https://ntrs.nasa.gov/search.jsp?R=20050185556 2019-08-29T20:20:25+00:00Z

Adjoint Algorithm for CAD-Based Shape Optimization Using a Cartesian Method

Marian Nemec*
ELORET Corp., Moffett Field, CA 94035

Michael J. Aftosmis†
NASA Ames Research Center, Moffett Field, CA 94035

Abstract submitted to the 17th AIAA Computational Fluid Dynamics Conference
June 6–9, 2005/Toronto, ON

I. Introduction

ADJOINT solutions of the governing flow equations are becoming increasingly important for the development of efficient analysis and optimization algorithms. A well-known use of the adjoint method is gradient-based shape optimization.[1,2,3,4,5,6] Given an objective function that defines some measure of performance, such as the lift and drag functionals, its gradient is computed at a cost that is essentially independent of the number of design variables (geometric parameters that control the shape). More recently, emerging adjoint applications focus on the analysis problem, where the adjoint solution is used to drive mesh adaptation,[7,8,9,10] as well as to provide estimates of functional error bounds and corrections.[11] The attractive feature of this approach is that the mesh-adaptation procedure targets a specific functional, thereby localizing the mesh refinement and reducing computational cost.

Our focus is on the development of adjoint-based optimization techniques for a Cartesian method with embedded boundaries.[12] In contrast to implementations on structured and unstructured grids, Cartesian methods decouple the surface discretization from the volume mesh.[13] This feature makes Cartesian methods well suited for the automated analysis of complex geometry problems, and consequently a promising approach to aerodynamic optimization. Melvin *et al.*[14] developed an adjoint formulation for the TRANAIR code,[15] which is based on the full-potential equation with viscous corrections. More recently, Dadone and Grossman[16] presented an adjoint formulation for the Euler equations. In both approaches, a boundary condition is introduced to approximate the effects of the evolving surface shape that results in accurate gradient computation.

Central to automated shape optimization algorithms is the issue of geometry modeling and control. The need to optimize complex, "real-life" geometry provides a strong incentive for the use of parametric-CAD systems within the optimization procedure. In previous work,[17] we presented an effective optimization framework that incorporates a direct-CAD interface.[18] In this work, we enhance the capabilities of this framework with efficient gradient computations using the discrete adjoint method. We present details of the adjoint numerical implementation, which reuses the domain decomposition, multigrid, and time-marching schemes of the flow solver. Furthermore, we explain and demonstrate the use of CAD in conjunction with the Cartesian adjoint approach. *The final paper will contain a number of complex geometry, industrially relevant examples with many design variables to demonstrate the effectiveness of the adjoint method on Cartesian meshes.*

*Research Scientist, Applications Branch, Advanced Supercomputing Division, MS T27B; nemec@nas.nasa.gov. Member AIAA.

†Research Scientist, Applications Branch, Advanced Supercomputing Division, MS T27B; maftosmis@mail.arc.nasa.gov. Senior Member AIAA.

II. Problem Formulation

The aerodynamic shape optimization problem consists of determining values of design variables X, such that the objective function $\mathcal{J}$ is minimized

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

subject to constraint equations C_j

$$C_j(X, Q) \leq 0 \qquad j = 1, \ldots, N_c \tag{2}$$

where the vector Q denotes the continuous, conservative flow variables and N_c denotes the number of constraint equations. The flow variables are forced to satisfy the governing flow equations within a feasible region of the design space Ω

$$\mathcal{F}(X, Q) = 0 \qquad \forall\ X \in \Omega \tag{3}$$

which implicitly defines $Q = f(X)$.

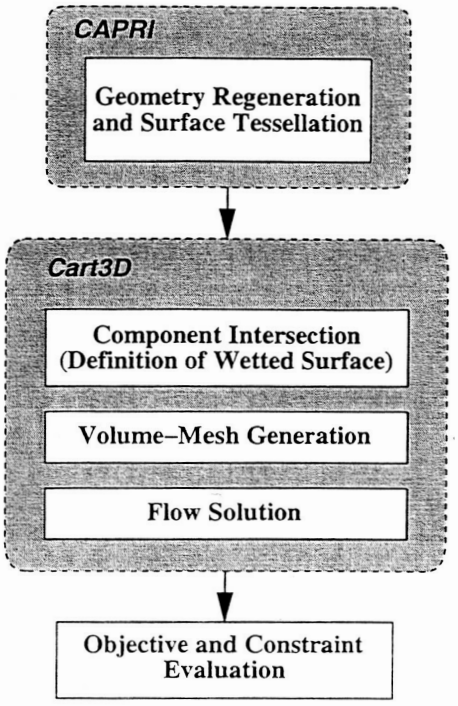


Figure 1. Components of the analysis module

The objective function defines the goals of the optimization problem, while the constraint equations limit the feasible region of the design space. The constraints may involve performance functionals, such as lift, geometric quantities, such as volumes and thicknesses, and also simple bound constraints for design variables. Performance objectives can be specified for the entire configuration or for a specific subset of components, for example moments on control surfaces. This is accomplished by using the Geometry Manipulation Protocol,[19] where we specify component hierarchies for parametric-CAD assemblies and triangulations to intuitively reflect aerodynamic design goals.

A modular framework[17] is used to solve the optimization problem defined by Eqs. 1–3. We cast the optimization problem as an unconstrained problem by lifting the side constraints, Eq. 2, into the objective function using a penalty method. The constraint imposed by the flowfield equations, Eq. 3, is satisfied at every point within the feasible design space, and consequently these equations do not explicitly appear in the formulation of the optimization problem. An unconstrained BFGS quasi-Newton algorithm coupled with a backtracking line search[20,21] is used to find the optimal solution. At the core of the optimization framework is the analysis module, which consists of a CAD system interface provided by CAPRI,[22,18] a Cartesian grid generator for component-based geometry,[23] and a flow solver, as outlined in Fig. 1. Below, we provide a brief description of the flow solver in order to clarify the subsequent discussion. Thereafter, we focus on the development of the gradient computation algorithm.

III. Governing Flow Equations and Numerical Method

The governing flow equations are the three-dimensional Euler equations of a perfect gas. For a finite region of space with volume V and surface area S, the integral form of the Euler equations is given by

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_V Q \, \mathrm{d}V + \oint_S \mathbf{F} \cdot n \, \mathrm{d}S = 0 \tag{4}$$

where $Q = [\rho, \rho u, \rho v, \rho w, \rho E]^{\mathrm{T}}$, $\mathbf{F}$ is the inviscid flux tensor

$$\mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(\rho E + p) \end{bmatrix} \hat{i} + \begin{bmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vw \\ v(\rho E + p) \end{bmatrix} \hat{j} + \begin{bmatrix} \rho w \\ \rho wu \\ \rho wv \\ \rho w^2 + p \\ w(\rho E + p) \end{bmatrix} \hat{k} \tag{5}$$

and n is the outward facing unit normal vector.

The Euler equations are solved with a finite-volume method on a regular Cartesian mesh with embedded boundaries. The mesh consists of hexahedral cells, except for a layer of body-intersecting cells, or cut-cells, that are arbitrary polyhedra adjacent to the boundaries.[23] Spatial discretization is based on a cell-centered approach, where the control volumes V correspond to the mesh cells and the cell-averaged value of Q, denoted by $\bar{Q}$, is located at the centroid of each cell. The control volumes are fixed in time, resulting in the following semi-discrete form of Eq. 4:

$$D \frac{\mathrm{d}\vec{Q}}{\mathrm{d}t} + \vec{R}\left(\vec{Q}\right) = 0 \tag{6}$$

where $\vec{Q} = [\bar{Q}_1, \bar{Q}_2, \ldots, \bar{Q}_N]^{\mathrm{T}}$ is the discrete solution vector for all N cells, D is a diagonal matrix containing the corresponding cell volumes, and $\vec{R}$ is the residual vector. The residual in each cell i is expressed as

$$R_i = \sum_{\mathrm{j} \in \mathrm{V_i}} \mathbf{F}_j \cdot n_j S_j \tag{7}$$

where j denotes the jth face of volume V_i with area S. The flux is evaluated at the face centroids based on a midpoint quadrature rule.

The residual evaluation for a second-order accurate discretization proceeds by first reconstructing the solution to the cell face, given by

$$Q_{\mathrm{L}} = \bar{Q}_i + d_{\mathrm{L}} \phi_i \nabla Q_i \tag{8}$$

where Q_{L} denotes the left state at a given face reconstructed from cell i, d_{L} is the distance from the cell centroid to the face centroid, ϕ is the limiter (in slope form) used to enforce monotonic solutions, and ∇Q is the solution gradient determined via a linear least-squares procedure. A similar formula is used to reconstruct the right state, denoted by Q_{R}. Next, the flux-vector splitting approach of van Leer is used to resolve the flux discontinuity at the face, represented by

$$\mathbf{F}(Q_{\mathrm{L}}, Q_{\mathrm{R}}) \cdot n = \mathrm{f}^{+}(Q_{\mathrm{L}}, n) + \mathrm{f}^{-}(Q_{\mathrm{R}}, n) \tag{9}$$

The assembly of the residual vector is accomplished by a loop over the faces of the mesh, where the flux contributions are scattered from the faces and accumulated in the cells. Steady-state solutions are obtained using a five-stage Runge–Kutta scheme in conjunction with multigrid and domain decomposition for parallel computing. For further details on the flow solution algorithm, see Aftosmis *et al.*[12,24] and Berger *et al.*[25]

IV. Adjoints and Sensitivities

A. Formulation

The gradient, G, of the discrete objective function $\mathcal{J}\left[X, \vec{Q}(X)\right]$ is given by

$$G = \frac{\mathrm{d}\mathcal{J}}{\mathrm{d}X} = \frac{\partial \mathcal{J}}{\partial X} + \frac{\partial \mathcal{J}}{\partial \vec{Q}} \frac{\mathrm{d}\vec{Q}}{\mathrm{d}X} \tag{10}$$

where $\vec{Q}$ is the steady-state solution of Eq. 4 for a given set of design variables

$$\vec{R}(X, \vec{Q}) = 0 \tag{11}$$

Note that this is the discrete equivalent of Eq. 3. We reduce the vector of design variables, X, to a scalar in order to clearly distinguish between partial and total derivatives. For problems with multiple design variables, it may be helpful to note that G and $\partial \mathcal{J}/\partial X$ are $[1 \times N_{\mathrm{D}}]$ row vectors, $\partial \mathcal{J}/\partial \vec{Q}$ is a $[1 \times N_{\mathrm{F}}]$ row vector, and $\mathrm{d}\vec{Q}/\mathrm{d}X$ is a $[N_{\mathrm{F}} \times N_{\mathrm{D}}]$ matrix, where N_{D} and N_{F} represent the number of design and flow variables, respectively.

In Eq. 10, the evaluation of the term $\mathrm{d}\vec{Q}/\mathrm{d}X$, referred to as the flow sensitivities, is obtained by differentiating Eq. 11 with respect to the design variables

$$\frac{\mathrm{d}\vec{R}}{\mathrm{d}X} = \frac{\partial \vec{R}}{\partial X} + \frac{\partial \vec{R}}{\partial \vec{Q}} \frac{\mathrm{d}\vec{Q}}{\mathrm{d}X} \tag{12}$$

where we assume that the implicit function $\vec{Q}(X)$ is sufficiently smooth.[26,5] Realizing that $\frac{\mathrm{d}\vec{R}}{\mathrm{d}X} = 0$, since for any design variable Eq. 11 is always satisfied, Eq. 12 simplifies to the following large sparse system of linear equations

$$\frac{\partial \vec{R}}{\partial \vec{Q}} \frac{\mathrm{d}\vec{Q}}{\mathrm{d}X} = -\frac{\partial \vec{R}}{\partial X} \tag{13}$$

The direct, or flow-sensitivity, method results from solving Eq. 13 for the flow sensitivities $\mathrm{d}\vec{Q}/\mathrm{d}X$ and using these values in Eq. 10 to obtain the gradient.

In order to formulate the discrete-adjoint method, substitute Eq. 13 into Eq. 10 to obtain

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}X} = \frac{\partial \mathcal{J}}{\partial X} - \frac{\partial \mathcal{J}}{\partial \vec{Q}} \left(\frac{\partial \vec{R}}{\partial \vec{Q}} \right)^{-1} \frac{\partial \vec{R}}{\partial X} \tag{14}$$

From the triple-product term in Eq. 14, define the following intermediate problem

$$\vec{\psi}^{\mathrm{T}} = \frac{\partial \mathcal{J}}{\partial \vec{Q}} \left(\frac{\partial \vec{R}}{\partial \vec{Q}} \right)^{-1} \tag{15}$$

where $\vec{\psi}$ is a $[N_{\mathrm{F}} \times 1]$ column vector. Post-multiplication of both sides by $\partial \vec{R}/\partial \vec{Q}$ and applying the transpose operator results in the following linear system of equations

$$\frac{\partial \vec{R}}{\partial \vec{Q}}^{\mathrm{T}} \vec{\psi} = \frac{\partial \mathcal{J}}{\partial \vec{Q}}^{\mathrm{T}} \tag{16}$$

This is known as the adjoint equation, and the vector $\vec{\psi}$ represents the adjoint variables. Substituting $\vec{\psi}$ into Eq. 14, the expression for the gradient becomes

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}X} = \frac{\partial \mathcal{J}}{\partial X} - \vec{\psi}^{\mathrm{T}} \frac{\partial \vec{R}}{\partial X} \tag{17}$$

Note that Eq. 13 is a linear system with multiple right-hand sides dependent on the number of design variables. In contrast, Eq. 16 is independent of the design variables, which is the reason for the efficiency of the adjoint method.

B. Numerical Implementation

Our goal is to reuse the solution method of the flow solver, in particular the Runge–Kutta time marching, local time stepping, multigrid, and domain decomposition schemes, to solve the adjoint and flow-sensitivity equations. This minimizes the code development time and simplifies code maintenance issues. A promising approach for generating flow-sensitivity (linear) and adjoint codes is automatic differentiation.[27,28,29,30] Although we do not use automatic differentiation in this work, we found that understanding the underlying principles and techniques is helpful for producing accurate and efficient sensitivity codes. Overall, the resulting approach is similar to the work of Giles *et al.*[31]

The solution of the adjoint and flow-sensitivity equations proceeds by introducing unsteady terms in Eqs. 13 and 16 to obtain

$$D\frac{\mathrm{d}\vec{Q}'}{\mathrm{d}t} + \left(\frac{\partial \vec{R}}{\partial \vec{Q}}\vec{Q}' + \frac{\partial \vec{R}}{\partial X}\right) = 0 \tag{18}$$

$$D\frac{\mathrm{d}\vec{\psi}}{\mathrm{d}t} + \left(\frac{\partial \vec{R}}{\partial \vec{Q}}^{\mathrm{T}}\vec{\psi} - \frac{\partial \mathcal{J}}{\partial \vec{Q}}^{\mathrm{T}}\right) = 0 \tag{19}$$

where $\vec{Q}' = \mathrm{d}\vec{Q}/\mathrm{d}X$. A key step in the solution procedure is the evaluation of the matrix-vector product terms

$$\frac{\partial \vec{R}}{\partial \vec{Q}}\vec{Q}' \quad \text{and} \quad \frac{\partial \vec{R}}{\partial \vec{Q}}^{\mathrm{T}}\vec{\psi} \tag{20}$$

We consider the flow-sensitivity equation first, since the explicit linearization of the residual equations can be avoided. The matrix-vector product is approximated as follows

$$\frac{\partial \vec{R}}{\partial \vec{Q}}\vec{Q}' = \frac{\vec{R}\left(\vec{Q} + \epsilon\vec{Q}'\right) - \vec{R}(\vec{Q})}{\epsilon} \tag{21}$$

with $\epsilon = \sqrt{(\epsilon_{\mathrm{m}})}/||v||$.[32,6] The final paper will include a detailed discussion of the flow-sensitivity and adjoint solvers.

The partial derivative term $\partial\mathcal{J}/\partial\vec{Q}$ in Eqs. 10 and 16 is evaluated using finite differences. The remaining partial derivative terms in Eqs. 10 and 17, namely the objective function sensitivity $\partial\mathcal{J}/\partial X$ and the residual sensitivity $\partial\vec{R}/\partial X$, are usually approximated with finite differences as well. Although in some instances the partial derivatives can be obtained analytically, finite differences are necessary when a CAD system is used to parameterize the geometry. For example, the centered-difference formula for the residual sensitivities is given by

$$\frac{\partial \vec{R}}{\partial X}_n = \frac{\vec{R}(X + he_n, \vec{Q}) - \vec{R}\left(X - he_n, \vec{Q}\right)}{2h} \tag{22}$$

where e_n denotes the nth unit vector,

$$h = \max\left(\epsilon_0 \cdot |X_n|, \sqrt{\epsilon_{\mathrm{m}}}\right) \tag{23}$$

and $n = 1, \ldots, N_{\mathrm{D}}$. Typical value of ϵ_0 is 1×10^{-3}. It is important to realize that Eq. 22 involves two evaluations of only the residual vector per design variable and *not* two flow solutions, i.e. $\vec{Q}$ is constant.

Before presenting our numerical implementation for the computation of $\partial\mathcal{J}/\partial X$ and $\partial\vec{R}/\partial X$ on Cartesian grids, it is insightful to briefly review the implementations on structured and unstructured grids. The standard finite-difference approach relies on a grid-perturbation strategy to smoothly "convect" the nodes of the volume grid with the perturbed surface geometry, as sketched in Fig. 2. This results in an accurate and efficient evaluation of the partial derivative terms. Alonso *et al.*[33] have successfully extended this approach to a CAD-based design environment by introducing geometry patches within the parametric-CAD model.

An important observation is that the grid-perturbation strategy explicitly couples shape sensitivities to volume grid sensitivities. However, a grid-perturbation without a corresponding surface perturbation should not, in principle, influence the objective function gradient. On the basis of this argument, Jameson and Kim[34] introduced a reduced adjoint gradient formulation where only surface perturbations are considered. Their results indicate that for shape optimization problems the reduced approach works well. Similar results are also presented by Soto and Löhner.[35] In contrast, work by Anderson and Venkatakrishnan[3] and later by Nielsen and Anderson[36] shows that for cases that involve rigid body motion, for example flap position optimization, the use of a grid-perturbation strategy is important. This appears to be related to the treatment of the trailing-edge singularity.

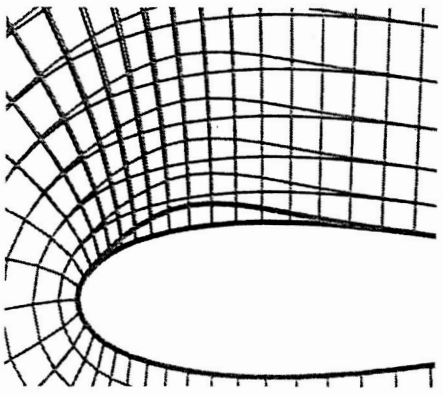


Figure 2. Grid-perturbation approach (baseline airfoil and grid are red, perturbation is green)

Turning our attention back to embedded-boundary Cartesian grids, we note that a perturbation of the surface shape affects only a few near-by cells of the volume grid, see Fig. 3. Therefore, the extent of grid sensitivities is naturally limited to roughly $\mathcal{O}(N^2)$ cells for a volume grid with $\mathcal{O}(N^3)$ cells. However, a straightforward implementation of finite differences is complicated by the emergence and disappearance of cut-cells at the wall boundary, as well as the relative motion of the discretized surface with respect to the volume grid.

To circumvent these difficulties, we use the following approach. For $\partial \mathcal{J}/\partial X$, the converged flow solution is reconstructed to the vertices of the underlying surface triangulation. Given a small perturbation, a new surface triangulation is generated using the CAPRI interface. We force the new triangulation to have a one-to-one triangle mapping with respect to the baseline triangulation. This is accomplished by using the parametric values of the baseline triangulation when tessellating the perturbed surface. The objective function values required for the finite-difference approximation are now evaluated using the surface triangulations instead of the volume grid. Hence, the potentially non-smooth changes in the cut-cells are avoided, and the enforcement of constant $\vec{Q}$ in the evaluation of this partial derivative term becomes a trivial task. This approach works well for both shape and rigid-body motion design variables.

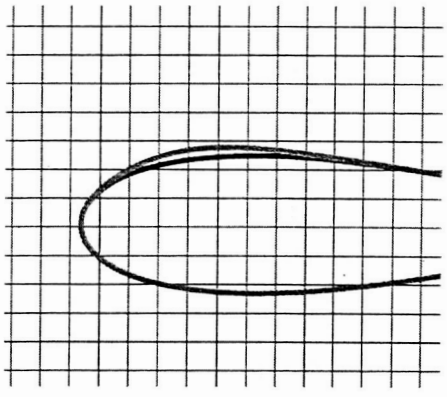


Figure 3. Cartesian grid with a baseline (red) and perturbed (green) airfoils

For the remaining term $\partial \vec{R}/\partial X$, we have implemented and studied two approaches, both based on finite-differences. In the first approach, we assume that the only parameter dependent on the design variables in Eq. 11 is the surface normal. This approach is similar to the "transpiration" boundary condition used in TRANAIR.[37,14] It is relatively easy to implement, and the evaluation of $\partial \vec{R}/\partial X$ is fast since the volume grid remains unchanged. The accuracy of this approximation can be sufficient when the design variables involve shape changes normal to the surface including wing-twist distributions. However, a possible limitation of this approach may be rigid-body motion.

In the second approach, we construct a new volume grid for the perturbed surface triangulation. The baseline solution is interpolated to the new grid via a fast interpolation algorithm presented in Ref. 24. We evaluate the residual on the new grid, which is then passed back to the baseline grid for the finite-difference computation. Cut-cells that appear or disappear during this procedure are tagged, and we either zero their contribution to the residual sensitivity computation, or we interpolate a value from their face neighbors. We evaluate both approaches in the following section.

V. Results and Discussion

In this abstract we study a simple, two-dimensional design example to validate our approach. The baseline geometry is the NACA 0012 airfoil parameterized with a cubic B-spline curve using 17 control points. We choose the vertical motion of a control point near the leading edge of the airfoil as the design variable, shown in Fig. 4. The freestream Mach number is 0.5 and the angle of incidence is fixed at 2 deg. The goal of the optimization is lift enhancement. The baseline C_L is 0.245 and we specify a target C_L of 0.5. The "volume" grid contains roughly 9,000 cells, but since this is a two-dimensional problem, the depth in the span-wise direction contains only two cells. To help validate the results, we solve a similar problem using *Optima2D*,[6] which is an aerodynamic optimization tool based on a structured-grid approach.

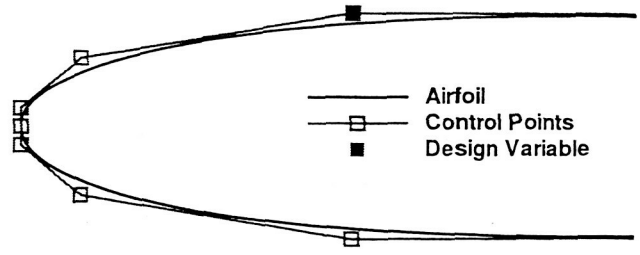


Figure 4. Design variable and B-spline parameterization of the NACA 0012 airfoil

We begin by examining the computation of the $\partial \mathcal{J}/\partial X$ term in Eq. 10. We evaluate the partial derivative of the lift functional (objective function) for the above problem with the angle of incidence set to 0 and 2 deg. Note that at 0 deg., this partial derivative term should vanish, providing a good validation exercise. The result at 0 deg. is indeed zero, and at 2 deg. we obtain -2.81×10^{-3}. *Optima2D* predicts -1.4×10^{-2}, which is a reasonable agreement considering the differences of the two approaches.

Prior to solving the flow-sensitivity equation, we evaluate the residual sensitivity $\partial \vec{R}/\partial X$. Figure 5(a) shows $\partial \vec{R}/\partial X$ values for the y-momentum equation. These are obtained by constructing (or re-cutting) a new volume grid for the perturbed shape and interpolating the steady-state solution. Figure 5(b) shows the results when only the surface normals are changed. The agreement in Fig.5 is excellent; however, we note that for the remaining field variables significant errors may arise due to this approximation. We plan to investigate this further in the final paper.

The solution of the flow-sensitivity equation is shown in Fig. 6. Note that the zone of influence for the shape design variable extends from the leading edge to roughly 60% chord on the upper surface. The convergence of the flow and sensitivity equations is shown in Fig. 7. The flow equations converge in 500 multigrid cycles (only a 2-level multigrid is used), and the flow sensitivities require additional 400 cycles. The final gradient value is -1.35. *Optima2D* computes a value of -1.67, while a finite-difference estimation of the gradient by recomputing the flow gives -1.55. *A grid-refinement study will be included in the final paper to validate gradient accuracy. Furthermore, the final paper will contain additional design examples to demonstrate the effectiveness of the proposed approach.*

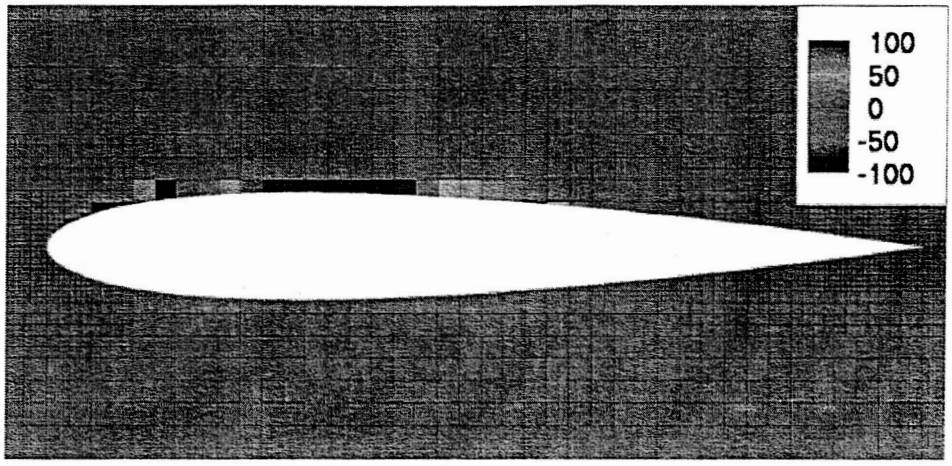
VI. Acknowledgments

The funding of the first author by the National Research Council Research Associateship Award is gratefully acknowledged.

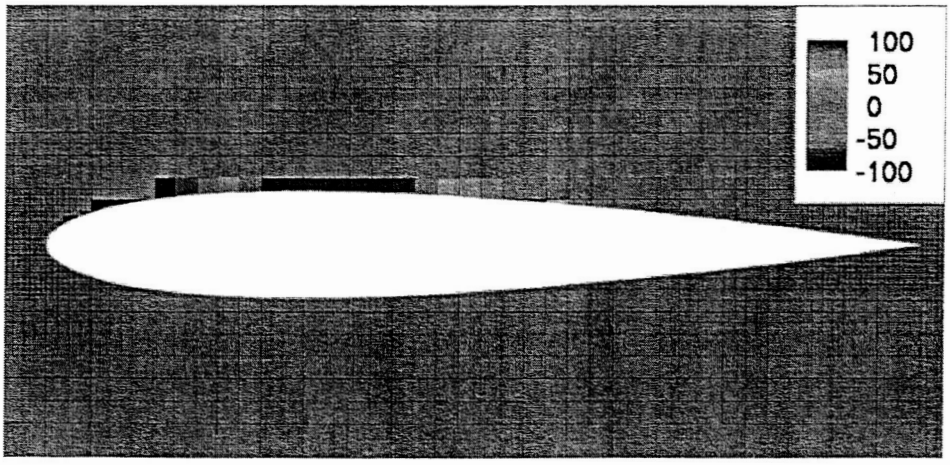
References

[1] Jameson, A., "Aerodynamic Design via Control Theory," *Journal of Scientific Computing*, Vol. 3, 1988, pp. 233–260, Also ICASE report 88–64.

[2] Reuther, J. J., *Aerodynamic Shape Optimization Using Control Theory*, Ph.D. thesis, University of California Davis, 1996.

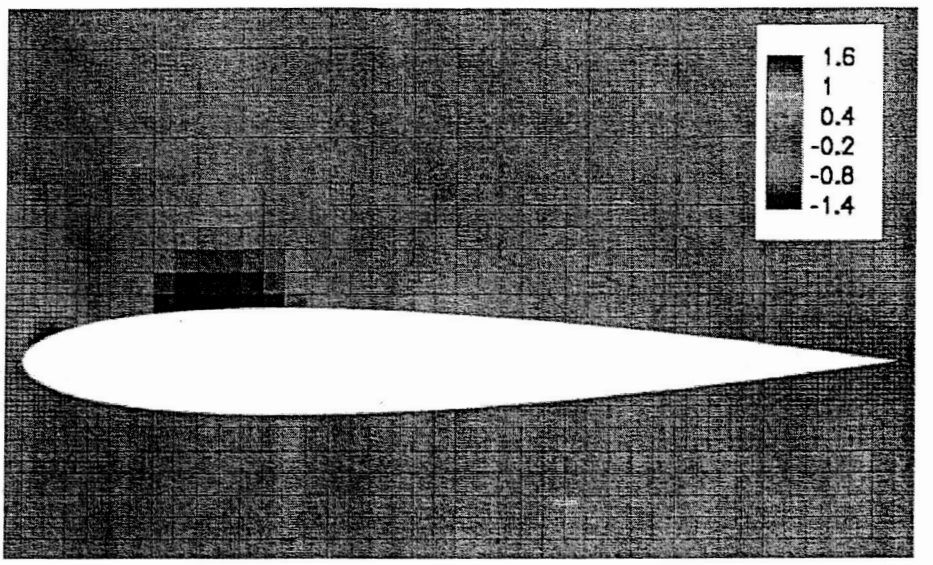


(a) Sensitivity of Eq. 11 via the grid re-cutting and interpolation approach

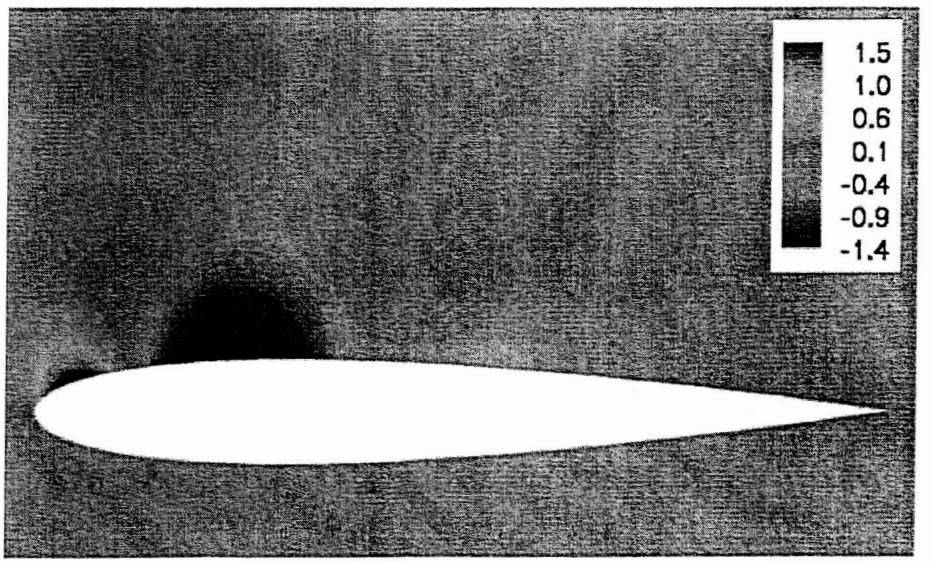


(b) Sensitivity of Eq. 11 using variations of only surface normals

Figure 5. Contours of $\partial\vec{R}/\partial X$ for the y-momentum equation



(a) Cartesian-grid formulation



(b) Structured-grid approach using *Optima2D*[6]

Figure 6. Contour plots of flow sensitivity: changes in density with respect to a shape design variable

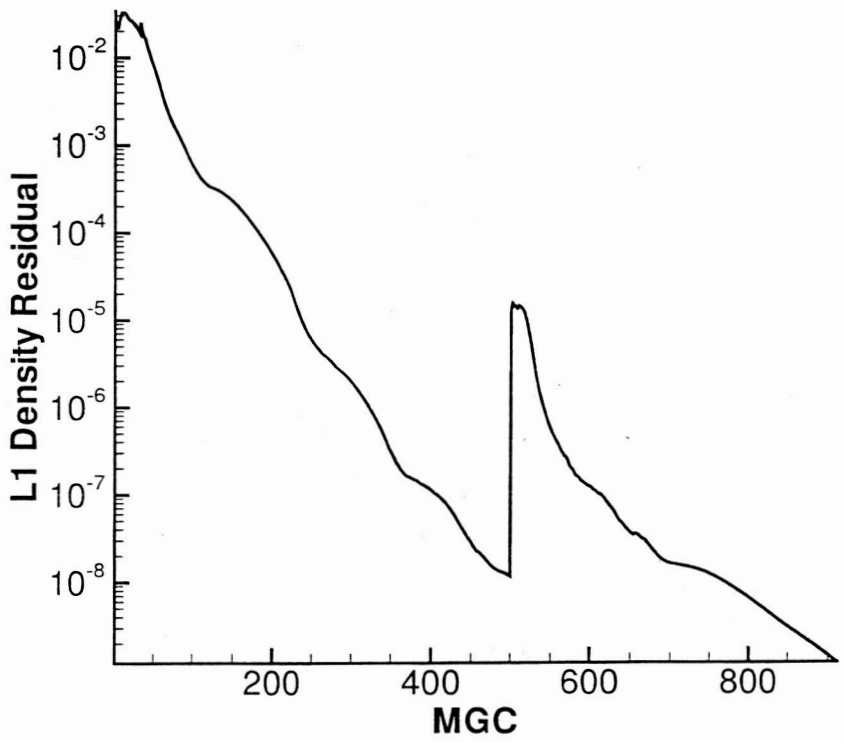


Figure 7. Convergence to steady-state for the mean flow and the flow-sensitivity equation

[3]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.

[4]Elliott, J. and Peraire, J., "Constrained, Multipoint Shape Optimisation for Complex 3D Configurations," *Aeronautical Journal*, Vol. 102, No. 1017, 1998, pp. 365–376.

[5]Giles, M. B. and Pierce, N. A., "An Introduction to the Adjoint Approach to Design," *Flow, Turbulence and Combustion*, Vol. 65, No. 3/4, 2000, pp. 393–415.

[6]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.

[7]Pierce, N. A. and Giles, M. B., "Adjoint Recovery of Superconvergent Functionals from PDE Approximations," *SIAM Review*, Vol. 42, No. 2, 2000, pp. 247–264.

[8]Venditti, D. A. and Darmofal, D. L., "Anisotropic Grid Adaptation for Functional Outputs: Application to Two-Dimensional Viscous Flow," *Journal of Computational Physics*, Vol. 187, 2003, pp. 22–46.

[9]Barth, T. J., "Numerical Methods and Error Estimation for Conservation Laws on Structured and Unstructured Meshes," Lecture notes, von Karman Institute for Fluid Dynamics, Series: 2003-04, Brussels, Belgium, March 2003.

[10]Park, M. A., "Adjoint-Based, Three Dimensional Error Prediction and Grid Adaptation," *AIAA Journal*, Vol. 42, No. 9, 2004, pp. 1854–1862.

[11]Pierce, N. A. and Giles, M. B., "Adjoint and defect error bounding and correction for functional estimates," *Journal of Computational Physics*, Vol. 200, 2004, pp. 769–794.

[12]Aftosmis, M. J., Berger, M. J., and Adomavicius, G., "A Parallel Multilevel Method for Adaptively Refined Cartesian Grids with Embedded Boundaries," AIAA Paper 2000–0808, Reno, NV, Jan. 2000.

[13]Aftosmis, M. J., "Solution Adaptive Cartesian Grid Methods for Aerodynamic Flows with Complex Geometries," Lecture notes, von Karman Institute for Fluid Dynamics, Series: 1997-02, Brussels, Belgium, March 1997.

[14]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," *International Journal for Numerical Methods in Fluids*, Vol. 30, 1999, pp. 205–216.

[15]Young, D. P., Melvin, R. G., Bieterman, M. B., Johnson, F. T., and Samant, S. S., "A Locally Refined Rectangular Grid Finite Element Method: Application to Computational Fluid Dynamics and Computational Physics," *Journal of Computational Physics*, Vol. 92, No. 1, 1991, pp. 1–66.

[16]Dadone, A. and Grossman, B., "Efficient Fluid Dynamic Design Optimization Using Cartesian Grids," AIAA Paper 2003–3959, Orlando, FL, June 2003.

[17]Nemec, M., Aftosmis, M. J., and Pulliam, T. H., "CAD-Based Aerodynamic Design of Complex Configurations Using a Cartesian Method," AIAA Paper 2004–0113, Reno, NV, Jan. 2004.

[18]Haimes, R. and Crawford, C., "Unified Geometry Access for Analysis and Design," Tech. rep., 12th International Meshing Roundtable, Santa Fe, NM, Sept. 2003.

[19]Murman, S. M., Chan, W. M., Aftosmis, M. J., and Meakin, R. L., "An Interface for Specifying Rigid-Body Motions for CFD Applications," AIAA Paper 2003–1237, Reno, NV, Jan. 2003.

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

[21]Moré, J. J. and Thuente, D. J., "Line Search Algorithms with Guaranteed Sufficient Decrease," *ACM Transactions on Mathematical Software*, Vol. 20, No. 3, 1994, pp. 286–307.

[22]Haimes, R. and Aftosmis, M. J., "On Generating High Quality "Water-tight" Triangulations Directly from CAD," Tech. rep., Meeting of the International Society for Grid Generation, (ISGG), Honolulu, HI, June 2002.

[23]Aftosmis, M. J., Berger, M. J., and Melton, J. E., "Robust and Efficient Cartesian Mesh Generation for Componenet-Based Geometry," *AIAA Journal*, Vol. 36, No. 6, 1998, pp. 952–960.

[24]Aftosmis, M. J., Berger, M. J., and Murman, S. M., "Applications of Space-Filling-Curves to Cartesian Methods for CFD," AIAA Paper 2004–1232, Reno, NV, Jan. 2004.

[25]Berger, M. J., Aftosmis, M. J., and Murman, S. M., "Linearity-Preserving Limiters on Irregular Grids," Preprint.

[26]Giles, M. B. and Pierce, N. A., "Analytic Adjoint Solutions for the Quasi-One-Dimensional Euler Equations," *Journal of Fluid Mechanics*, Vol. 426, 2001, pp. 327–345.

[27]Giering, R. and Kaminski, T., "Recipes for Adjoint Code Construction," *ACM Transactions on Mathematical Software*, Vol. 24, No. 4, 1998, pp. 437–474.

[28]Mohammadi, B. and Pironneau, O., *Applied Shape Optimization for Fluids*, Oxford University Press, New York, USA, 2001.

[29]Cusdin, P. and Müller, J.-D., "Deriving Linear and Adjoint Codes for CFD using Automatic Differentiation," Preprint submitted to AIAA J., www.ea.qub.ac.uk/jmueller/.

[30]Cusdin, P. and Müller, J.-D., "Generating Efficient Code with Automatic Differentiation," *4th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS)*, edited by P. Neittaanmäki, T. Rossi, K. Majava, and O. Pironneau, July 2004.

[31]Giles, M. B., Duta, M. C., Müller, J.-D., and Pierce, N. A., "Algorithm Developments for Discrete Adjoint Methods," *AIAA Journal*, Vol. 41, No. 2, 2003, pp. 198–204.

[32]Barth, T. J., "Parallel CFD Algorithms on Unstructured Meshes," *Special Course on Parallel Computing in CFD*, AGARD-R-807, Oct. 1995, pp. 7–1–7–41.

[33]Alonso, J. J., Martins, J. R. R. A., Reuther, J. J., Haimes, R., and Crawford, C., "High-Fidelity Aero-Structural Design Using a Parametric CAD-Based Model," AIAA Paper 2003–3429, Orlando, FL, June 2003.

[34]Jameson, A. and Kim, S., "Reduction of the Adjoint Gradient Formula in the Continuous Limit," AIAA Paper 2003–0040, Jan. 2003.

[35]Soto, O. and Löhner, R., "On the Computation of Flow Sensitivities from Boundary Integrals," AIAA Paper 2004–0112, Jan. 2004.

[36]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.

[37]Huffman, W. P., Melvin, R. G., Young, D. P., Johnson, F. T., Bussoletti, J. E., Bieterman, M. B., and Hilmes, C. L., "Practical Design and Optimization in Computational Fluid Dynamics," AIAA Paper 93–3111, 1993.