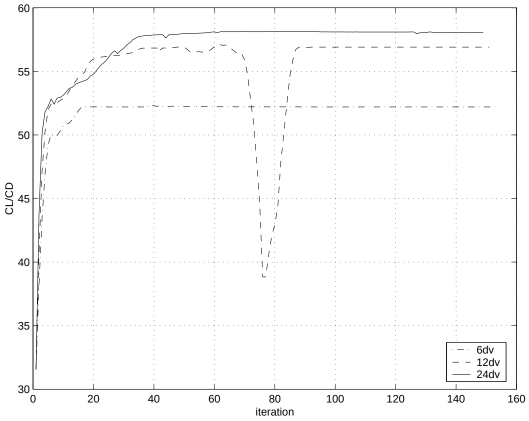

Fig. 9 Lift-to-drag ratio: SNOPT-A

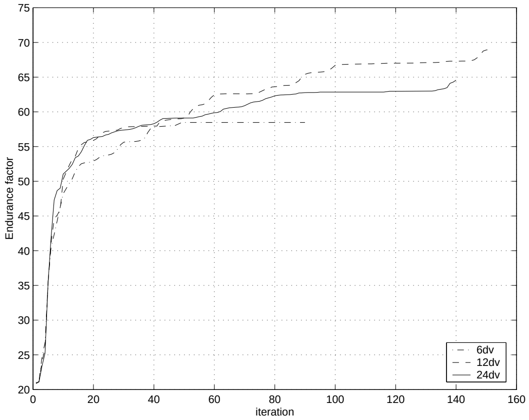

Fig. 10 Endurance factor: BFGS

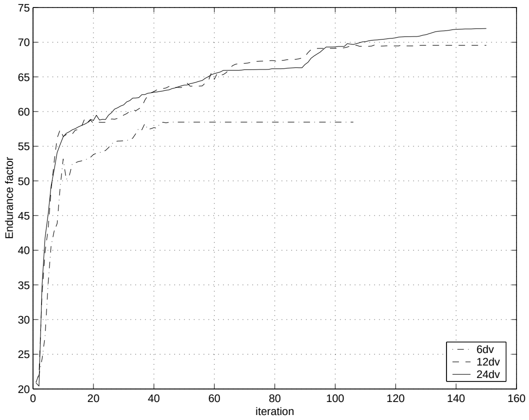

Fig. 11 Endurance factor: SNOPT-Q

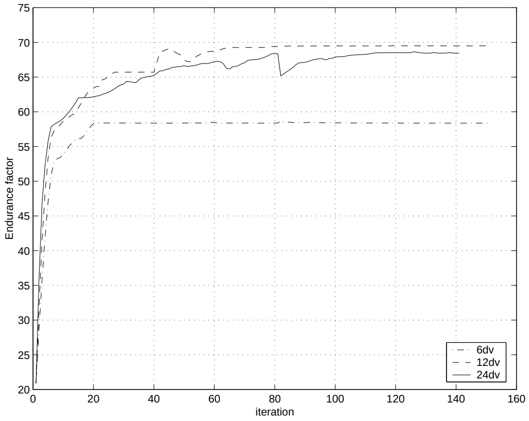

Fig. 12 Endurance factor: SNOPT-A

12 and 24 design variable cases. SNOPT-A converges very well for the 6 and 12 design variable cases, but does not converge within 150 iterations for the 24 design variable case. It is not clear why the performance of SNOPT-A for the 24 design variable case is inferior to its performance with 24 design variables for the lift-to-drag ratio maximization.

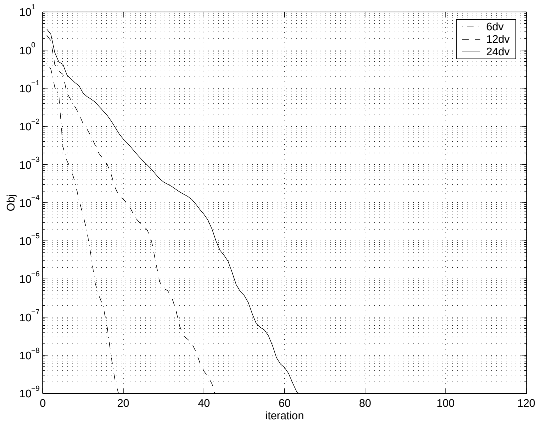

Fig. 13 Inverse design using second scaling option: BFGS

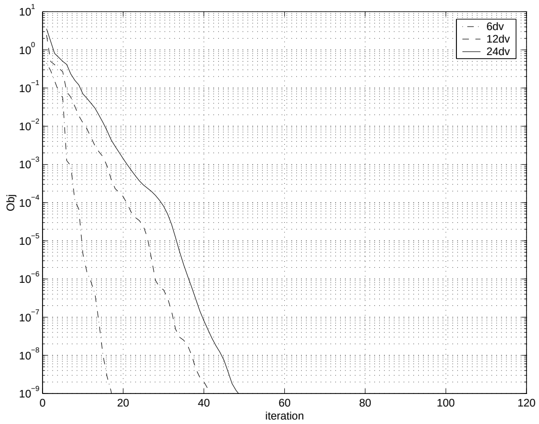

Fig. 14 Inverse design using second scaling option: SNOPT

Effect of Design-Variable Scaling

All of the results presented thus far have used the first scaling option, Eq. 3. We now examine the performance of BFGS and SNOPT-A with the second scaling option, Eq. 4. Figs. 13 to 18 show the results for the three optimization problems using the second scaling option. With a few exceptions, the second option leads to significantly improved convergence. For the inverse problem with 24 design variables, the number of iterations is cut in half for both SNOPT-A and BFGS. For lift-to-drag ratio maximization using BFGS with 24 design variables, convergence is obtained in about 80 iterations when more than 150 iterations were required with the first scaling method. Using SNOPT-A for lift-to-drag ratio maximization with the second scaling method, convergence is obtained in about 20 iterations independent of the number of design variables, and most of the improvement is realized in the first 10 iterations. The endurance factor maximization problem is again more difficult, but the second scaling option is superior in most cases.

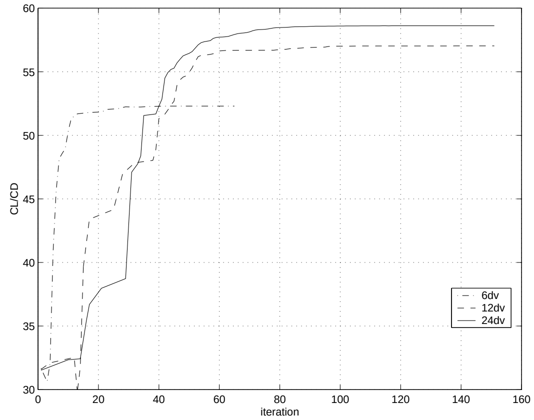

Fig. 15 Lift-to-drag ratio using second scaling option: BFGS

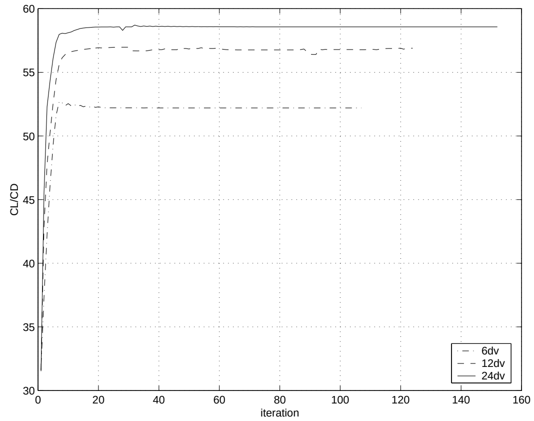

Fig. 16 Lift-to-drag ratio using second scaling option: SNOPT-A

Lift-Constrained Drag Minimization

Finally we consider single- and multi-point lift-constrained drag-minimization. The objective function is given by[11]

$$\mathcal{J} = \begin{cases} \omega_{\mathrm{L}} \left(1 - \frac{C_{\mathrm{L}}}{C_{\mathrm{L}}^*}\right)^2 + \omega_{\mathrm{D}} \left(1 - \frac{C_{\mathrm{D}}}{C_{\mathrm{D}}^*}\right)^2 & \text{if } C_{\mathrm{D}} > C_{\mathrm{D}}^* \\ \omega_{\mathrm{L}} \left(1 - \frac{C_{\mathrm{L}}}{C_{\mathrm{L}}^*}\right)^2 & \text{otherwise} \end{cases} \tag{17}$$

where C_{L}^* and C_{D}^* represent the target lift and drag coefficients. The weights ω_{L} and ω_{D} are user-specified constants. This is a very useful objective function which can be used to achieve a variety of goals. For drag minimization at fixed lift, the target lift coefficient is set to the desired value, and the target drag coefficient is set to an unattainably low value. The designer can control the process by careful selection of the target drag coefficient and the weights. For the present test, C_{L}^* and C_{D}^* are set to 0.733 and 0.013, respectively. The weights are both set to unity. The initial airfoil is the RAE 2822 airfoil, and the thickness constraints are listed in Table 2. The constraint

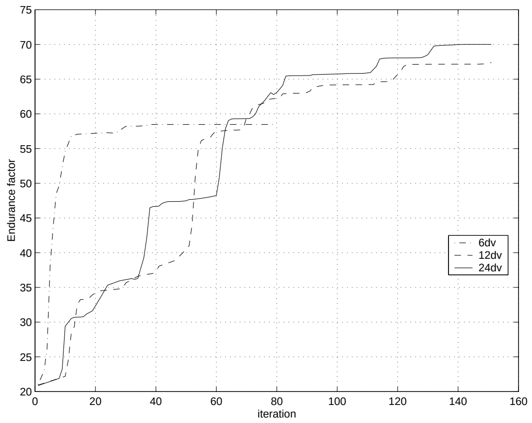

Fig. 17 Endurance factor using second scaling option: BFGS

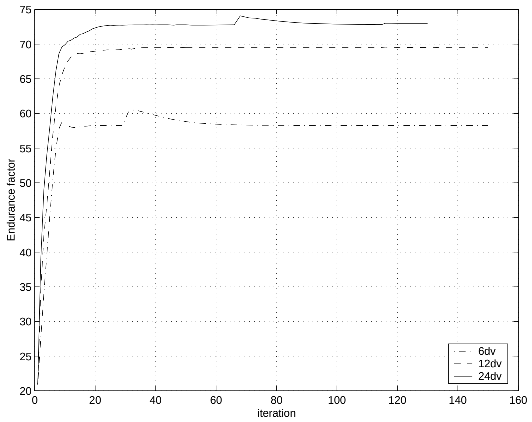

Fig. 18 Endurance factor using second scaling option: SNOPT-A

Constraint #	1	2	3
x/c	35.0	96.0	99.0
t/c	12.06	0.5	0.12

Table 2 Thickness constraints for the lift-constrained drag minimization

at 35%c maintains the initial airfoil thickness, while the constraints near the trailing edge are used to prevent airfoil surface crossover. The Reynolds number is 2.88×10^6.

Convergence histories obtained using BFGS and SNOPT with the first scaling option for this design problem are shown in Figs. 19 and 20. For the SNOPT results the lift constraint is imposed through the objective function, and the thickness constraints are imposed explicitly. With both methods the target lift is achieved reasonably well but not exactly. In order to achieve the target lift more closely, the weight ω_{L} must be increased. The superior convergence of SNOPT relative to BFGS seen in the previous cases is not seen in this case. In fact, with 24 design variables

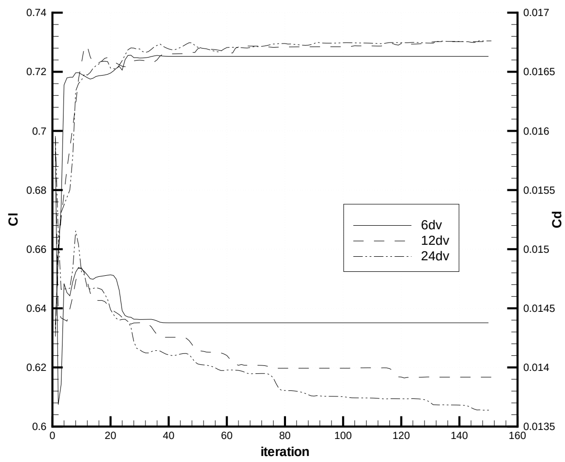

Fig. 19 Lift and drag for lift-constrained drag-minimization: BFGS

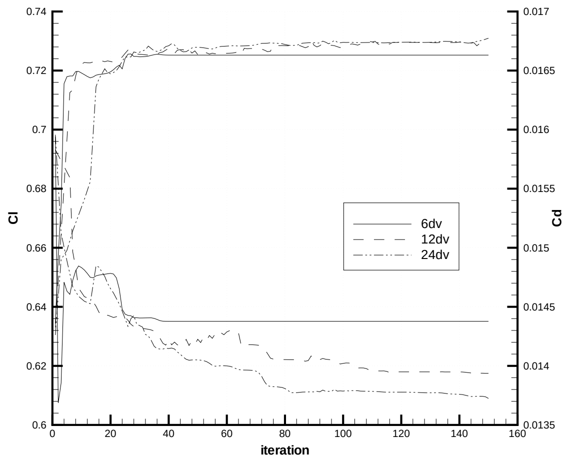

Fig. 20 Lift and drag for lift-constrained drag-minimization: SNOPT

BFGS converges faster than SNOPT and produces a lower drag coefficient after 150 iterations. The resulting airfoil and the associated pressure distribution for the 24 control-point design variable case obtained using BFGS are displayed in Fig. 21. The leading edge geometry is probably not practical for a variety of reasons, indicating that additional objectives or constraints should be applied to achieve a practical airfoil. Furthermore, as shown in Fig. 22, although the drag-divergence Mach number has been successfully increased, there is a significant drag penalty at Mach numbers below 0.71. This is the motivation for multi-point optimization.

The weighted-sum method is used for multi-point optimization problems,[24]

$$\mathcal{J} = \sum_{m=1}^{M} w_m \, \mathcal{J}_m \tag{18}$$

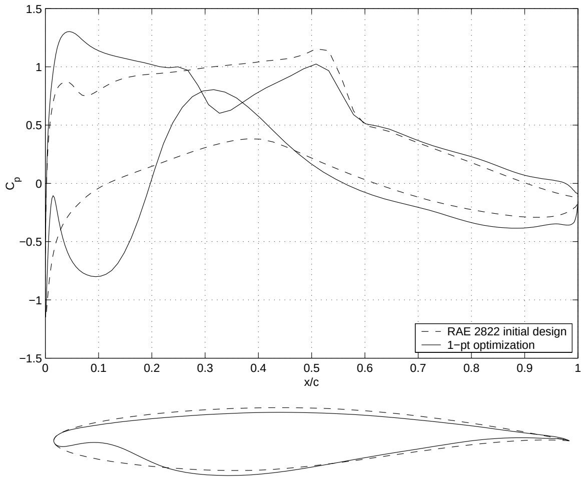

Fig. 21 C_p distribution for lift-constrained drag-minimization (BFGS, 24dv)

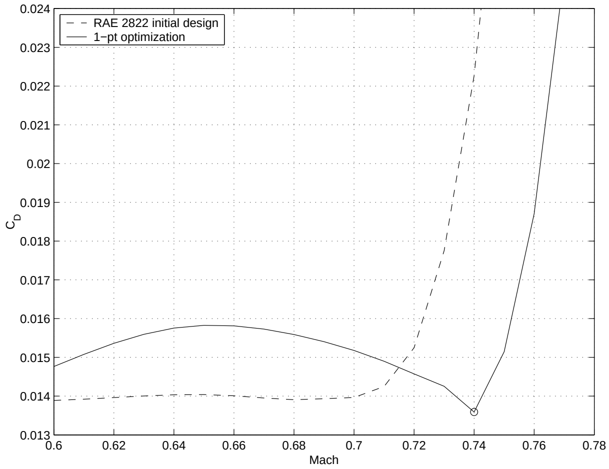

Fig. 22 Drag coefficient for lift-constrained drag-minimization (BFGS, 24dv)

	Mach number	Weight (w_m)
Single-point	0.74	1
Two-points	0.68, 0.74	1, 2
Four-points	0.68, 0.71, 0.74, 0.76	1, 1, 2, 3

Table 3 Mach numbers and weights for the multi-point optimization problem

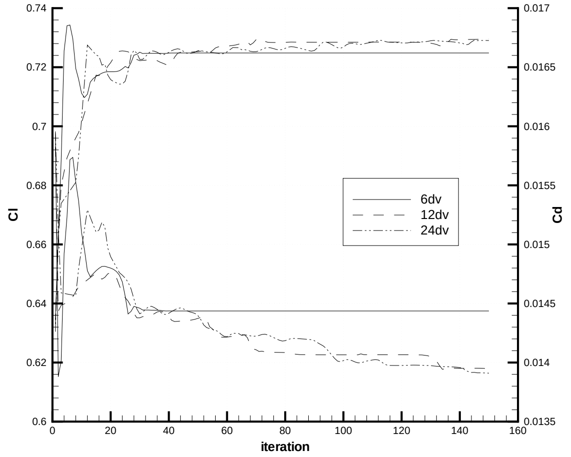

Fig. 23 Lift and drag for the two-point optimization problem: BFGS

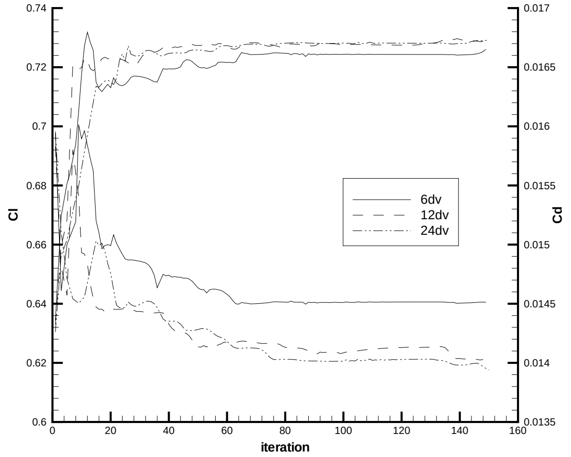

Fig. 24 Lift and drag for the two-point optimization problem: SNOPT

where M denotes the number of design points, w_m represents a user-assigned weight for each design point, and $\mathcal{J}_m$ represents the objective function associated with a specific operating point. For our example we will consider lift-constrained drag minimization at several Mach numbers. The goal is to increase the drag-divergence Mach number without increasing the drag at lower Mach numbers. Each operating point, i.e. Mach number, has its own angle-of-attack as a design variable. Table 3 shows a list of design Mach numbers and the weight assigned to to each Mach number for single-point, two-point, and four-point optimization.

Figs. 23 to 26 show the results for the two-point optimization problem. BFGS again converges somewhat faster than SNOPT. The airfoil and pressure distribution are displayed in Fig. 25, and the drag coefficient is plotted in Fig. 26. The drag is slightly higher at a Mach number of 0.74 than that obtained using single-point optimization, but the drag is lower than the original airfoil for the entire Mach number range shown.

Finally, we consider a four-point optimization in which low drag is sought for a Mach number of 0.76 without sacrificing performance at lower Mach numbers. The results are shown in Figs. 27 to 30. For this problem BFGS substantially outperforms SNOPT, producing significantly higher lift and lower drag after 150 iterations. The performance of SNOPT is inadequate in this case, as it is far from convergence after 150 iterations, as demonstrated by the BFGS results. Furthermore, the SNOPT results are quite flat so there

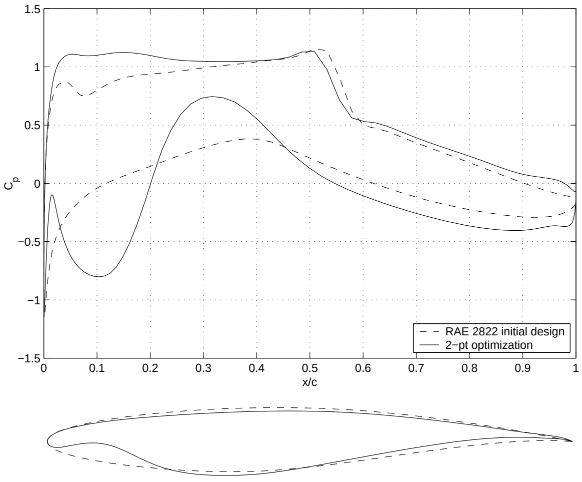

Fig. 25 C_p distribution for the two-point optimization problem (BFGS, 24dv)

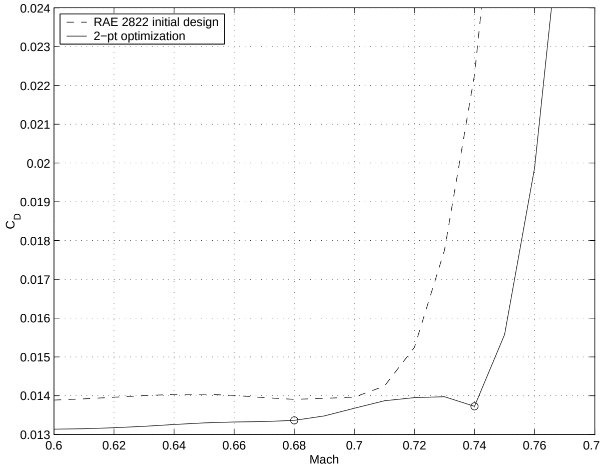

Fig. 26 Drag coefficient for the two-point optimization problem (BFGS, 24dv)

is no indication that convergence will be achieved with further iterations. If the BFGS results were not available, the designer could be forgiven for falsely assuming that the SNOPT results are converged. Since the case labelled “6 dv” has only seven design variables and three constraints, it is not clear why SNOPT converges so poorly.

Conclusions

Two gradient-based optimization algorithms with various constraint handling approaches have been applied to aerodynamic design. These include the BFGS optimizer with a quadratic penalty method and the SNOPT optimizer with the active-set method and the quadratic penalty method. Two different scalings have also been tested. The scaling option in which the control-point design variables are weighted equally (option 2) leads to significantly faster convergence for the cases tested and should be studied further. The comparison between constrained optimization using SNOPT and unconstrained optimization using BFGS

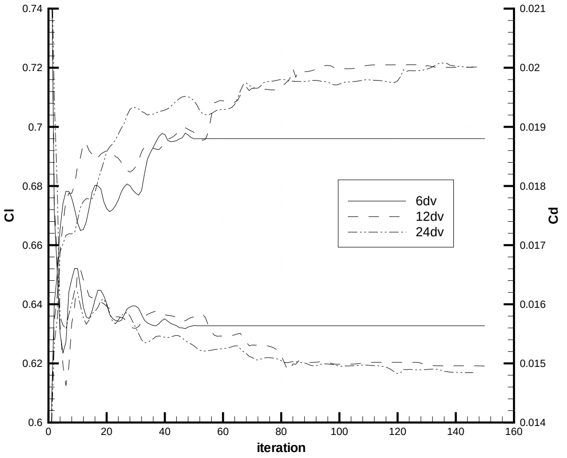

Fig. 27 Lift and drag for the four-point optimization problem: BFGS

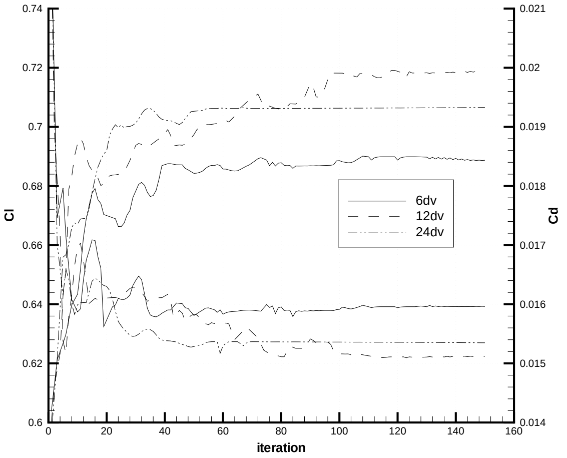

Fig. 28 Lift and drag for the four-point optimization problem: SNOPT

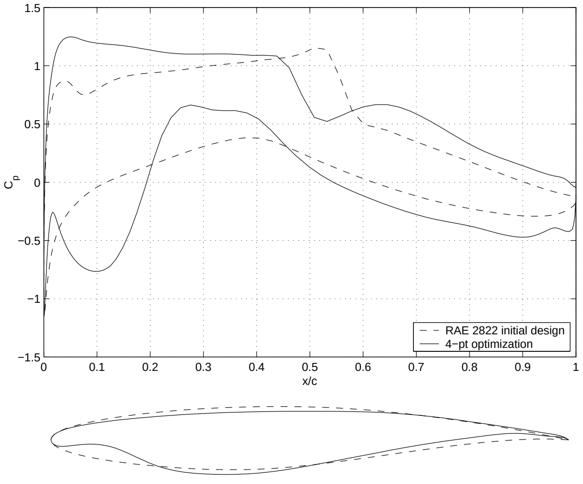

Fig. 29 C_p distribution for the four-point optimization problem (BFGS, 24dv)

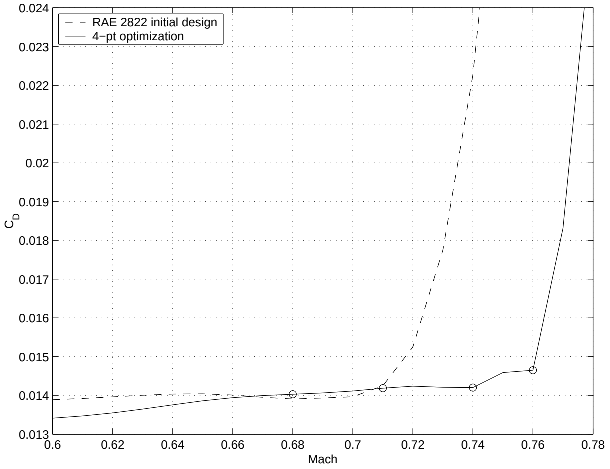

Fig. 30 Drag coefficient for the four-point optimization problem (BFGS, 24dv)

with penalty terms is inconclusive. SNOPT is clearly superior for several design problems but equally clearly inferior for one important objective function. Further study is needed to better understand these results.

References

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

[2] Pironneau, O., "On Optimum Design in Fluid Mechanics," *Journal of Fluid Mechanics*, Vol. 64, No. 1, 1974, pp. 97-110.

[3] Jameson, A., "Aerodynamic Design Via Control Theory," *Journal of Scientific Computing*, 3, 1988, pp. 233-260.

[4] Pulliam, T.H., Nemec, M., Holst, T., and Zingg, D.W., "Comparison of Evolutionary (Genetic) Algorithm and Adjoint Methods for Multi-Objective Viscous Airfoil Optimizations," AIAA Paper 2003-0298, Reno, Jan. 2003.

[5] Weinerfelt, P., "Aerodynamic Optimization Using Control Theory and Surface Mesh Points as Control Variables," Tech. Rep. FAU-97.044, SAAB, Linkőping, 1997.

[6] Obayashi, S., "Inverse Optimization Method for Aerodynamic Shape Design," *Recent Development of Aerodynamic Design Methodologies - Inverse Design and Optimization*, edited by K. Fujii and G. S. Dulikravich, Notes on Numerical Fluid Mechanics, Vol. 68, 1999.

[7] Dennis Jr., J. E. and Schnabel, R. B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall Inc., Englewood Cliffs, N.J., 1983.

[8] Nocedal, J. and Wright, S. J., *Numerical Optimization*, Springer–Verlag, New York, 1999.

[9] Jameson, A. and Vassberg, J. C., "Studies of Alternative Numerical Optimization Methods Applied to the Brachistochrone Problem," *Computational Fluid Dynamics Journal*, Vol. 9, No. 3, 2000, pp. 281–296.

[10] Mohammadi, B., "A New Optimal Shape Design Procedure for Inviscid and Viscous Turbulent Flows," *International Journal for Numerical Methods in Fluids*, Vol. 25, 1997, pp. 183–203.

[11] Nemec, M. and Zingg, D. W., "Newton-Krylov Algorithm for Aerodynamic Design Using the Navier–Stokes Equations," *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1146–1154.

[12] Wrenn, G. A., "An Indirect Method for Numerical Optimization Using the Kreisselmeier-Steinhauser Function," NASA CR–4220, March 1989.

[13] Anderson, W. K. and Venkatakrishnan, V., "Aerodynamic Design Optimization on Unstructured Grids with a Continuous Adjoint Formulation," *Computers & Fluids*, Vol. 28, 1999, pp. 443–480.

[14] Nielsen, E. J. and Anderson, W. K., "Aerodynamic Design Optimization on Unstructured Meshes Using the Navier-Stokes Equations," *AIAA Journal*, Vol. 37, No. 11, 1999, pp. 1411–1419.

[15] Baysal, O. and Ghayour, K., "Continuous Adjoint Sensitivities for Optimization with General Cost Functionals on Unstructured Meshes," *AIAA Journal*, Vol. 39, No. 1, 2001, pp. 48–55.

[16] Gill, P. E., Murray, W., and Wright, M. H., *Practical Optimization*, Academic Press Inc., Toronto, 1981.

[17] Murray, W., "Sequential quadratic programming methods for large-scale problems," *J. Comput. Optim. Appl.*, No. 7, 1997, pp. 127–142.

[18] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM J. OPTIM.*, Vol. 12, No. 4, 2002, pp. 979–1006.

[19] Willcox, K. and Wakayama, S., "Simultaneous Optimization of a Multiple-Aircraft Family," AIAA Paper 02–1423, Apr. 2002.

[20] Soemarwoto, B. I. and Labrujère, T. E., "Airfoil Design and Optimization Methods: Recent Progress at NLR," *Int. J. Numer. Meth. Fluids*, Vol. 30, 1999, pp. 217–228.

[21] Li, W., Huyse. L., and Padula, P., "Robust Airfoil Optimization to Achieve Drag Reduction over a Range of Mach Numbers," *Struct. Multidisc. Optim.* , Vol. 24, 2002, pp. 38-50.

[22] Melvin, R. G., Huffman, W. P., Young, D. P., Johnson, F. T., Hilmes, C. L., and Bieterman, M. B., "Recent Progress in Aerodynamic Design Optimization," *Int. J. Numer. Meth. Fluids*, Vol. 30, 1999, pp. 205–216.

[23] Zingg, D.W., Nemec, M., and Chisholm, T.T., "A Newton-Krylov Algorithm for Aerodynamic Analysis and Design," in *Computational Fluid Dynamics 2002*, Armfield, Morgan and Srinivas, eds., Springer-Verlag, Germany, 2003.

[24] Nemec, M., and Zingg, D.W., "Multi-Point and Multi-Objective Aerodynamic Shape Optimization," accepted by the AIAA J., 2003.

[25] Pulliam, T. H., "Efficient Solution Methods for the Navier-Stokes Equations," Tech. rep., Lecture Notes for the von Kármán Inst. for Fluid Dynamics Lecture Series: Numerical Techniques for Viscous Flow Computation in Turbomachinery Bladings, Brussels, Belgium, Jan. 1986.

[26] Spalart, P. R. and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows," AIAA Paper 92–0439, Jan. 1992.

[27] Pueyo, A. and Zingg, D. W., "Efficient Newton-Krylov Solver for Aerodynamic Computations," *AIAA Journal*, Vol. 36, No. 11, 1998, pp. 1991–1997.

[28] Nemec, M., *Optimal Shape Design of Aerodynamic Configurations: A Newton-Krylov Approach*, Ph.D. thesis, University of Toronto Institute of Aerospace Studies, 2002.

[29] Jameson, A., Pierce, N. A., and Martinelli, L., "Optimum Aerodynamic Design using the Navier-Stokes Equations," *Theoretical and Computational Fluid Dynamics*, Vol. 10, No. 1, 1998, pp. 213–237.

[30] Hua, J., Kong, F.M., Liu, P.J., and Zingg, D.W., "Optimization of Long-Endurance Airfoils," AIAA Paper 2003-3500, Orlando, June 2003.