AIAA Journal
Vol. 43, No. 10, October 2005

Technical Notes

TECHNICAL NOTES are short manuscripts describing new developments or important results of a preliminary nature. These Notes should not exceed 2500 words (where a figure or table counts as 200 words). Following informal review by the Editors, they may be published within a few months of the date of receipt. Style requirements are the same as for regular contributions (see inside back cover).

Numerical Evaluation of Optimization Algorithms for Low-Reynolds-Number Aerodynamic Shape Optimization

Marc Secanell* and Afzal Suleman†
University of Victoria,
Victoria, British Columbia V8W 3P6, Canada

I. Introduction

In recent years, active research has produced a variety of efficient methods for constrained optimization.[1–4] However, in only a few papers are different optimization algorithms used to solve the same aerodynamic design optimization problem and the results compared. As an example, Pulliam et al.[5] compared the results obtained by using genetic algorithms and a quasi-Newton method. Similarly, a numerical evaluation of the performance of different existing gradient-based optimization algorithms used for solving low-Reynolds-number aerodynamic shape optimization problems could be beneficial. Such comparisons exist in other disciplines, such as structural mechanics[6] and flow control problems,[7] and they showed that careful selection of the optimization algorithm can considerably reduce the amount of iterations needed to reach the optimum.

In this Note, the performances of several gradient-based optimization algorithms are evaluated for their suitability to the solution of low-Reynolds-number aerodynamic shape optimization problems. In particular, the algorithms available in the optimization package DOT[8] are compared: the modified method of feasible directions,[4] sequential linear programming,[2,4] and sequential quadratic programming.[1] The algorithms in the DOT package are used because they have been extensively tested and are proven to yield good results in the field of structural optimization. Furthermore, these algorithms are considered to be representative of the most common optimization methods used for gradient-based optimization.

II. Methodology for Aerodynamic Shape Optimization

The optimization package DOT and a computational-fluid-dynamics (CFD) solver are linked to create a program for aerodynamic shape optimization of airfoils at low Reynolds number and at any angle of attack.

The CFD code used is the Structured Parallel Research Code (SPARC) developed by Magagnato.[9] SPARC is a parallel solver that uses a multiblock structured grid to solve the flowfield. It uses local time stepping and a multigrid technique to accelerate convergence to the steady state. In this Note, the flow around the airfoils is assumed to be steady, turbulent, viscid, and incompressible. The Reynolds-averaged Navier–Stokes equations are solved using a finite volume formulation and a central-difference discretization scheme in space, and the turbulence is accounted for by the Spalart–Allmaras one-equation turbulence model without the tripping term.[10]

To represent and control the airfoil shapes a uniform cubic B-spline with 15 control points is used. From the 15 control points, the y coordinates of control points numbered 1–5 and 7–11 in Fig. 1 are used as design variables. Furthermore, the y distance between the two most external points at the leading edge is also used as a design variable to control the sharpness of the leading edge.

During optimization, the mesh used for the aerodynamic solver is deformed to adapt to the different airfoil shapes. To adapt the mesh to the deformed airfoil, a mesh-deformation technique that uses a combination of the spring analogy and transfinite interpolation is used.[11]

Finally, to compute the gradients used during the optimization, forward difference is used.[12] The main reasons for this choice are as follows: faster computation of the gradients compared to automatic differentiation[13] and complex differentiation[12,14] and ease of implementation and parallelization. To reduce the computational time necessary to compute the gradients, the gradient calculations are parallelized by means of a scheduling program, namely, portable batch system (PBS).[15] All of the analysis runs are sent to PBS at once, and PBS allocates the necessary number of processors for each analysis run.

III. Optimization Algorithms

The design process just described is used with the three different optimization algorithms implemented in DOT: the modified method of feasible directions, sequential linear programming, and sequential quadratic programming.

The modified method of feasible directions (MMFD) appeared in 1983 as a combination of the method of feasible direction[4,16] and the reduced gradient method of Wolfe[4,16,17] Assuming a feasible

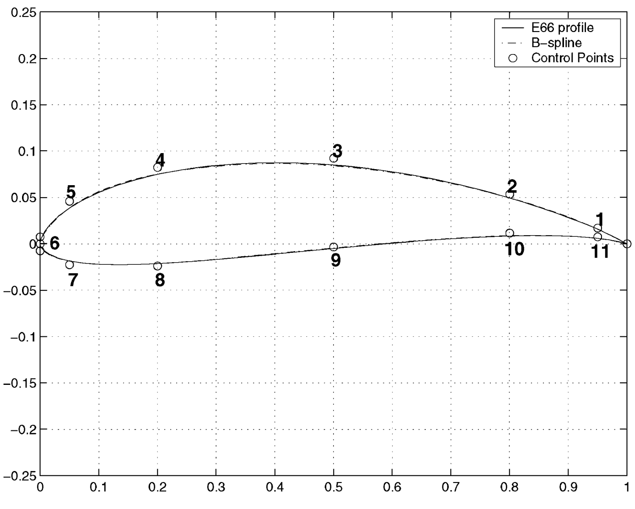


Fig. 1 Eppler 66 airfoil profile represented by using a 15-point B-spline.

Received 30 July 2004; revision received 14 February 2005; accepted for publication 24 February 2005.

*M.A.Sc. Student, Mechanical Engineering Department; secanell@uvic.ca. Student Member AIAA.

†Professor, Mechanical Engineering Department; suleman@uvic.ca. Associate Fellow AIAA.

initial point, this method solves the optimization problem by finding a search direction $\boldsymbol{d}$ that reduces the objective function and, at the same time, guarantees the satisfaction of the constraints. Next, a one-dimensional search is made to obtain a parameter α^* that minimizes the objective function and satisfies the constraints in the search direction $\boldsymbol{d}$. Finally, the design variables are updated by using $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha^* \boldsymbol{d}$. This process is repeated until a solution is found. If the initial point is infeasible, an initial subproblem must be solved first in order to obtain a feasible initial point, and then the process just described can be executed.

Sequential linear programming (SLP)[8,18–21] is considered to be one of the simplest methods to program, the only requirement being an efficient linear-programming solver. To solve a nonlinear-programming problem, a linear-programming (LP) subproblem is created by linearizing the original objective function and the constraints at each iteration. To guarantee the accuracy of the linearization, move limits that constrain the maximum possible change of the design variables from their reference values are introduced.[18–21] Then, the LP problem is solved to obtain an increment of the design variables that moves toward the optimal solution. The design variables are updated using the increment $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{d}$. This process is repeated iteratively until the optimum is found.

Sequential quadratic programming (SQP) is considered to be one of the most efficient methods for solving nonlinear constraint optimization problems.[4,8] To solve a nonlinear-programming problem, the SQP method creates a quadratic approximation of the nonlinear problem using the Karush–Kuhn–Tucker conditions of the original problem. The solution of the quadratic subproblem gives a direction, $\boldsymbol{d}$, that leads the current design to an improved design. Then, a line search is performed in this direction, $\boldsymbol{d}$ to obtain the next design. Finally, the quadratic subproblem is updated, and the process is repeated until convergence is achieved.

For all aforementioned methods, convergence to the optimum is assumed if $[f(\boldsymbol{x}_{k+1}) - f(\boldsymbol{x}_k)]/f(\boldsymbol{x}_k) \leq 0.001$ in two consecutive iterations or $\boldsymbol{d}_i \leq 0.0001\ \forall\, i = 1, \ldots, n$.

IV. Discussion and Results

Once the computational design tool described in the preceding sections has been implemented in FORTRAN90, the design tool can then be used to evaluate the performance of the different optimization algorithms in the design of airfoils for aircraft at low Reynolds numbers. The only requirements are an initial airfoil shape and fluid mesh that yield accurate results at the Reynolds numbers under study. Section IV.A focuses on the testing and validation of a fluid mesh. Once a grid has been selected, a parametric study of different step sizes is performed in Sec. IV.B to obtain the step size that yields the most accurate gradients. Finally, in Sec. IV.C the performance of the different optimization algorithms is evaluated by solving a lift-constrained minimum drag airfoil.

A. Grid Study

Several variations of the grid used by Stockdill[23] to successfully predict the lift and the drag of a NACA0012 at $Re = 3 \times 10^6$ were used in this paper to design the grid to be used to solve the fluid flow at the Reynolds numbers of interest. In this case, the Reynolds numbers of interest are of the order of $Re = 5 \times 10^5$. Because experimental data exist for an Eppler 64 airfoil at $Re = 2 \times 10^5$, the grid studies are performed for the Eppler 64 at the aforementioned Reynolds number so that numerical and experimental results can be compared.

To perform the grid study, a grid was generated, and this grid was refined four times in all directions. Grid 1 is the coarsest grid, and it has 384 nodes (48 × 9), 20 nodes around the airfoil, and the first node from the boundary of the airfoil is set at a distance of 2×10^{-4}. Grid 5 is the most refined grid, and it is obtained by refining grid 1 four times in all directions; therefore, it has 98,304 nodes (768 × 128), 320 nodes around the airfoil, and the first node from the boundary of the airfoil is at a distance of 2×10^{-5}. Each refined grid was used to solve the flowfield. Tables 1 and 2 show the total lift c_l, total drag c_d, friction drag c_{df}, and pressure drag c_{dp}. Values for the Eppler 64 airfoil were computed using the different grids at an angle of attack of 0 and 4 deg, respectively.

Table 1 Lift and drag values for different grid refinements at $\alpha = 0$, $Re_c = 2 \times 10^5$

Grid	c_l	c_d	c_{df}	c_{dp}
1	0.39400	0.04200	0.00446	0.03754
2	0.48946	0.02760	0.01417	0.01343
3	0.51137	0.01755	0.01207	0.00549
4	0.50472	0.00957	0.00596	0.00360
5	0.47964	0.00780	0.00490	0.00290
Exp.[24]	0.50	0.0125	——	——

Table 2 Lift and drag values for different grid refinements at $\alpha = 4$, $Re_c = 2 \times 10^5$

Grid	c_l	c_d	c_{df}	c_{dp}
1	0.76468	0.05826	0.00434	0.05392
2	0.89256	0.03978	0.01335	0.02644
3	0.92985	0.02408	0.01181	0.01227
4	0.90990	0.01417	0.00558	0.00859
5	div	div	div	div
Exp.[24]	0.925	0.0145	——	——

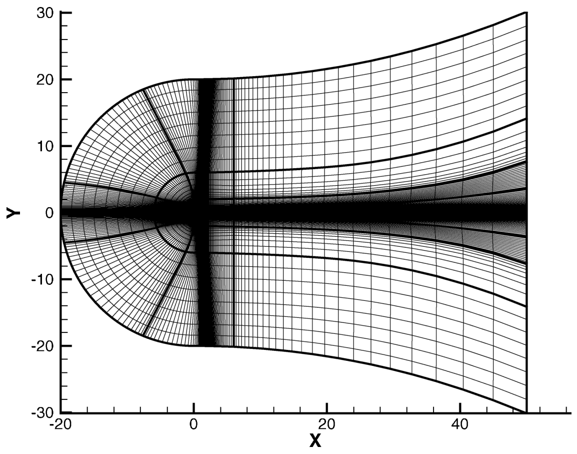


Fig. 2 Grid 4 around the Eppler 64 airfoil.

Looking at the evolution of the lift coefficient in Tables 1 and 2, the predicted lift coefficient can be observed to converge to a value near the experimental lift, with an error of less than 5% with respect to experimental data for grids 4 and 5. The evolution of the total drag in Tables 1 and 2 also shows convergence toward the experimental drag with an error of less than 20% with respect to experimental data for grid 4. The total predicted drag has a larger error than the predicted lift. This is because of the strong dependence that the total drag has on the friction drag. The friction drag is difficult to predict using the Spalart–Allmaras turbulence model. This model does not accurately predict laminar-to-turbulent transition, and this inaccuracy has a strong impact on the prediction of the friction drag at the low Reynolds numbers under consideration.[10,24,25] Grid 5 did not converge to a solution in Table 2 most likely because of the large level of refinement around the airfoil boundary, which is necessary to properly resolve the boundary layer and to predict the laminar-to-turbulent transition. This results in the formation of large-aspect-ratio cells near the airfoil boundary. In further studies, a method for predicting the laminar-to-turbulent transition and a tripping source term should be introduced in the Spalart–Allmaras model together with a wall function. This will reduce the number of grid refinements required and will reduce the convergence problems of the solver. Because grid 4 yields reasonably good results for lift and drag predictions and because a coarse grid is preferable as a result of its lower computational cost for optimization, grid 4 is selected as the base grid. Grid 4 is shown in Fig. 2 and a detail of the grid around the airfoil is shown in Fig. 3 Grid 4 has 36 blocks, 24,576 nodes, 160 nodes around the airfoil, and the first node from the boundary of the airfoil is at a distance of 4×10^{-5}, which results in a y^+ value of 0.5.

B. Study of the Step Size Used to Compute the Gradient

The gradients of objective function and constraints are computed using forward differentiation. This method for computing the gradients is subjected to the step-size dilemma. To obtain the optimal step size for the gradient calculations, the lift and drag gradients for each design variable in the optimization problem are plotted vs step sizes from 10^{-2} to 10^{-7} in Figs. 4 and 5. From Figs. 4 and 5 it appears that the most appropriate step size is 10^{-4}, 10^{-5}, or 10^{-6} depending on the variable. In this case, a step size of 10^{-5} is chosen for all of the variables as the step size for gradient calculations.

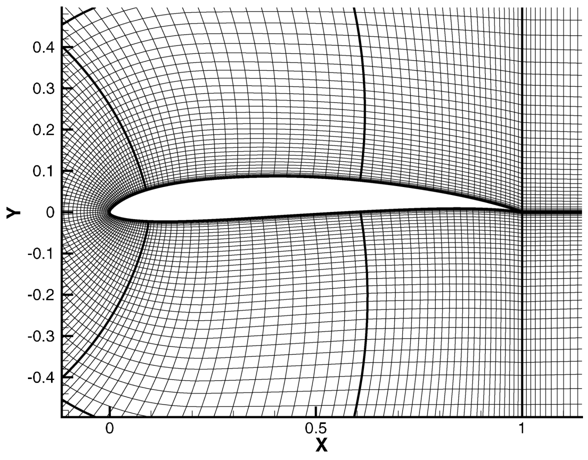


Fig. 3 Detail of grid 4 around the Eppler 64 airfoil.

C. Drag Minimization

In this section, a drag coefficient minimization problem subject to a minimum lift coefficient requirement is solved. The design problem is to obtain an airfoil with minimum drag and a minimum lift coefficient of 0.8, at a $Re = 5 \times 10^5$ and with a 2-deg angle of attack. Thickness constraints are imposed on the geometry of the airfoil, and bounds are also imposed on the design variables. In particular, geometrical constraints are imposed to obtain a minimum thickness of 1% of the chord as described in Table 3. The bounds of the design variables are presented in Table 4, where the design variables are the y coordinate of the control points of the B-spline that represents the airfoil, and the numbering corresponds to the numbering in Fig. 1

Table 3 Geometric constraints of the design problem

Constraint	Value
1	$x_1 - x_{11} \geq 0.01$
2	$x_2 - x_{10} \geq 0.01$
3	$x_3 - x_9 \geq 0.01$
4	$x_4 - x_8 \geq 0.01$
5	$x_5 - x_7 \geq 0.01$

Table 4 Lower and upper bounds of the design variables

Bound	x_1	x_2	x_3	x_4	x_5	x_{LE}	x_7	x_8	x_9	x_{10}	x_{11}
Lower	−0.2	−0.2	−0.2	−0.2	0.0	0.005	−0.2	−0.2	−0.2	−0.2	−0.2
Upper	0.2	0.2	0.2	0.2	0.2	0.05	0.0	0.2	0.2	0.2	0.2

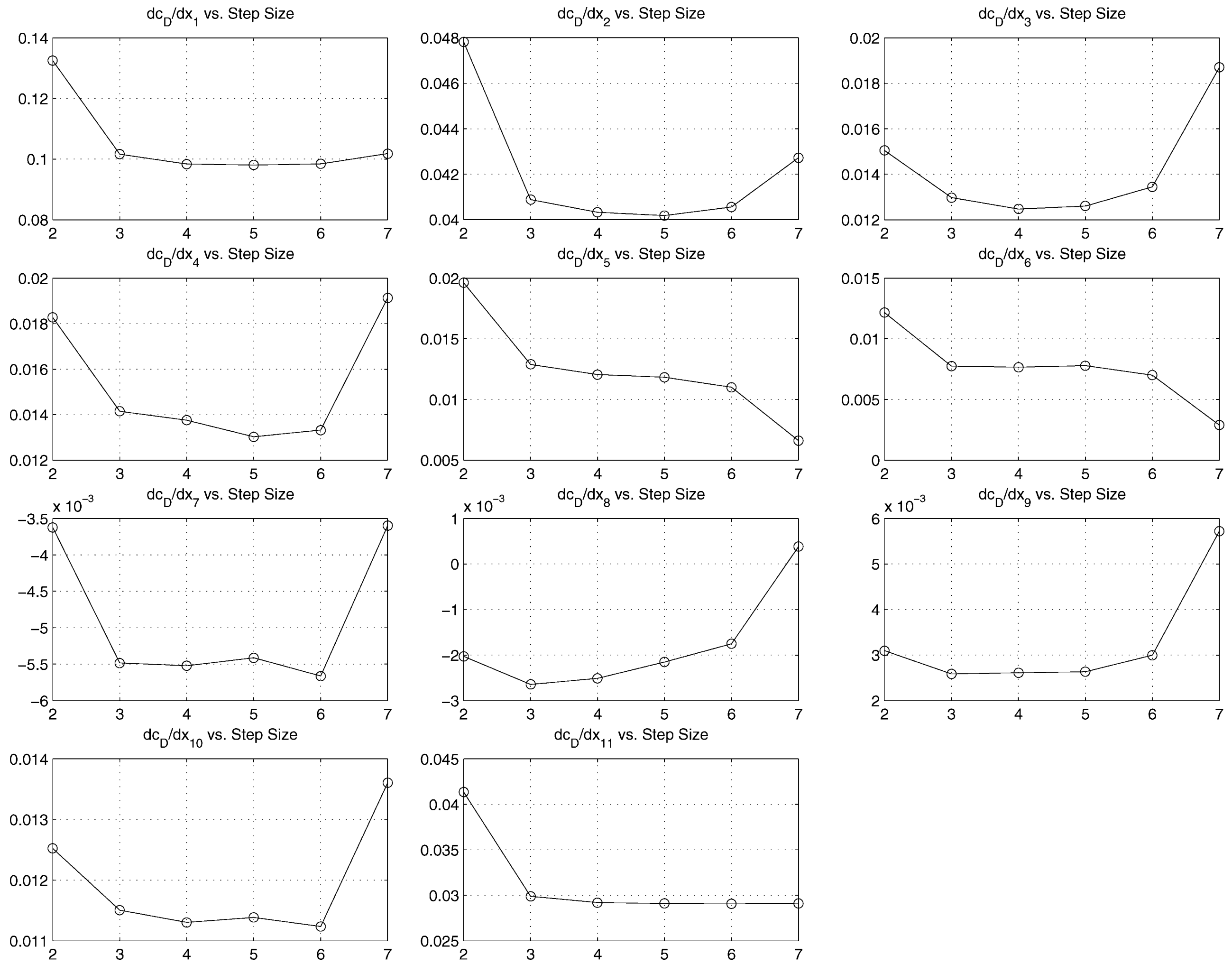
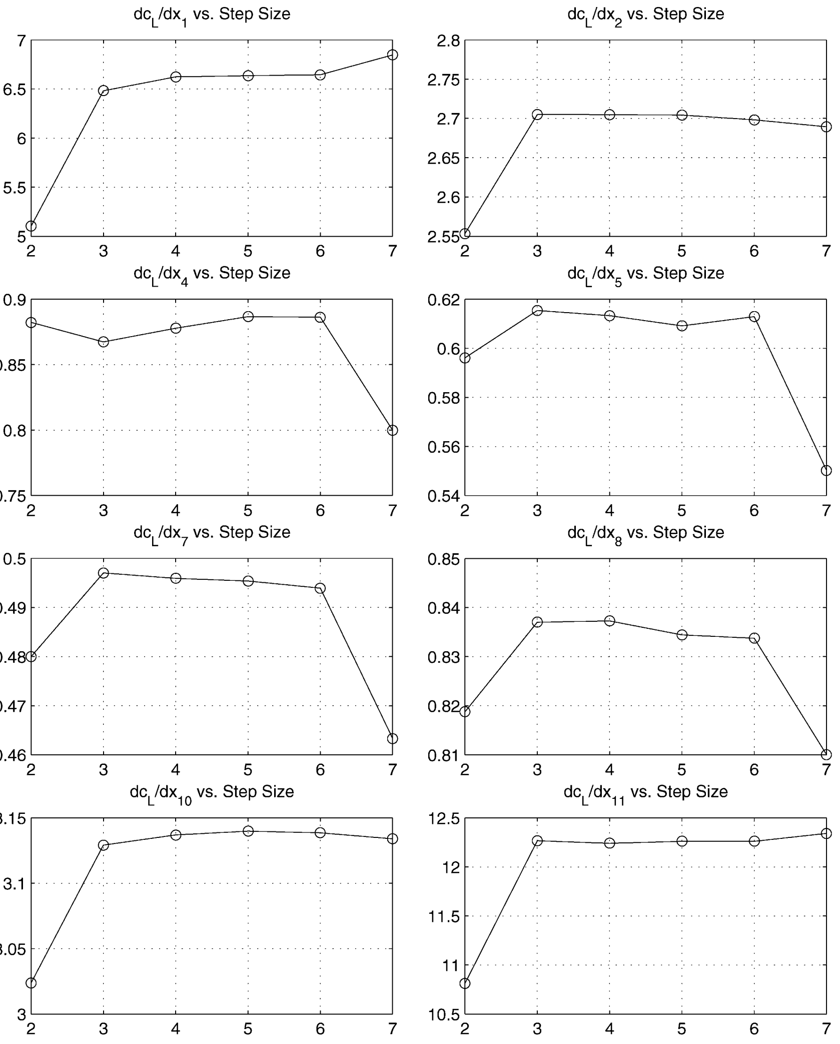


Fig. 4 Value of the drag coefficient gradient with respect to the decimal logarithm of the step size used to compute the gradient using forward differences.



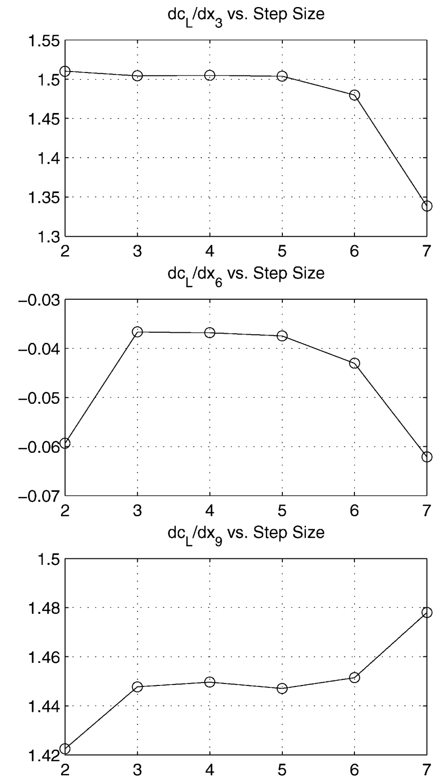


Fig. 5 Value of the lift coefficient gradient with respect to the decimal logarithm of the step size used to compute the gradient by using forward differences.

with x_{LE} representing the distance between leading-edge points. Notice that the variables x_5 and x_7 have bounds different from those of the other variables. This is because of the method used to deform the grid. If the same bounds are used, the mesh-deformation algorithm creates a fluid mesh with negative cells, and the analysis program is unable to solve the flow around the airfoil; x_{LE} also has different bounds. This is because of the different nature of the variable in that x_{LE} represents the distance between the two points at the leading edge and, therefore, must always be positive.

To solve the design problem, an Eppler 66 airfoil was used as the initial airfoil for the optimization procedure. This airfoil was chosen because it is one of the airfoils recommended for the design of low-Reynolds-number aircraft.[24] Previous to the optimization, the lift and drag coefficients for the airfoil were computed to be 0.864 and 1.009×10^{-2}, respectively. The lift and the drag were obtained by using SPARC with three processors on a 22-processor Linux cluster. The lift and drag coefficients were obtained after approximately 45 min of CPU time. During the optimization, the lift and drag coefficients were obtained using the same grid and number of processors. However, the flowfield was restarted from the last flowfield solution, thereby enabling a reduction in the number of iterations prior to convergence. This results in a reduction of 15 min of CPU time.

Starting with the Eppler 66 airfoil as the initial design, the design problem was solved using the three optimization algorithms described in Sec. III. The three optimization algorithms converged to a solution with similar aerodynamic characteristics as shown in Table 5 and with a similar airfoil shape as illustrated in Fig. 6 and Table 6. Furthermore, the optimal solution obtained with all

Table 5 Aerodynamic characteristics of the initial and optimal solution at $Re = 5 \times 10^5$ and $\alpha = 2$

Aerodynamic characteristics	Eppler 66	MMFD	SLP	SQP
c_l	0.86428	0.80000	0.79993	0.80007
$c_d \times 10^{-2}$	1.00867	0.81144	0.81235	0.81270
$c_{\mathrm{df}} \times 10^{-2}$	0.50905	0.47188	0.47418	0.47083
$c_{\mathrm{dp}} \times 10^{-2}$	0.49962	0.33957	0.33817	0.34188
L/D[a]	85.68	98.59	98.47	98.45

[a]L/D = lift-to-drag ratio.

three methods satisfied all aerodynamic and geometric constraints as shown in Tables 5 and 7. From Table 5, it can be observed that the lift constraint is active, that is, lift is 0.8. This was expected, as it is well known that lift and drag are opposing goals. From Table 7, it can be observed that all geometric constraints are active or near active. Therefore, it can be concluded that to obtain an airfoil with minimum drag, the airfoil must be extremely thin. Note that, from a structural point of view, this presents a challenge and could prove to be problematic.

The SQP algorithm was able to reach the optimum in the least number of function evaluations, as shown in Table 8 and in the convergence plot in Fig. 7. Similar results were also reported by Schittowski et al.[6] and Ghattas and Bark,[7] where optimization involving partial differential equations was also performed. The SQP method also obtained the optimum airfoil shape using the fewest iterations, that is, gradient evaluations, and without relying heavily on the line search: it required only 26 internal function evaluations.

Table 6 Value of the design variables at the optimal solution

Airfoil	x_1	x_2	x_3	x_4	x_5	x_{LE}	x_7	x_8	x_9	x_{10}	x_{11}
Eppler 66	$1.725E-02$	$5.350E-02$	$9.250E-02$	$8.250E-02$	$4.600E-02$	$1.500E-02$	$-2.250E-02$	$-2.400E-02$	$-3.500E-03$	$1.150E-02$	$7.450E-03$
MMFD	$1.323E-02$	$3.334E-02$	$6.212E-02$	$5.303E-02$	$2.743E-02$	$5.302E-03$	$-4.526E-10$	$4.303E-02$	$5.212E-02$	$2.339E-02$	$3.233E-03$
SLP	$1.278E-02$	$3.295E-02$	$6.428E-02$	$5.684E-02$	$2.882E-02$	$6.010E-03$	0.000	$4.684E-02$	$5.428E-02$	$2.295E-02$	$2.778E-03$
SQP	$1.378E-02$	$3.069E-02$	$6.294E-02$	$5.430E-02$	$2.771E-02$	$6.265E-03$	$-2.849E-13$	$4.419E-02$	$5.294E-02$	$2.069E-02$	$3.775E-03$

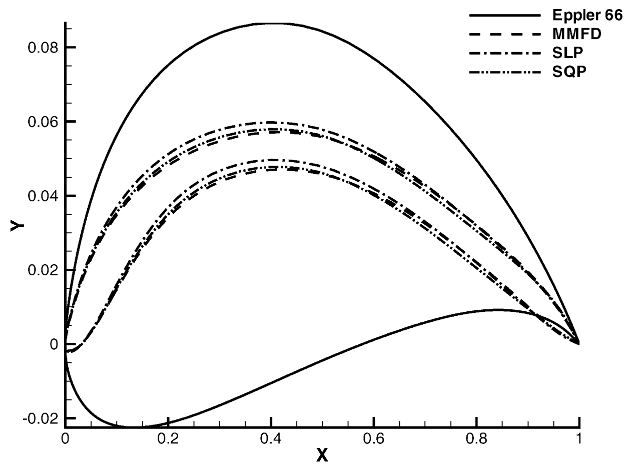


Fig. 6 Initial and optimal airfoil shapes for MMFD, SLP, and SQP.

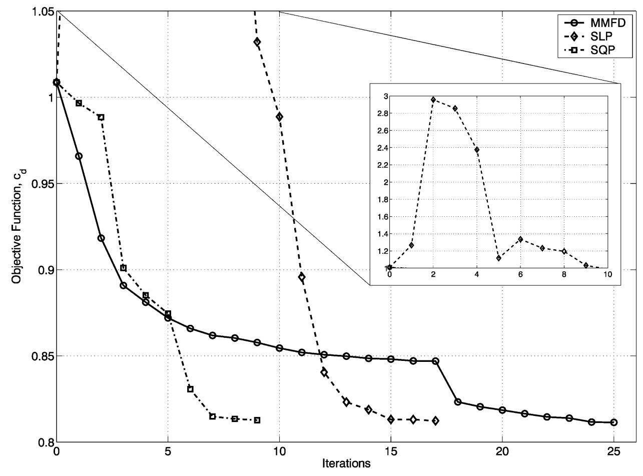


Fig. 7 Convergence history plot of the drag minimization problem solved by using MMFD, SLP, and SQP algorithms.

The MMFD needed to perform 26 gradient evaluations and 231 internal function evaluations during the line search to obtain the optimal solution. The number of internal function evaluations considerably reduced the efficiency of the method because these cannot be parallelized. The reason for the large number of internal function evaluations in each iteration is that the lift coefficient constraint must be satisfied at each iteration. The MMFD algorithm has to solve a Newton–Raphson problem to guarantee that the lift constraint is active in the next iteration. This proved to be extremely expensive in terms of function evaluations.

The SLP algorithm converged to the solution after 17 iterations. It needed to perform 17 gradient evaluations and only 25 internal function evaluations. Therefore, in this case, the SLP method is less efficient than the SQP, but more efficient than the MMFD method. Even though the SLP is less efficient than the SQP method, it is important to note the small number of internal function evaluations it required. The SLP method used seven iterations more than the SQP method; however, the number of internal evaluations is 25, one internal function evaluation less than the SQP method. The reduction in the number of internal function evaluations is achieved by using moving limits instead of a line search. Taking into account that efficient ways exist to compute the gradients of the objective function and constraints[8–21] and that function evaluations are always expensive, the SLP has to be considered as a potential candidate for aerodynamic shape optimization.

Through careful observation of the convergence history in Fig. 7, the SLP algorithm increases the objective function in the first iterations instead of decreasing it. Then, from iteration 9 to iteration 17 the algorithm starts to reduce the objective function until it reaches the optimum. The initial behavior is caused by a poor

Table 7 Value of the geometric constraint at the optimal solution

	Constraint				
Airfoil	1	2	3	4	5
Eppler 66	$2.500E-04$	$-3.200E-02$	$-8.600E-02$	$-9.650E-02$	$-5.850E-02$
MMFD	$9.313E-10$	$1.863E-09$	$1.863E-09$	$-1.863E-09$	$-1.743E-02$
SLP	$2.328E-10$	$2.241E-06$	$-2.312E-06$	$-6.670E-06$	$-1.882E-02$
SQP	$-1.164E-09$	0.000	$1.863E-09$	$-1.132E-04$	$-1.771E-02$

Table 8 Number of function and gradient evaluations before optimum

Computational analyses	MMFD	SLP	SQP
Iterations	26	17	10
Gradient evaluations	26	17	10
Internal function evaluations	231	25	26
Total function evaluations	543	229	146

selection of the initial move limits. Initially, the move limits chosen are too large to guarantee an accurate linear approximation of the objective function and constraints. However, as the algorithm evolves, the move limits are reduced to the appropriate values. Therefore, even though the move limits techniques are responsible for a reduction in the number of internal function evaluations, it is also responsible for an increase in the number of iterations required before the optimum is reached. Therefore, the SLP method could be improved upon by implementing an efficient method for obtaining the initial move limits, for example, one of the methods discussed by Chen.[19] Once this technique is implemented, it is possible that the SLP algorithm could be as efficient as the SQP algorithm because the method in DOT used to dynamically increase or decrease the move limits once the initial move limits have been selected appears to perform well for this problem. Finally, to reduce the number of function evaluations in the SQP algorithm, the line search could be substituted with a moving limits methodology.

V. Conclusions

A numerical tool for optimal airfoil design has been described. The design tool uses a fully viscous Navier–Stokes solver to obtain the aerodynamic characteristics of airfoils. The design tool is used to compare several optimization algorithms by solving a minimum drag airfoil subject to a minimum lift requirement using three different optimization algorithms: the MMFD, the SLP method, and the SQP method. The SQP method outperformed the SLP and MMFD methods in computational efficiency. Therefore, of the three existing methods, the SQP method is the method recommended. Furthermore, the paper highlights that it could be possible to increase the efficiency of the SQP algorithm by substituting the line search with a moving limits methodology similar to the one used in the SLP algorithm. This would reduce the number of function evaluations in the SQP algorithm. The SLP method is also considered to be a potential candidate for shape optimization if an efficient method to compute the gradients is available. When using the SLP method, special care must be taken when selecting the initial move limits. Finally, the MMFD method is not recommended because it requires a large number of internal function evaluations that cannot be parallelized, making this method highly computationally inefficient.

Acknowledgments

This research was supported by a National Science and Engineering Research Council of Canada Discovery grant and an Advanced Systems Institute of BC Fellowship.

References

[1]Antoniou, A., and Lu, W.-S., *Optimization: Methods, Algorithms, and Applications*, Kluwer Academic, 2003.

[2]Arora, J., *Introduction to Optimum Design*, McGraw–Hill, New York, 1989.

[3]Gill, P., Murray, W., and Wright, M., *Practical Optimization*, Academic Press, New York, 1981.

[4]Vanderplaats, G., *Numerical Optimization Techniques for Engineering Design with Applications*, McGraw–Hill, New York, 1984.

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

[6]Schittowski, K., Zillober, C., and Zotemantel, R., "Numerical Comparison of Nonlinear Programming Algorithms for Structural Optimization," *Structural Optimization*, Vol. 7, No. 1, 1994, pp. 1–28.

[7]Ghattas, O., and Bark, J.-H., "Optimal Control of Two- and Three-Dimensional Incompressible Navier–Stokes Flows," *Journal of Computational Physics*, Vol. 136, No. 2, 1997, pp. 231–244.

[8]*DOT: Design Optimization Tools Users Guide*, ver. 5.0, Vanderplaats Research and Development, Inc., Colorado Springs, CO, 2001.

[9]Magagnato, F., *SPARC Manual (Structured PArallel Research Code)*, Dept. of Fluid Machinery, Univ. of Karlsruhe, Karlsruhe, Germany, 2000.

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

[11]Tsai, H. M., Wong, A. S. F., Cai, J., Zhu, Y., and Liu, F., "Unsteady Flow Calculations with a Parallel Multiblock Moving Mesh Algorithm," *AIAA Journal*, Vol. 39, No. 6, 2001, pp. 1021–1029.

[12]Martins, J., "A Coupled-Adjoint Method for High-Fidelity Aero-Structural Optimization," Ph.D. Dissertation, Aerospace Dept., Stanford Univ., Stanford, CA, Nov. 2002.

[13]Bischof, C., Carle, A., Corliss, G., Griewank, A., and Hovland, P., "ADIFOR—Generating Derivative Codes from Fortran Programs," *Scientific Programming*, Vol. 1, No. 1, 1992, pp. 1–29.

[14]Martins, J., Kroo, I., and Alonso, J., "An Automated Method for Sensitivity Analysis Using Complex Variables," AIAA Paper 2000-0689, Jan. 2000.

[15]*Portable Batch System User Guide*, Altair Grid Technologies, March 2003.

[16]Bazaraa, M. S., and Shetty, C., *Nonlinear Programming: Theory and Algorithms*, Wiley, New York, 1979.

[17]Abadie, J. (ed.), *Nonlinear Programming*, Wiley, New York, 1967.

[18]Thomas, H., Vanderplaats, G., and Shyy, Y.-K., "A Study of Move Limit Adjustment Strategies in the Approximation Concepts Approach to Structural Synthesis," *Proceeding of the 4th AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization*, 1992.

[19]Chen, T.-Y., "Calculation of the Move Limits for the Sequential Linear Programming Method," *International Journal for Numerical Methods in Engineering*, Vol. 36, No. 15, 1993, pp. 2661–2679.

[20]Lamberti, L., and Pappalettere, C.,"Move Limits Definition in Structural Optimization with Sequential Linear Programming. Part I," *Computers and Structures*, Vol. 81, No. 3, 2003, pp. 197–213.

[21]Lamberti, L., and Pappalettere, C., "Move Limits Definition in Structural Optimization with Sequential Linear Programming. Part II," *Computers and Structures*, Vol. 81, No. 3, 2003, pp. 214–238.

[22]Boggs, P., and Tolle, J., "Sequential Quadratic Programming," *Acta Numerica*, Vol. 4, 1995, pp. 1–51.

[23]Stockdill, B. T., "Turbulence Modelling of Unsteady Separated Flow over an Airfoil," M.S. Thesis, Dept. of Mechanical Engineering, Univ. of Victoria, Victoria, BC, Canada, Dec. 2003.

[24]Eppler, R., *Airfoil Design and Data*, Springer-Verlag, Berlin, 1990.

[25]Lissaman, P., "Low-Reynolds-Number Airfoils," *Annual Review of Fluid Mechanics*, Vol. 15, 1983, pp. 223–239.

A. Messac
Associate Editor