# A Newton-Krylov Approach for Aerodynamic Shape Optimization of Wings

Timothy M. Leung* and David W. Zingg†

*Institute for Aerospace Studies, University of Toronto, Toronto, Ontario, M3H 5T6, Canada*

**A Newton-Krylov algorithm is presented for aerodynamic shape optimization in three dimensions using the Euler equations. An inexact-Newton method is used in the flow solver, a discrete-adjoint method to compute the gradient, and a quasi-Newton method to find the optimum. The Krylov subspace method flexible generalized minimal residual is used with approximate-Schur preconditioning to solve both the flow equation and the adjoint equation in a parallel computing environment. The wing geometry is parameterized by a B-spline control net, and a fast algebraic algorithm is used for grid movement. The discrete-adjoint gradient can be obtained in approximately one-fourth the time required for a converged flow solution. The accuracy of the gradient is compared against finite differencing and is found to be comparably accurate. A single-point test case is presented for a cruise configuration optimization at transonic speed. This example as well as an inverse design demonstrate that the optimizer is able to decrease the objective function and gradient by several orders of magnitude efficiently for problems with over 170 design variables.**

## I. Introduction

Over the past several decades, computational fluid dynamics (CFD) has evolved to become an integral tool in the design cycle. Constant improvements in computer technology, especially in parallel architectures, help fuel interest in solving larger and more complex CFD problems. State-of-the-art flow solvers can accurately find solutions for viscous turbulent flows in less than one minute for 2D problems on a desktop computer, while 3D flows over an entire aircraft can be solved in a matter of hours using parallel computers.[1,2]

The development of CFD naturally leads to its incorporation into automated aerodynamic design optimization tools. Such tools are useful in continuous refinement of the dominant "wing-fuselage" configuration. But in the long term, they can also serve as an inexpensive numerical testbed for more unconventional configurations where empirical data may be lacking. However, at the moment, the use of aerodynamic design optimization in large-scale 3D problems is still limited. For a gradient-based optimizer, hundreds of flow solves and gradient evaluations may be required. Difficulties may also arise from noise in the design space due to the high number of design variables, and from defining and modifying complex shapes and meshes during the optimization cycle.

For gradient-based optimization, the main focus has been on fast and accurate computation of the gradient. Finite-difference gradients used in early research[3,4] were too impractical to handle large numbers of design variables, and suffer from round-off errors when step sizes are too small. One major advance in this area is the development of the adjoint method, developed independently by Pironneau[5] and Jameson.[6] Adjoint methods are further divided into continuous[7–12] and discrete[13–21] approaches. Pros and cons of the two approaches have been discussed extensively,[22,23] and both have been implemented successfully in aerodynamic design optimization. The main advantage of the adjoint method is that the time required for each gradient computation is nearly independent of the number of design variables. A gradient-based optimizer is now an inexpensive alternative to optimizers using genetic algorithms. For a comparison between these two classes of optimizers, see Zingg *et al.*[24]

[*] PhD Candidate

[†] Professor and Director, Tier 1 Canada Research Chair in Computational Aerodynamics, Associate Fellow AIAA

In this paper, we present our work on a robust and efficient gradient-based algorithm for aerodynamic design in 3D. Our algorithm is based on the Newton-Krylov approach of Nemec and Zingg.[16,17] The steady-state flow solution is computed using the inexact-Newton method, with the Krylov subspace method flexible generalized minimal residual (FGMRES)[25,26] solving the linear system at each outer iteration. The optimizer is based on the quasi-Newton optimizer BFGS (Broyden-Fletcher-Goldfarb-Shanno),[27] and the adjoint equation is again solved by FGMRES. The surface geometry is parameterized by a B-spline control net, which provides the design variables. We present results on the accuracy of the adjoint gradient, as well as single-point results for cruise configuration optimization at transonic speed. Our objective is to demonstrate the viability and efficiency of the present approach for large-scale aerodynamic shape optimization.

## II. Problem Formulation

The goal of aerodynamic design optimization is to find a set of design variables $X$ such that a scalar objective function $\mathcal{J}$ is minimized:

$$\min_{X} \mathcal{J}(X, Q) \tag{1}$$

In this case, $Q$ represents the flow solution. To ensure that the optimizer yields a physically realistic shape, a set of inequality constraints may be imposed. For this research, only geometric constraints that are functions of design variables are considered, thus

$$C_j(X) \leq 0 \tag{2}$$

In addition, the discrete flow equation must also be satisfied:

$$R(X, Q) = 0 \tag{3}$$

### A. Objective Functions

For aerodynamic shape optimization problems, the objective functions are based on performance measures, such as lift and drag coefficients ($C_L$, $C_D$). In our present work, we consider lift-constrained drag minimization and maximization of lift-to-drag ratio as our objectives. For lift-constrained drag minimization problems, we employ the equation:

$$\mathcal{J} = \begin{cases} \omega_L \left(1 - \frac{C_L}{C_L^*}\right)^2 + \omega_D \left(1 - \frac{C_D}{C_D^*}\right)^2 & \text{if } C_D > C_D^* \\ \omega_L \left(1 - \frac{C_L}{C_L^*}\right)^2 \text{otherwise} \end{cases} \tag{4}$$

Target lift and drag coefficients $C_L^*$, $C_D^*$, and weights $\omega_L$, $\omega_D$ are parameters supplied by the user.

For maximizing the lift-to-drag ratio, we use the objective function:

$$\mathcal{J} = \frac{C_D}{C_L} \tag{5}$$

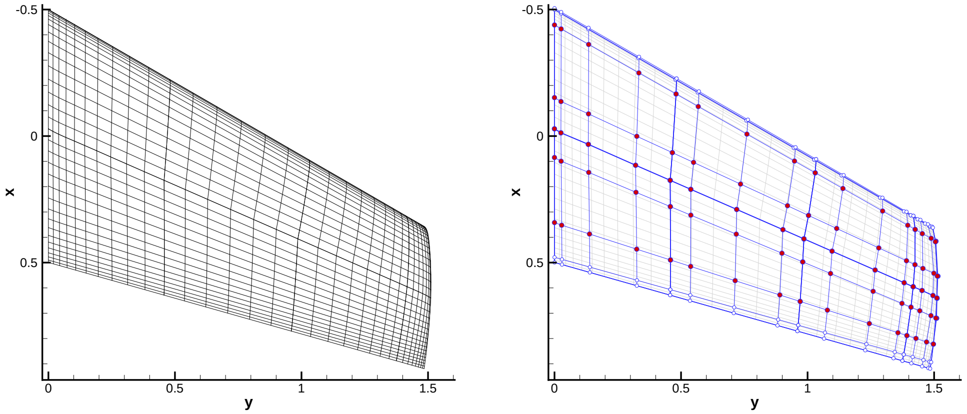
Finally, we also consider inverse design as a validation tool. The objective function for an inverse design problem is:

$$\mathcal{J} = \frac{1}{2} \sum_{1}^{N_p} \left[(C_p)_i - (C_p^*)_i\right]^2 \tag{6}$$

In this case, the user specifies a known target pressure distribution on each node $(C_p^*)_i$, and the optimizer recovers the shape that minimizes the least-square error. Inverse design is useful as a validation tool for the optimizer.

### B. Geometry Parameterization and Design Variables

The geometry of the wing is parameterized using a CAD-free geometry control system based on Fudge *et al*,[28] where the wing surface is represented by a B-spline control net. Fig. 1 shows the parameterization of the top surface of a ONERA M6 wing. In this example, the wing is represented using fourth-order B-spline control points, with 9 control points in the spanwise direction, and 17 in the chordwise direction. Control



(a) Surface nodes top view

(b) B-spline control net top view

**Figure 1. An example of B-spline parameterization of an ONERA M6 wing.**

points can be grouped together so that they can be described by planform variables, such as span, chord, sweep, dihedral and twist. Furthermore, the location of each individual B-spline control point can also be a design variable. The latter is necessary to change the wing cross-section. For example, the control points that are highlighted in red in Fig. 1 are selected as design variables for our test cases. Increasing the number of control points improves the flexibility of the parameterization, and allows the designer more control over changes in the geometry.

## C. Governing Equation

The governing equation for the optimization is the Euler equations. In transformed coordinates $(\xi_1, \xi_2, \xi_3)$, they can be expressed in the form:

$$\partial_t Q + \partial_{\xi_i} E_i = 0 \tag{7}$$

where

$$Q = \frac{1}{J}\begin{pmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ e \end{pmatrix}, \qquad \text{and} \qquad E_i = \frac{1}{J}\begin{pmatrix} \rho U_i \\ \rho u_1 U_i + p\partial_x \xi_i \\ \rho u_2 U_i + p\partial_y \xi_i \\ \rho u_3 U_i + p\partial_z \xi_i \\ (e+p)U_i \end{pmatrix}.$$

The scalar $J$ denotes the Jacobian of the mapping from physical space to computational space, and the $U_i = u_j \partial_{x_j} \xi_i$ are contravariant velocities.

The flow equations are discretized for a multi-block structured grid. In our parallel strategy, each block in the grid and the component of $Q$ is distributed to separate processors. As such, the discretization of Eq. 7 is done in parallel in each block. Second-order centered differencing is used in interior nodes, and first-order one-sided differencing is used at block boundaries. For numerical stability, we use the JST scalar dissipation model,[29,30] with second-difference dissipation near shocks and fourth-difference dissipation everywhere else. The blocks are coupled using simultaneous approximation terms (SATs). Details of the implementation of SATs can be found in Hicken and Zingg.[31]

## D. Geometric Constraints

We have implemented two geometric constraints: the volume enclosed by the geometry, and its surface area. The constraints are expressed as penalty terms in the objective function. The volume penalty term is added when the current volume $V$ deviates from the initial volume $V_0$ by more than a prescribed factor $v_f$:

$$|V - V_0| > v_f V_0 \tag{8}$$

Then the volume penalty equation is specified as a quadratic function:

$$\mathcal{J}_p = \frac{1}{2}\omega_V \left(|V - V_0| - v_f V_0\right)^2 \tag{9}$$

The penalty weight $\omega_V$ is user-supplied. Surface area constraints are implemented in a similar way, when the surface area $A$ deviates from the initial surface $A_0$ by more than a factor $a_f$:

$$\mathcal{J}_p = \frac{1}{2}\omega_A \left(|A - A_0| - a_f A_0\right)^2 \tag{10}$$

Again, $\omega_A$ and $a_f$ are supplied by the user. Both penalty terms are added to the objective function:

$$\mathcal{J} = \mathcal{J}_0 + \sum_i \mathcal{J}_{pi} \tag{11}$$

By casting the constraints as penalty terms, our original optimization problem (Eq. 1) becomes an unconstrained problem.

## III. Numerical Method

### A. Flow Solver

Discretization of the Euler equations produces a set of nonlinear algebraic equations. At steady-state, they can be represented by the equation

$$R(Q) = 0 \tag{12}$$

which is the same flow constraint equation in Eq. 3. In order to find this steady solution, we start with an initial guess $Q_0$ based on freestream properties and apply the Newton method, solving a linear system in the form:

$$\left(\frac{\partial R}{\partial Q}\right)^{(n)} \Delta Q^{(n)} = -R(Q^{(n)}) \tag{13}$$

and updating $Q$ until the norm of $R(Q)$ is reduced to machine zero. In practice, reduction of 10 orders of magnitude is sufficient to achieve a converged solution. Eq. 13 is solved using the Krylov subspace iterative method FGMRES.[26] When a Krylov method is used to solve the linear system, only a matrix-vector product with the flow Jacobian is required, and this can be approximated by a Frechet derivative (one-sided differencing):

$$\frac{\partial R}{\partial Q} v \approx \frac{R(Q + \epsilon v) - R(Q)}{\epsilon} \tag{14}$$

leading to a Jacobian-free approach for the flow solver. The flow Jacobian–or more accurately, its transpose–is therefore only needed for the adjoint equation (see Section B) in the optimizer. The matrix is derived by hand linearization of $R(Q)$. The linearization of the artificial dissipation is performed by freezing the coefficients. A complex step is used to linearize the SATs at block boundaries. Our approach is similar to Nielsen and Kleb,[32] but our application of the complex step method is limited to the SATs. To improve the convergence of FGMRES, the linear system in Eq. 13 is right-preconditioned. The preconditioner uses a first-order Jacobian matrix that combines the second- and fourth-difference dissipation terms:

$$\hat{\kappa}_2 = \kappa_2 + \sigma \kappa_4 \tag{15}$$

A value of $4 \leq \sigma \leq 6$ is recommended based on previous work.[33] An approximate-Schur procedure based on Saad and Sosonkina[34] is applied to the first-order Jacobian to form the preconditioner. Details of the implementation can be found in Hicken and Zingg.[31]

The Newton method converges quadratically when $Q$ is sufficiently close to the solution. However, during the start-up phase, when the iterate is far from the solution, convergence may not be possible. Therefore, for stability during the start-up period, the flow solver uses an approximate-Newton method, where the first-order Jacobian replaces the full Jacobian in Eq 13. A spatially varying ime step is also added in during the start-up phase to add diagonal dominance:

$$\Delta t_i^{(n)} = \frac{\Delta t_{\text{ref}}^{(n)}}{J_i(1 + \sqrt[3]{J_i})} \tag{16}$$

where the reference time step for iteration $n$ is defined as

$$\Delta t_{\text{ref}}^{(n)} = A(B)^n \tag{17}$$

Values of $A = 0.1$ and $B = 1.5$ are used in the present work.

The flow solver switches to the Newton method when the normalized residual has dropped below a threshold $\tau$, ie:

$$\frac{||R^{(n)}||_2}{||R^{(0)}||_2} \equiv \mathcal{R}^{(n)} < \tau \tag{18}$$

For inviscid flows, $\tau = 0.1$ is usually sufficient. During the Newton phase, the reference time step is based on Mulder and van Leer:[35]

$$\Delta t_{\text{ref}}^{(n)} = \max\left[\alpha\left(\mathcal{R}^{(n)}\right)^{-\beta}, \Delta t_{\text{ref}}^{(n-1)}\right] \tag{19}$$

with $\alpha = 1.0$ and $\beta = 1.75$. As the residual decreases, the time step approaches infinity, and the full Newton step is recovered.

## B. Adjoint Solver

At the heart of any gradient-based optimization is the fast and accurate evaluation of the objective function gradient $\mathcal{G} = dJ/dX$. The gradient can be expressed as:

$$\mathcal{G} = \frac{\partial \mathcal{J}}{\partial X} - \psi^T \frac{\partial R}{\partial X} \tag{20}$$

where the vector $\psi$ is the adjoint variable. The adjoint variables are obtained from solving the adjoint equation:

$$\left(\frac{\partial R}{\partial Q}\right)^T \psi = \frac{\partial \mathcal{J}}{\partial Q}^T \tag{21}$$

It should be noted that Eq. 21 is independent of the design variables. The adjoint system is solved by adapting the strategy used for the flow solution. The adjoint system is solved using the Krylov subspace method FGMRES as the iterative solver. The system is right-preconditioned using the approximate-Schur preconditioner. We specify a linear tolerance of $10^{-8}$ for the adjoint system. Note that the left-hand-side of the adjoint equation differs from Eq. 13 by a transpose. As a result, the matrix-free approach used in the flow solver cannot be used, and the matrix must be formed explicitly. However, the matrix is not explicitly transposed in the parallel environment. Instead, transpose operations on the Jacobian and preconditioner in FGMRES are computed in parallel. The right-hand-side term $\partial \mathcal{J}/\partial Q$ is evaluated analytically for each objective function. Finally, the partial derivatives with respect to design variables $\partial \mathcal{J}/\partial X$, $\partial R/\partial X$ are evaluated using second-order centered differencing.

## C. Optimizer

When the geometric constraints are implemented as quadratic penalty terms, we can now consider the optimization as an unconstrained problem. The BFGS quasi-Newton optimizer[27] is used. In BFGS, the inverse of the Hessian matrix is approximated to get the search direction. BFGS guarantees that the search direction is a descent direction. A line-search algorithm with backtracking is used to satisfy the strong Wolfe conditions. In the line-search algorithm, a cubic interpolant is constructed for the function

$$\phi(\alpha) \equiv \mathcal{J}(X_n + \alpha p_n) \tag{22}$$

where $p_n$ is the search direction, and $\alpha$ is the step size along that direction. The minimum of the interpolant is either at its endpoint or in the interior, which can be found by the line-search iteration until the strong Wolfe conditions are satisfied:

$$\alpha_{k+1} = \alpha_k - (\alpha_k - \alpha_{k-1})\left[\frac{\phi'(\alpha_k) + r_2 - r_1}{\phi'(\alpha_k) - \phi'(\alpha_{k-1}) + 2r_2}\right] \tag{23}$$

where

$$r_1 = \phi'(\alpha_{k-1}) + \phi'(\alpha_k) - 3\frac{\phi(\alpha_{k-1}) - \phi(\alpha_k)}{\alpha_{k-1} - \alpha_k} \tag{24}$$

$$r_2 = \sqrt{r_1^2 - \phi'(\alpha_{k-1})\phi'(\alpha_k)}$$

In the event that the line search algorithm stalls, the optimizer is re-started from the steepest descent direction $-\mathcal{G}$. The optimizer is considered converged when $||\mathcal{G}||_2$ falls below a user-defined tolerance. However, for many applications, it is often impractical to reduce the gradient by more than one or two orders of magnitude.

## D. Variable Scaling

The scaling of the design variables is crucial to the performance of the optimizer. A problem is considered poorly scaled if changes in one variable produces much larger variations in the value of the objective function than other variables. This problem arises when we mix B-spline design variables with planform variables. In our experience, $\Delta X_i$ from initial to optimized geometry range from $10^{-5}$ to $10^{-3}$ for B-spline variables, to $10^0$ for angle-of-attack and planform variables. We scale the design variables by their initial values:

$$X_s = L^{-1}X \tag{25}$$

where the entries in $X_s$ are the scaled design variables, those in $X$ are the unscaled design variables, and the diagonal matrix $L$ contains the initial unscaled design variables.

## E. Grid Movement Strategy

A high-quality computational grid is necessary to compute the flow solution at each design iteration, and also to evaluate the partial derivatives $\partial\mathcal{J}/\partial X$, $\partial R/\partial X$ in Eq. 20. For aerodynamic shape optimization using structured grids, expensive grid re-generation can be avoided by employing a suitable grid movement algorithm. Each time the wing surface changes, the grid is adjusted accordingly. For the current work, a fast algebraic grid movement method is used. The movement of the nodes $k = 2$ to $k_{max}$ along a normal grid line is determined by the algebraic equation:

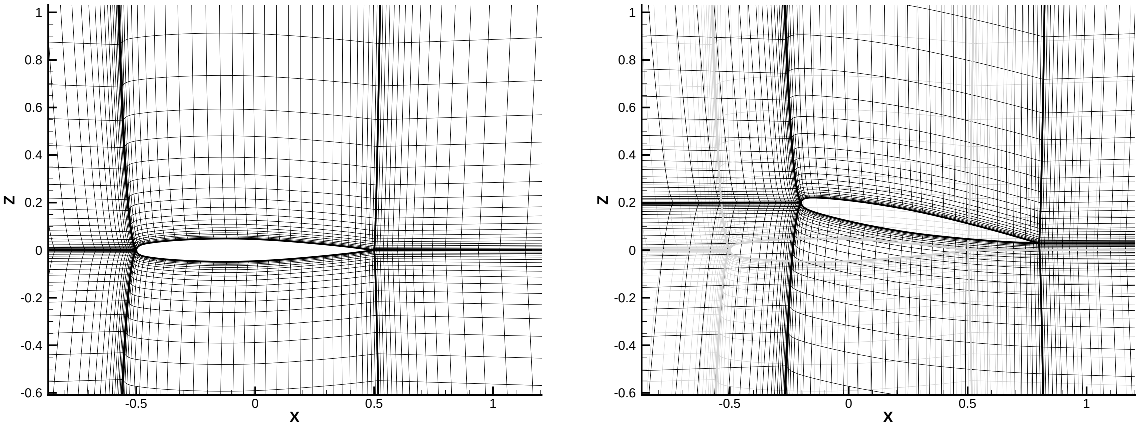
$$\mathbf{x}_k^{\text{new}} = \mathbf{x}_k^{\text{old}} + \frac{\Delta\mathbf{x}_1}{2}\left[1 + \cos\left(\pi S_k\right)\right] \tag{26}$$

where $\Delta\mathbf{x}_1$ is the displacement of the surface node, and

$$S_k = \frac{\sum_{i=2}^{k} |\mathbf{x}_i - \mathbf{x}_{i-1}|}{\sum_{i=2}^{k_{max}} |\mathbf{x}_i - \mathbf{x}_{i-1}|} \tag{27}$$

is the normalized arclength distance along the grid line.

An example of the grid movement algorithm is shown in Fig. 2. In this example, the root location of an ONERA M6 wing is displaced, and the wing is rotated to a positive angle of attack. The grid at the symmetry plane is shown.



ONERA M6 grid at symmetry plane

Perturbed grid at symmetry plane

**Figure 2. Example of algebraic grid movement algorithm.**

# IV. Results and Discussion

We obtain flow solver and optimization results using a distributed-memory Beowulf-class cluster. The cluster uses Intel Itanium 2 processors with a CPU speed of 1.5GHz. Each computation node consists of 4 processors, with 8GB of shared memory per node. The nodes are connected by a high-bandwidth low-latency Myrinet network. Communication between processors is done using the message passing library MPICH.

For the inverse design and optimization cases, a 431,000-node grid over an ONERA M6 wing (Fig. 3) is used. The grid has an H-H topology with 48 blocks. For gradient accuracy evaluations, a coarse ONERA M6 grid with 146,000 nodes is used. This grid is too coarse to accurately capture the physics of the flow field, but it can converge to machine zero quickly to verify the accuracy of the gradient.

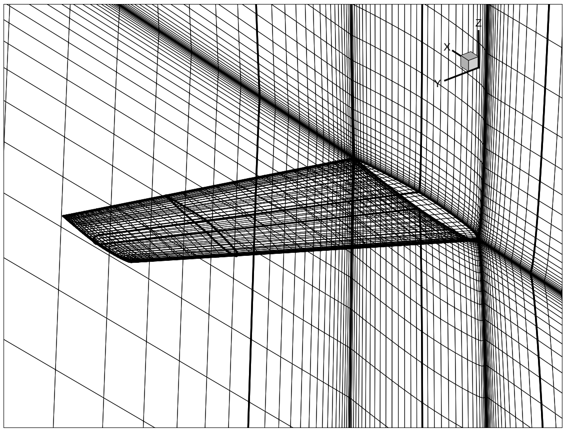


**Figure 3. H-H grid over an ONERA M6 wing used for inverse design and optimization cases.**

## A. Gradient Accuracy Evaluation

The factor that most affects the accuracy of the adjoint gradient is the linearization of the residual vector $R(Q)$ to obtain the flow Jacobian $\partial R/\partial Q$. However, exact linearization of non-differentiable functions used in the artificial dissipation terms poses a significant challenge. For the current tests, the second-difference dissipation is turned off (ie: $\kappa_2 = 0$), and the fourth-difference terms are linearized exactly in the flow Jacobian.

Discrete adjoint gradients are compared to finite-difference gradients. The finite-difference gradient is obtained using second-order centered-differencing, with a step size of $\epsilon_i = 10^{-5} \times X_i$ for each design variable. The same step size is also used for evaluating $\partial \mathcal{J}/\partial X$ and $\partial R/\partial X$ in the adjoint method for geometric variables. For angle-of-attack design variables, $\partial \mathcal{J}/\partial \alpha$ is derived analytically, while $\partial R/\partial \alpha$ is computed using a complex step with $\epsilon = 10^{-20}$.

For the first case, a freestream Mach number of $M_\infty = 0.3$ with an angle of attack of $\alpha = 5.0°$ is used. The minimization of $C_D/C_L$ (Eq. 5) is used as the objective function. The design variables are: change in sweep angle $\Delta\Lambda$ from original configuration, 3 arbitrary B-spline control points, and the angle of attack. The gradient components using the two methods are summarized in Table 1. The second case has a freestream Mach number of $M_\infty = 0.84$ with an angle of attack of $\alpha = 3.0°$. The lift-constrained drag minimization objective (Eq. 4) is used, with the targets $C_L^* = 0.35$, $C_D^* = 0.01$ and weights $\omega_L = 100.0$ and $\omega_D = 1.0$ for lift and drag respectively. The same design variables are used in the transonic test case. The results are shown in Table 2. In both test cases, the adjoint gradient agrees well with the finite-difference gradient.

## B. Inverse Design

The goal of the inverse design is to find a wing geometry that best matches a target pressure distribution (Eq. 6). In this case, our target pressure is based on the surface pressure of the ONERA M6 wing at $M_\infty = 0.80$ at an angle of attack of $\alpha = 3.0°$. We begin with a wing that has the same planform as the ONERA M6, but with a different cross section. The initial angle of attack was also perturbed to 3.5°. The

| Design Variable | Finite-Difference | Adjoint | Relative Diff. |
|---|---|---|---|
| $\Delta\Lambda$ | 6.7066465E-3 | 6.7131130E-3 | 0.09% |
| B-spline 1 | 1.4214947E-3 | 1.4228262E-3 | 0.09% |
| B-spline 2 | 3.7034843E-3 | 3.7012754E-3 | 0.06% |
| B-spline 3 | 7.8118853E-4 | 7.8110487E-4 | 0.01% |
| $\alpha$ | 8.4637219E-3 | 8.4634355E-3 | 0.003% |

**Table 1. Gradient comparison for the first case.**

| Design Variable | Finite-Difference | Adjoint | Relative Diff. |
|---|---|---|---|
| $\Delta\Lambda$ | 2.2621572 | 2.2641458 | 0.087% |
| B-spline 1 | 1.7330761 | 1.7333275 | 0.015% |
| B-spline 2 | 1.8041299 | 1.8038633 | 0.015% |
| B-spline 3 | 3.5786079 | 3.5786861 | 0.002% |
| $\alpha$ | 3.7008111 | 3.7008132 | 0.0003% |

**Table 2. Gradient comparison for the second case.**

initial wing cross section was arbitrarily created. Design variables are the $z$-coordinates of 165 B-spline control points, previously described in Fig. 1, and the angle of attack.

Convergence history is shown in Fig. 4. After 600 iterations, the objective function has decreased by seven orders of magnitude. The gradient $L_2$-norm has also decreased by five orders of magnitude. This is excellent convergence behaviour for such a large number of design variables. The initial and final geometries are compared at various spanwise stations in Fig. 5. The converged geometry is indistinguishable from the ONERA M6 wing.

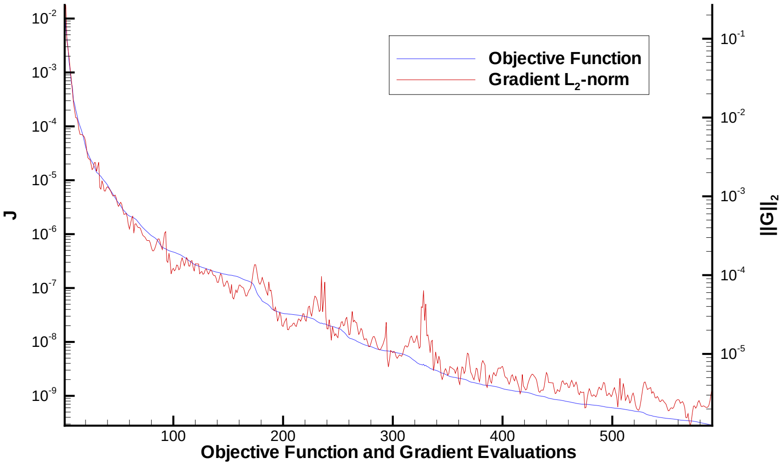


**Figure 4. Convergence history for the inverse design case.**

## C. Single-Point Wing Optimization

We present results from a single-point optimization of a wing at a transonic speed. The goal of this optimization case is to minimize drag at $M = 0.90$ while maintaining the lift coefficient of the original geometry. The ONERA M6 wing initially operates at $\alpha = 2.50°$. At this operating condition, the lift and drag coefficients are $C_L = 0.350$ and $C_D = 0.0320$ respectively. The lift-constrained drag minimization objective function

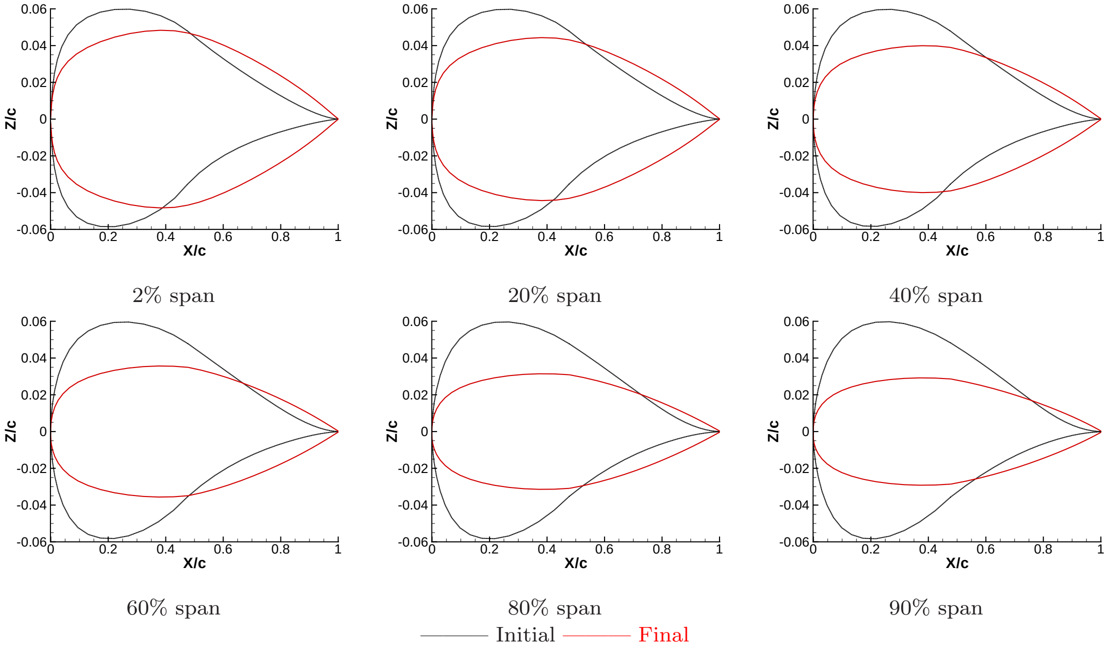


**Figure 5. Comparison of initial and final (ONERA M6) wing cross sections at various spanwise stations.**

(Eq. 4) is used for this case, with a targets in lift and drag specified as:

$$\begin{aligned} C_L^* &= 0.351 \\ C_D^* &= 0.0085 \end{aligned}$$

Weights on lift and drag are $\omega_L = 20.0$ and $\omega_D = 1.0$ respectively. The high value on $\omega_L$ is found to be necessary for the optimizer to maintain the lift coefficient.

For the optimization case, the design variables are the $z$-coordinates of 165 B-spline control points, the angle of attack, the change in the leading-edge sweep angle at the wing root, the changes in two sweep angles near the wing tip, chord, span, twist and dihedral, for a total of 173 design variables. The sweep angle is measured as the difference between the optimized geometry and the original ONERA M6 wing. As the sweep angle changes, the wing is sheared along the chordwise direction, thus the planform area remains constant throughout the optimization cycle. A volume constraint (Eq. 9) was added to maintain the optimized wing's volume to within 2% of the original wing. A penalty weight of $\omega_V = 50.0$ was used for this case.

During the optimization cycle, a flow solve requires an average of 8.2 minutes to reduce the residual by 10 orders of magnitude, using 48 processors. The adjoint solver takes an average of 2.2 minutes to reduce the residual by eight orders of magnitude. That the flow solution takes about 4 times longer to solve than the adjoint solution is consistent with previous experience with our 2D adjoint solver.

The convergence history is shown in Figure 6. After 120 iterations, the objective function is reduced by 12 orders of magnitude and the gradient norm by over six. At this point, the lift and drag values are:

$$C_L = 0.3510 \quad \text{and} \quad C_D = 0.00850$$

in good agreement with the targets. This representing 73.4% improvement in drag. In the final geometry, the wing span increased by 30.0%, the sweep angle $\Lambda_{25\%}$ was increased slightly from 26.7° to 29.19°. The angle of attack was also increased 2.50° to 3.0565°. A small twist angle of $-7.35°$ was added. Finally, a small dihedral angle of $-2.41°$ was added. Mach contours and pressure coefficients on the surface of the optimized wings are compared to the original ONERA M6 wing in Fig. 7. The optimizer has eliminated the wave drag by removing the shock, and furthermore, since the drag lies vary close to the theoretical drag for an elliptical load distribution, the induced drag has been reduced as well. The optimized wing's cross sections are compared to the ONERA M6 wing in Fig. 8.

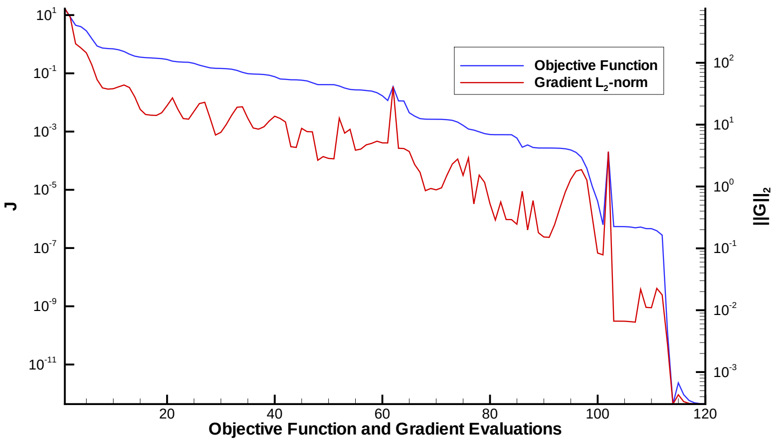


**Figure 6. Convergence history for the single-point optimization case.**

## V. Conclusion

An efficient Newto-Krylov algorithm for aerodynamic shape optimization in three dimensions is presented. Our discrete gradient calculations are found to be accurate compared to finite differencing, and when a fast algebraic grid movement algorithm is used, the gradient calculation time can be independent of the number of design variables. A B-spline control net allows us to parameterize the wing geometry with a high degree of flexibility. A test case is presented for the optimization of a cruise configuration at transonic speed. Our algorithm is found to perform well with the combined number of B-spline and planform design variables exceeding 170. This means that the optimizer can efficiently optimize the planform as well as the cross section of the wing. Further research will focus on multi-point optimization, as well as maximizing the range and endurance.

## Acknowledgments

This research was supported with funding through the University of Toronto, Kenneth M. Molson Fellowship, Ontario Graduate Scholarship for Science and Technology (OGSST) from the Government of Ontario. The authors gratefully acknowledge Jason Hicken for his assistance and discussions.

## References

[1] May, G., van der Weide, E., Jameson, A., and Martinelli, L., "Drag prediction of the DLR-F6 configuration," AIAA Paper 2004-0396, January 2004.

[2] Lee-Raush, E. M., Frink, N. T., Mavriplis, D. J., Rausch, R. D., and Milholen, W. E., "Drag prediction on a DLR-F6 transport configuration using unstructured grid solvers," AIAA Paper 2004-0000, January 2004.

[3] Hicks, R. M., Murman, E. M., and Vanderplaats, G. N., "An assessment of airfoil design by numerical optimization," NASA TM X-3092, NASA, July 1974.

[4] Hicks, R. M. and Henne, P. A., "Wing design by numerical optimization," *Journal of Aircraft*, Vol. 15, 1978, pp. 407–412.

[5] Pironneau, O., *Optimal Shape Design for Elliptic Systems*, Springer-Verlag, 1983.

[6] Jameson, A., "Aerodynamic design via control theory," *J. of Sci. Comp.*, Vol. 3, 1988, pp. 233–260.

[7] Anderson, W. K. and Venkatakrishnan, V., "Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation," AIAA Paper 97-0643, 1997.

[8] Jameson, A., Martinelli, L., and Pierce, N., "Optimum aerodynamic design using the Navier-Stokes equations," *Theoretical Fluid Dynamics*, Vol. 10, 1998, pp. 213–237.

[9] Reuther, J. J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, Part 1," *Journal of Aircraft*, Vol. 26, No. 1, 1999, pp. 51–60.

[10] Reuther, J. J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, Part 2," *Journal of Aircraft*, Vol. 26, No. 1, 1999, pp. 61–74.

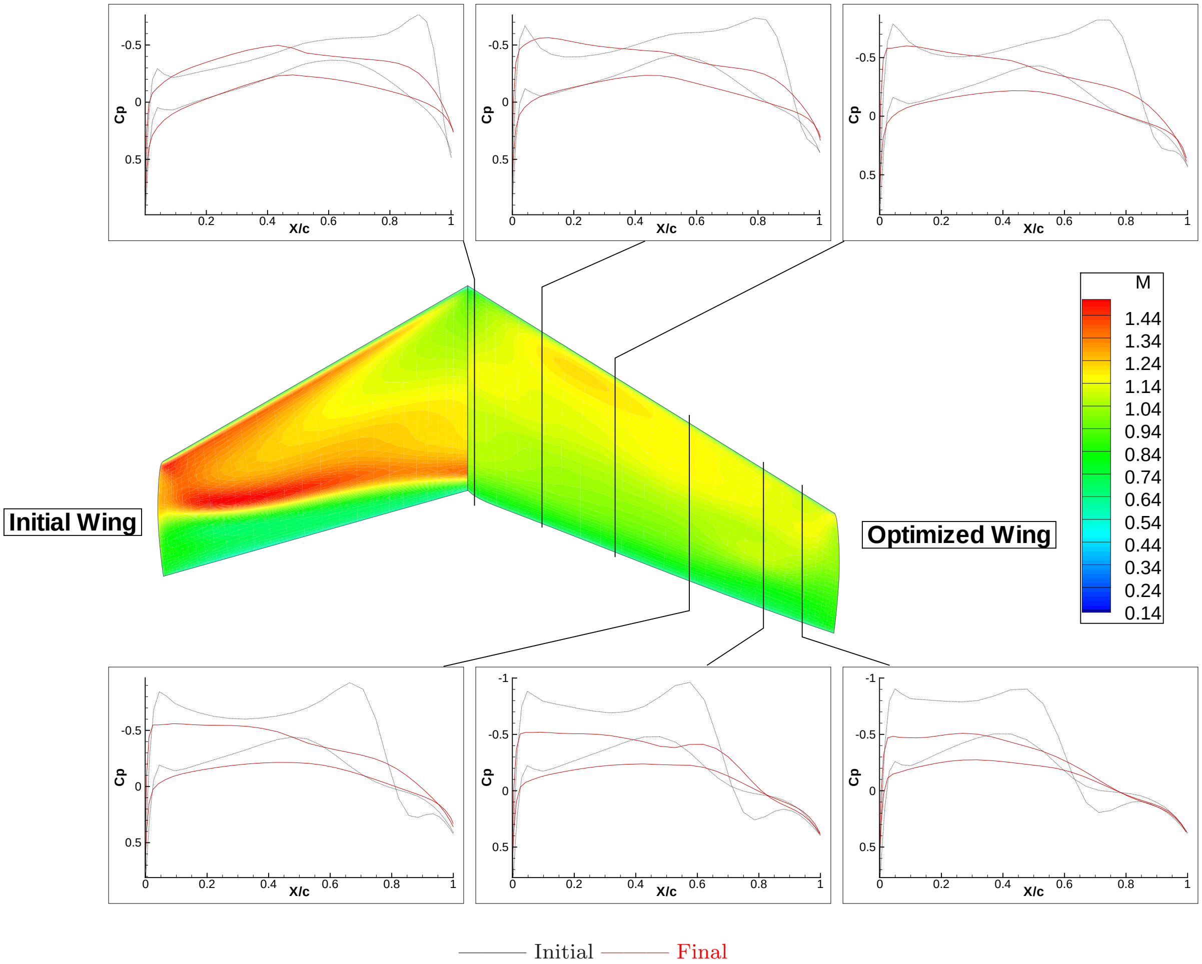


**Figure 7. Comparison of Mach contours and pressure distributions at various spanwise stations.**

[11] Jameson, A., Sririam, Martinelli, L., and Haimes, B., "Aerodynamic shape optimization of complete aircraft configurations using a unstructured grids," AIAA Paper 2004-0533, 2004.

[12] Leoviriakit, K. and Jameson, A., "Multi-point wing planform optimization via control theory," AIAA Paper 2005-0450, Reno, NV, 2005.

[13] Nielsen, E. J. and Anderson, W. K., "Recent improvements in aerodynamic design optimization on unstructured meshes," *AIAA Journal*, Vol. 40, No. 6, June 2002, pp. 1155–1163.

[14] Nadarajah, S. K., Jameson, A., and Alonso, J. J., "Sonic boom reduction using an adjoint method for wing-body configurations in supersonic flow," AIAA Paper 2002-5547, 2002.

[15] Nadarajah, S. K., *The Discrete Adjoint Approach to Aerodynamic Shape Optimization*, Ph.D. thesis, Stanford University, 2003.

[16] Nemec, M. and Zingg, D. W., "Newton-Krylov algorithm for aerodynamic design using the Navier-Stokes equations," *AIAA Journal*, Vol. 40, No. 6, June 2002, pp. 1146–1154.

[17] Nemec, M., Zingg, D. W., and Pulliam, T. H., "Multipoint and multi-objective aerodynamic shape optimization," *AIAA Journal*, Vol. 42, No. 6, 2004, pp. 1057–1065.

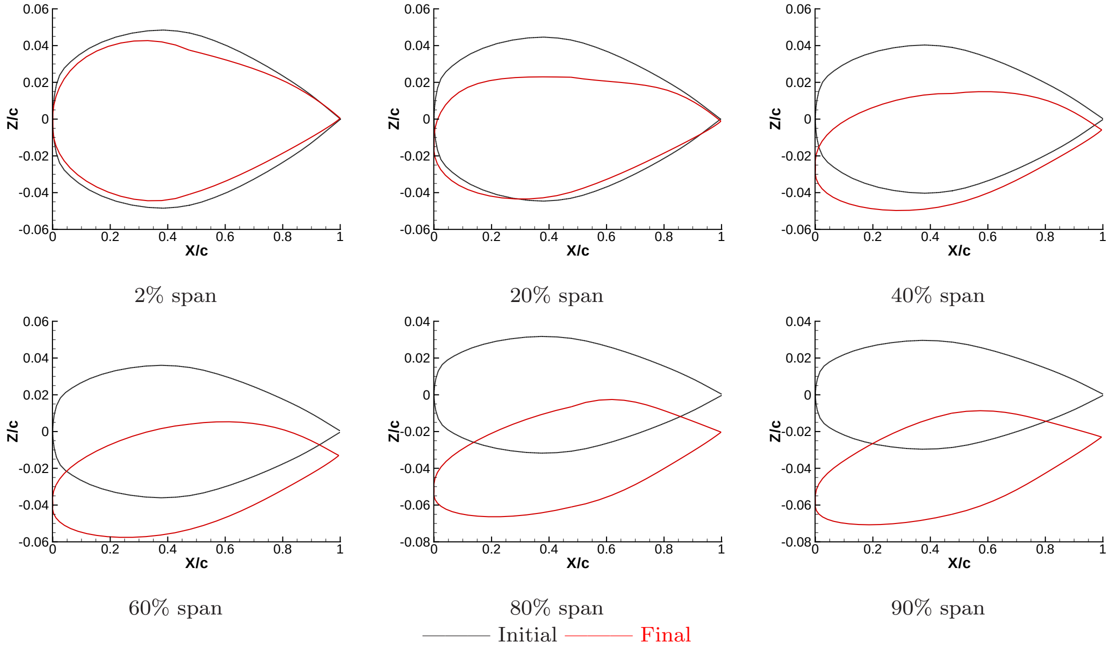
[18] Nemec, M. and Aftosmis, M., "Adjoint agorithm for CAD-Based shape optimization using a Cartesian method," AIAA Paper 2005-4987, June 2005.

[19] Mavriplis, D., "Formulation and multigrid solution of the discrete adjoint for optimization problems on instructured meshes," AIAA Paper 2005-0319, 2005.

[20] Mavriplis, D., "A discrete adjoint for optimization problems on three-dimensional unstructured meshes," AIAA Paper 2006-50, 2006.

[21] Carpentieri, G., van Tooren, M. J. L., and Koren, B., "Improving the efficiency of aerodynamic shape optimization on unstructured meshes," AIAA Paper 2006-298, Reno, NV, 2006.

[22] Nadarajah, S. and Jameson, A., "A Comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization," AIAA Paper 2000-0667, 2000.



2% span

20% span

40% span

60% span

80% span

90% span

**Figure 8. Comparison of wing cross sections at various spanwise stations.**

[23] Giles, M. B. and Pierce, N. A., "In introduction to the adjoint approach design," *Flow, Turbulence and Combustion*, Vol. 65, 2000, pp. 393–415.

[24] Zingg, D., Nemec, M., and Pulliam, T., "A comparative evaluation of genetic and gradient-based algorithms applied to aerodynamic optimization," *Revue Européenne de Mécanique Numérique*, Vol. 17, 2008, pp. 103–126.

[25] Saad, Y. and Schultz, M. H., "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear problems," *SIAM J. Sci. Stat. Comp.*, Vol. 7, 1986, pp. 856–869.

[26] Saad, Y., *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, PA, 2nd ed., 2003.

[27] Nocedal, J. and Wright, S. J., *Numerical Optimization*, Springer-Verlag, 1999.

[28] Fudge, D., Zingg, D. W., and Haimes, R., "A CAD-free and a CAD-based geometry control system for aerodynamic shape optimization," AIAA Paper 2005–0451, January 2005.

[29] Jameson, A., Schmidt, W., and Turkel, E., "Numerical solution of the Euler equations by finite volume methods using Runge-Kutta time-stepping schemes," AIAA Paper 81–1259, 14th Fluid and Plasma Dynamics Conference, Palo Alto, CA, 1981.

[30] Pulliam, T. H., "Efficient solution methods for the Navier-Stokes equations," 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.

[31] Hicken, J. E. and Zingg, D. W., "A parallel Newton-Krylov flow solver for the Euler equations on multi-block grids," AIAA Paper 2007–4333, 18th AIAA Computational Fluid Dynamics Conference, Miami, FL, June 2007.

[32] Nielsen, E. J. and Kleb, B., "Efficient construction of discrete adjoint operators on unstructured grids by using complex variables," AIAA Paper 2005-0324, Reno, NV, 2005.

[33] Nichols, J. and Zingg, D. W., "A three-dimensional multi-block Newton-Krylov flow solver for the Euler equations," AIAA Paper 2005-5230, 2005.

[34] Saad, Y. and Sosonkina, M., "Distributed Schur complement techniques for general sparse linear systems," *SIAM Journal of Scientific Computing*, Vol. 21, No. 4, 1999, pp. 1337–1357.

[35] Mulder, W. A. and van Leer, B., "Experiments with implicit upwind methods for the Euler equations," *Journal of Computational Physics*, Vol. 59, 1985, pp. 232–246.