Some Advanced Concepts in Discrete Aerodynamic Sensitivity Analysis

Arthur C. Taylor, Lawrence L. Green, Perry A. Newman, Michele M. Putko

Institutions: Old Dominion University, Langley Research Center

Published on: 01 Jan 2001 - AIAA Journal (American Institute of Aeronautics and Astronautics (AIAA))

Topics: Airfoil, Inviscid flow, Aerodynamics, Lift (force) and Aerodynamic drag

AIAA JOURNAL
Vol. 41, No. 7, July 2003

Some Advanced Concepts in Discrete Aerodynamic Sensitivity Analysis

Arthur C. Taylor III*
Old Dominion University, Norfolk, Virginia 23529
Lawrence L. Green† and Perry A. Newman‡
NASA Langley Research Center, Hampton, Virginia 23681
and
Michele M. Putko§
Old Dominion University, Norfolk, Virginia 23529

An efficient incremental-iterative approach for differentiating advanced flow codes is successfully demonstrated on a two-dimensional inviscid model problem. The method employs the reverse-mode capability of the automatic-differentiation software tool ADIFOR 3.0 and is proven to yield accurate first-order aerodynamic sensitivity derivatives. A substantial reduction in CPU time and computer memory is demonstrated in comparison with results from a straightforward, black-box reverse-mode application of ADIFOR 3.0 to the same flow code. An ADIFOR-assisted procedure for accurate second-order aerodynamic sensitivity derivatives is successfully verified on an inviscid transonic lifting airfoil example problem. The method requires that first-order derivatives are calculated first using both the forward (direct) and reverse (adjoint) procedures; then, a very efficient noniterative calculation of all second-order derivatives can be accomplished. Accurate second derivatives (i.e., the complete Hessian matrices) of lift, wave drag, and pitching-moment coefficients are calculated with respect to geometric shape, angle of attack, and freestream Mach number.

I. Introduction

THIS paper revisits and focuses entirely on the computational challenges that are associated with the efficient calculation of aerodynamic sensitivity derivatives (SDs) from advanced computational fluid dynamics (CFD) codes. Of course, an accurate efficient methodology for obtaining these derivatives is a critical prerequisite concern that must be addressed first by the aerodynamic design engineer who chooses any gradient-based method(s) for design optimization and/or for estimating quantities related to aerodynamic uncertainty. Thus computing SDs from high-fidelity nonlinear CFD codes is an enabling technology for design of advanced concept vehicles.

In recent years significant progress has been achieved in the efficient calculation of accurate SDs from these CFD codes.[1] The automatic differentiation (AD) software tool ADIFOR (Automatic Differentiation of FORTRAN) has been proven an effective tool for extracting aerodynamic SDs from these modern CFD codes.[2–6] The foundation of the present work is found in Refs. 3 and 6; the present study builds on these earlier studies in an effort to exploit the full potential of the latest version of ADIFOR 3.0 (Ref. 7) for obtaining SDs from CFD codes.

In Ref. 2 a strategy known as the ADII method was first proposed and later successfully demonstrated in Ref. 3, whereby AD was applied to a CFD code in incremental-iterative (I-I) form. The ADII method is a hybrid (compromise) scheme, designed to maintain as much as possible the computational efficiency of a hand-differentiated (HD) approach and the ease of implementation of a straightforward black-box (BB) application of AD; at the same time the accuracy of the SDs is not compromised. A comprehensive overview of the development of the ADII scheme is given in Ref. 3. Also included in Ref. 3 is a comparison of the ADII scheme with the HD and BB approaches; computational issues associated with CPU time, computer memory, and SD accuracy are discussed. The two-dimensional effort of Ref. 3 was later extended to the three-dimensional code CFL3D, including "in-parallel" computation of the derivatives.[8,9] Appropriate references to the version of CFL3D used can be found in Ref. 6.

The success reported in these previous works[3,8,9] could be considered limited, however, because all ADIFOR implementations reported therein were forward-mode (direct) differentiations. It is very difficult to make any forward-mode implementation of derivative calculations computationally competitive with a reverse-mode (adjoint) implementation whenever the number of design variables (NDV) of interest is considerably larger than the number of output functions (NOF) of interest, and NDV much greater than NOF is more typical for aerodynamic design problems. In recent studies the new reverse-mode capability of ADIFOR 3.0 (not available for the earlier referenced studies) has been successfully verified in Ref. 6 by application to a parallel version of CFL3D and in Ref. 10 by application to a sequential linear aerodynamics code. These applications resulted in accurate design SDs as well as stability and control derivatives, respectively. The application reported in Ref. 6 involved BB AD of the entire CFD code, but iterative execution of the reverse mode was required only over the last iteration of the function evaluation.

In the present study it is proposed and demonstrated that the reverse-mode capability of ADIFOR 3.0 can also be applied to CFD codes in I-I form, resulting in a hybrid adjoint-variable (AV) scheme (known herein as the ADII-AV method) that is analogous to the forward-mode ADII scheme of Ref. 3 and elsewhere. The motivation of this new reverse-mode ADII-AV scheme is identical to that of the earlier forward-mode ADII method: greater computational efficiency is sought over a BB implementation of AD, without any loss of accuracy in the calculated SDs and without unmanageable complications upon implementation.

Received 5 September 2001; revision received 9 January 2003; accepted for publication 24 January 2003.

*Professor, Department of Mechanical Engineering. Member AIAA.

†Research Scientist, Multidisciplinary Optimization Branch. Senior Member AIAA.

‡Senior Research Scientist, Multidisciplinary Optimization Branch.

§LTC, U.S. Army, and Ph.D. Candidate, Department of Mechanical Engineering.

Following development of the proposed new ADII-AV scheme, the second focus of the present study is that of calculating second-order (SO) aerodynamic SDs from CFD codes. The motivation for calculating SO SDs is to advance the possibility of (or greater capability of) 1) second-order gradient-based aerodynamic design optimization, 2) analysis and design involving vehicle stability and control and 3) robust design [i.e., design under uncertainty, where a first-order, second-moment method requires SO SDs[11]]. This second part of the present study is another extension of Ref. 3, wherein the computational issues associated with calculating these higher-order derivatives were addressed, and sample calculations of SO derivatives using AD were reported from a two-dimensional CFD code.

In Ref. 3, four procedures for calculating SO CFD SDs were proposed, but only one of the less efficient methods was actually tested; ADIFOR 3.0 currently provides three forward-mode variations for the calculation of SO SDs by similarly inefficient methods. The most efficient (for large NDV) SO SD scheme was not tested in the earlier study[3] but has been successfully implemented in the present study. Reverse-mode (adjoint-based) differentiation is required within this efficient SO SD scheme, via either HD or AD. However, with the availability of ADIFOR 3.0 and the new ADII-AV scheme the door has been opened for AD implementation and testing of this SO SD scheme for CFD codes. The results of this effort to date are reported here. These efficiently computed SO SDs have been used to demonstrate an approach for CFD input uncertainty propagation and robust design optimization for a quasi-one-dimensional flow application in Ref. 11.

II. Basic Equations and Theoretical Development

The equations summarized subsequently are discussed in greater detail in the references, in particular, Ref. 3. These concepts are known in the mathematical optimization community,[12] but the details developed here do not appear to be generally known throughout the CFD community. The aerodynamic output functions of interest $\boldsymbol{F}$ and the discretized conservation laws of steady compressible fluid flow $\boldsymbol{R}$, including boundary conditions, can be represented symbolically as follows.

Aerodynamic output functions:

$$\boldsymbol{F} = \boldsymbol{F}[\boldsymbol{Q}(\boldsymbol{b}), \boldsymbol{X}(\boldsymbol{b}), \boldsymbol{b}] \tag{1}$$

Nonlinear state equations:

$$\boldsymbol{R} = \boldsymbol{R}[\boldsymbol{Q}(\boldsymbol{b}), \boldsymbol{X}(\boldsymbol{b}), \boldsymbol{b}] = \boldsymbol{0} \tag{2}$$

where $\boldsymbol{Q}$ is the vector of state (field) variables, $\boldsymbol{X}$ is the vector of computational grid coordinates, and $\boldsymbol{b}$ is the vector of input (design) variables.

A. First-Order Sensitivity Derivatives

Subject to the following definitions, index (summation) notation is now introduced:

$$F'_{ij} \equiv \frac{\mathrm{d}F_i}{\mathrm{d}b_j} = \left(\frac{\mathrm{d}\boldsymbol{F}}{\mathrm{d}\boldsymbol{b}}\right)_{ij}, \qquad R'_{lj} \equiv \frac{\mathrm{d}R_l}{\mathrm{d}b_j} = \left(\frac{\mathrm{d}\boldsymbol{R}}{\mathrm{d}\boldsymbol{b}}\right)_{lj}$$

$$Q'_{mj} \equiv \frac{\mathrm{d}Q_m}{\mathrm{d}b_j} = \left(\frac{\mathrm{d}\boldsymbol{Q}}{\mathrm{d}\boldsymbol{b}}\right)_{mj}, \qquad X'_{pj} \equiv \frac{\mathrm{d}X_p}{\mathrm{d}b_j} = \left(\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\boldsymbol{b}}\right)_{pj}$$

(This notation will be necessary to avoid subsequent ambiguity when the SO SD methods are presented.) The forward-mode (direct) approach for calculating first-order (FO) SDs is developed by differentiation of Eqs. (1) and (2) with respect to the design variables; the result is

$$F'_{ij} = \frac{\mathrm{d}F_i}{\mathrm{d}b_j} = \frac{\partial F_i}{\partial Q_m}Q'_{mj} + \frac{\partial F_i}{\partial X_p}X'_{pj} + \frac{\partial F_i}{\partial b_j} \tag{3}$$

$$R'_{lj} = \frac{\mathrm{d}R_l}{\mathrm{d}b_j} = \frac{\partial R_l}{\partial Q_m}Q'_{mj} + \frac{\partial R_l}{\partial X_p}X'_{pj} + \frac{\partial R_l}{\partial b_j} = 0_{lj} \tag{4}$$

In the preceding equations i, j, and l are "free" indices, and repeated indices m and p are (by convention) "summation" indices. The reverse-mode (adjoint) approach for the FO SDs is developed starting with an application of the chain rule:

$$\frac{\partial F_i}{\partial R_l}\frac{\partial R_l}{\partial Q_m} = \frac{\partial F_i}{\partial Q_m} \tag{5}$$

in order to show that the conventional adjoint variable is indeed a derivative. With Eq. (5) it then follows from Eq. (4) that Eq. (3) can be written as

$$F'_{ij} = \frac{\mathrm{d}F_i}{\mathrm{d}b_j} = -\frac{\partial F_i}{\partial R_l}\left(\frac{\partial R_l}{\partial X_p}X'_{pj} + \frac{\partial R_l}{\partial b_j}\right) + \frac{\partial F_i}{\partial X_p}X'_{pj} + \frac{\partial F_i}{\partial b_j} \tag{6}$$

A more conventional derivation of the adjoint-variable method (i.e., the Lagrange-multiplier method) gives

$$F'_{ij} = \frac{\mathrm{d}F_i}{\mathrm{d}b_j} = \lambda_{il}\left(\frac{\partial R_l}{\partial X_p}X'_{pj} + \frac{\partial R_l}{\partial b_j}\right) + \frac{\partial F_i}{\partial X_p}X'_{pj} + \frac{\partial F_i}{\partial b_j} \tag{7}$$

$$G_{im} \equiv \lambda_{il}\frac{\partial R_l}{\partial Q_m} + \frac{\partial F_i}{\partial Q_m} = 0_{im} \tag{8}$$

where λ_{il} is called the adjoint variable. Comparison of Eqs. (6) and (5) with Eqs. (7) and (8), respectively, reveals the identity

$$-\frac{\partial F_i}{\partial R_l} = \lambda_{il} \tag{9}$$

One objective of this particular development of the AV method for aerodynamic SDs is to ensure that the relationship given by Eq. (9) is clearly understood; that is, λ_{il} is the derivative of the output F_i with respect to the intermediate variable R_l and is accumulated in the reverse-mode AD.

The I-I strategies for solving the preceding equations for Q'_{mj} and/or λ_{il} required for F'_{ij} are reviewed here; additional detail is found in Refs. 3 and 13 and elsewhere. The I-I method for solving the nonlinear flow of Eq. (2) is

$$Q_m^{N+1} = Q_m^N - P_{ml}^N R_l^N \tag{10}$$

where the superscript N is the iteration (pseudo-time step) index and the operator

$$P_{ml}^N = \frac{\partial Q_m}{\partial \tilde{R}_l^N} = \left(\frac{\partial \tilde{R}_l^N}{\partial Q_m}\right)^{-1} \tag{11}$$

represents the solution algorithm of the particular CFD code of choice. The tilde in Eq. (11) serves to indicate that P_{ml}^N can be viewed as any computationally efficient approximation (often a very crude approximation) of the exact operator associated with true Newton–Raphson iteration. Thus the CFD solution algorithm is simply quasi-Newton iteration.

The I-I method for solving the forward mode FO SD equation (4) is

$$Q'^{M+1}_{mj} = Q'^{M}_{mj} - P_{ml}R'^{M}_{lj} \tag{12}$$

where superscript M is the FO SD iteration index and

$$R'^{M}_{lj} = \frac{\partial R_l}{\partial Q_m}Q'^{M}_{mj} + \frac{\partial R_l}{\partial X_p}X'_{pj} + \frac{\partial R_l}{\partial b_j} \tag{13}$$

With the I-I methodology the CFD flow solution operator P_{ml} is also used to solve the SD equations; this operator in Eq. (12) is evaluated and fixed using the steady-state solution for the nonlinear flow. The requisite terms of Eq. (13) are constructed either by hand differentiation (i.e., the HDII method, which is very tedious and time consuming to complete with accuracy for advanced CFD codes) or by AD, which is the forward-mode ADII method of previous studies.

In contrast with the ADII method, a straightforward BB application of AD to the CFD code, which is the ADBB method, is represented symbolically as

$$Q'^{N+1}_{mj} = Q'^{N}_{mj} - P^N_{ml} R'^N_{lj} - P'^N_{mlj} R^N_l \tag{14}$$

Clearly ADII [Eqs. (12) and (13)] and ADBB [Eq. (14)] yield the same result at steady-state convergence of each [recall Eq. (2)]; however, ADII is potentially more efficient than ADBB because of user intervention in the application of AD. With ADII the following is true:

1) The operator P^N_{ml} can be evaluated only once [hence denoted P_{ml} in Eq. (12)] using the steady-state field variables Q and then reused for all M iterations and for all $j =$ NDV design variables in obtaining the Q'_{mj}.

2) All derivatives except Q'_{mj} can be computed once outside the iteration loop and frozen for reuse inside the loop.

3) Evaluation of the terms P'^N_{mlj} in Eq. (14) can be avoided completely for all iterations and all design variables.

The I-I method for solving the reverse-mode AV, FO SD Eq. (9) is

$$\lambda^{M+1}_{il} = \lambda^M_{il} - P_{ml} G^M_{im} \tag{15}$$

where

$$G^M_{im} \equiv \lambda^M_{il} \frac{\partial R_l}{\partial Q_m} + \frac{\partial F_i}{\partial Q_m} \tag{16}$$

The requisite terms of Eq. (16) are constructed either by hand (i.e., the HDII-AV method, having the same drawbacks as the forward-mode HDII method) or by AD, which is the proposed new ADII-AV scheme. The BB AD in reverse mode (the ADBB-AV method) has been verified in Ref. 6. The objective of the proposed ADII-AV scheme is improved computational efficiency over the ADBB-AV approach without resulting loss of accuracy or significant loss in the ease of implementation. The mechanisms from which improved computational efficiency can be expected are analogous to those explained before when the forward-mode ADII and ADBB methods were contrasted. Furthermore, the ADII-AV scheme should lend itself to more permanent generalized coding implementations than the ADBB-AV approach. This is because with the ADII-AV method the manner in which AD is applied is independent of, yet valid for, all of the particular aerodynamic inputs and outputs of interest.

The forward-mode application of ADIFOR produces FORTRAN source code for very efficient calculation of the vector (or matrix) product that results from the postmultiplication of a large Jacobian matrix by a known input vector (or matrix). This attribute of forward-mode AD is exactly what was required to construct the ADII method; specifically, the terms $(\partial R_l/\partial Q_m)Q'^M_{mj}$ and $(\partial R_l/\partial X_p)X'_{pj}$ of Eq. (13) are of this type. In contrast, however, the forward-mode application of ADIFOR produces source code that is prohibitively inefficient for calculating the premultiplication of a large Jacobian matrix by a known input vector (or matrix). This weakness of the forward-mode application of ADIFOR is exactly the strength of the reverse-mode option now available in ADIFOR 3.0. Thus, the proposed new efficient ADII-AV scheme has become possible with this reverse-mode capability. That is, through reverse-mode application of ADIFOR 3.0 it is now feasible to produce (automatically) the source code required for efficient evaluation of the term $\lambda^M_{il}(\partial R_l/\partial Q_m)$ in Eq. (16), that is, the premultiplication of a large Jacobian matrix by a vector.

B. Second-Order Sensitivity Derivatives

The SO SD methods are presented in index notation subject to the following definitions:

$$F''_{ijk} \equiv \frac{d^2 F_i}{db_k\, db_j} = \left(\frac{d^2F}{db^2}\right)_{ijk}, \qquad R''_{ljk} \equiv \frac{d^2 R_l}{db_k\, db_j} = \left(\frac{d^2R}{db^2}\right)_{ljk}$$

$$Q''_{mjk} \equiv \frac{d^2 Q_m}{db_k\, db_j} = \left(\frac{d^2Q}{db^2}\right)_{mjk}, \qquad X''_{pjk} \equiv \frac{d^2 X_p}{db_k\, db_j} = \left(\frac{d^2X}{db^2}\right)_{pjk}$$

$$\lambda'_{ilk} \equiv \frac{d\lambda_{il}}{db_k} = \left(\frac{d\lambda}{db}\right)_{ilk} = -\frac{\partial^2 F_i}{\partial b_k \partial R_l}, \qquad G'_{imk} = \frac{dG_{im}}{db_k} = \left(\frac{dG}{db}\right)_{imk}$$

The following differential operator is also introduced for subsequent notational compactness:

$$\frac{D()}{Db_k} \equiv \frac{\partial()}{\partial Q_n} Q'_{nk} + \frac{\partial()}{\partial X_q} X'_{qk} + \frac{\partial()}{\partial b_k} \tag{17}$$

where repeated indices n and q are summation indices.

Differentiation of the FO forward-mode equations (3) and (4) with respect to the design variables yields SO method 1:

$$F''_{ijk} = \frac{d^2 F_i}{db_k\, db_j} = \frac{\partial F_i}{\partial Q_m} Q''_{mjk} + \frac{\partial F_i}{\partial X_p} X''_{pjk} + \frac{DF'_{ij}}{Db_k} \tag{18}$$

$$R''_{ljk} = \frac{d^2 R_l}{db_k\, db_j} = \frac{\partial R_l}{\partial Q_m} Q''_{mjk} + \frac{\partial R_l}{\partial X_p} X''_{pjk} + \frac{DR'_{lj}}{Db_k} = 0_{ljk} \tag{19}$$

The terms of DF'_{ij}/Db_k and DR'_{lj}/Db_k are many and very complicated; detailed expansion of these terms is provided in the Appendix. Using symmetry of the Hessian $Q''_{mjk} = Q''_{mkj}$, SO method 1 requires $(NDV^2 + NDV)/2$ solutions of the large linear systems of Eq. (19) for Q''_{mjk}; in addition, the method requires NDV solutions of Eq. (4) for the FO SDs Q'_{mj}. SO method 1 was verified for a two-dimensional CFD code in Ref. 3 by ADBB differentiation of the code's existing HDII scheme [Eqs. (12) and (13)] for the FO SDs.

Alternatively, differentiation of the FO reverse-mode, Eqs. (7–9), with respect to the design variables, yields SO method 2:

$$F''_{ijk} = \frac{d^2 F_i}{db_k\, db_j} = \lambda'_{ilk}\left(\frac{\partial R_l}{\partial X_p} X'_{pj} + \frac{\partial R_l}{\partial b_j}\right) + \left(\frac{\partial F_i}{\partial X_p} + \lambda_{il}\frac{\partial R_l}{\partial X_p}\right) X''_{pjk} + \frac{DF'_{ij}}{Db_k} \tag{20}$$

$$G'_{imk} = \frac{dG_{im}}{db_k} = \lambda'_{ilk}\frac{\partial R_l}{\partial Q_m} + \frac{DG_{im}}{Db_k} = 0_{imk} \tag{21}$$

SO method 2 requires $NDV \times NOF$ solutions of the large linear systems of Eq. (21) for λ'_{ilk}; in addition, the method requires NDV solutions of the FO equation (4) for Q'_{mj} plus NOF solutions of the FO equation (9) for λ_{il}. This SO method 2 is eliminated from further consideration because it is unconditionally less computationally efficient than the remaining two SO SD methods.

Introduction of the AV approach within SO method 1 to eliminate Q''_{mjk} yields SO method 3:

$$F''_{ijk} = \left(\frac{\partial F_i}{\partial X_p} + \lambda_{il}\frac{\partial R_l}{\partial X_p}\right) X''_{pjk} + \frac{DF'_{ij}}{Db_k} + \lambda_{il}\frac{DR'_{lj}}{Db_k} \tag{22}$$

SO method 4 is similar and computationally equivalent to SO method 3 and is developed by introduction of the AV approach within SO method 2 to eliminate λ'_{ilk}; the result is the identity

$$\lambda_{il}\frac{DR'_{lj}}{Db_k} = Q'_{mj}\frac{DG_{im}}{Db_k} \tag{23}$$

where SO method 4 uses Eq. (23) to replace equivalent terms within SO method 3. The equivalent SO SD methods 3 and 4 do not require solution of large systems of linear equations for higher-order derivatives such as Q''_{mjk} or λ'_{ilk}. These two SO SD schemes do, however, require solution of both forward-mode and reverse-mode equations (4) and (8) for Q'_{mj} and λ_{il}, respectively. This is a total of only $NDV + NOF$ solutions of large systems of linear equations.

One significant conclusion of the preceding analysis is that SO methods 3 or 4 should be computationally more efficient whenever $NDV^2 + NDV$ is greater than $2 \times NOF$. With typical design problems in aerodynamics, NDV is often much larger than NOF;

typically, NOF is three or less (often only one), whereas NDV is on the order of tens to hundreds. The advantage in favor of method 3 or 4 for SO SDs is then overwhelming because of the NDV^2 term, which dominates. Once both the forward-mode and reverse-mode schemes are in place for calculating the FO SDs, then complete SO SD information is available almost "for free," that is, the SO SD are obtained through an explicit, noniterative calculation. The source code for implementation of method 3 or 4 is constructed "automatically" via BB application of the forward-mode capability of ADIFOR to appropriate pieces of the existing source code from which the FO SDs are obtained. For example, the extremely complex terms DF'_{ij}/Db_k, DR'_{ij}/Db_k, and/or DG_{im}/Db_k (see Appendix) of methods 3 and/or 4 are easily constructed with a forward-mode application of AD.

In Ref. 3, SO methods 3 and 4 were proposed but not actually tested. Consequently, one primary goal of the present study is successful implementation and verification of the highly efficient SO method 3 (or equivalently, method 4); method 3 is actually chosen in this study.

III. Results and Discussion

A. First-Order Sensitivity Derivatives, ADII-AV Method, Model Problem

The proposed ADII-AV method [Eqs. (15) and (16)] has been successfully implemented in a CFD code and verified for accuracy on a simple two-dimensional inviscid internal flow model problem. This CFD code solves the two-dimensional Euler equations by a conventional upwind finite volume approach on a very coarse grid but one that is sufficient for computationally verifying SDs. As expected, when FO SDs computed by the new ADII-AV scheme are compared with SDs computed by a hand-differentiated implementation of Eqs. (15) and (16) (i.e., the HDII-AV approach), the results are the same at convergence, as well as at each I-I step. In addition, the accuracy of the computed SDs has been successfully verified by a finite difference method.

Preliminary timings were conducted on a Sun workstation to evaluate the potential for improved computational efficiency of the new ADII-AV scheme with respect to the ADBB-AV approach of Ref. 6. Computational timing comparisons are given in Table 1, which focuses exclusively on AD performance. Therefore, relative timings are given as CPU time per iteration per grid point per differentiated-aerodynamic-output function. Furthermore, each timing result has been scaled by the comparable timing result obtained from the very efficient hand-differentiated reverse-mode scheme (i.e., the HDII-AV method). Table 1 illustrates that although the new ADII-AV scheme is almost five times slower than the efficient HDII-AV scheme it represents a substantial improvement over results obtained from the straightforward black-box procedure (i.e., ADBB-AV is about eight times slower than HDII-AV).

The improvement in computational efficiency achieved to date is substantial when the reverse-mode application of ADIFOR 3.0 in incremental-iterative form is compared with the black-box approach. Furthermore, the timing result for the ADII-AV scheme is projected to improve by an additional 30% over that reported here. Thus the relative timing given in Table 1 for ADII-AV/HDII-AV is projected to drop from 4.7 to about 3.3. This projection is based on using a strategy where the forward-pass execution of the ADIFOR-enhanced, reverse-mode code will be performed only once (instead of during each iteration) in order to create the required ADIFOR log files. Thereafter, by repeatedly reusing these fixed log files only reverse passes will be repeatedly executed during the iterative solution process for all aerodynamic output functions of interest.

Another important computational concern mitigated by the new ADII-AV method is computer memory, particularly the issue of large disk files created during execution of reverse-mode derivative code created by ADIFOR 3.0. With the black-box (ADBB-AV) approach these large ADIFOR log files (which are created on a forward-pass execution and are read during the reverse pass) will accumulate and become larger with every iteration of the ADIFOR-enhanced flow code. This file growth can rapidly deplete the available disk space, even on the largest computers. In Ref. 6 this difficulty was addressed by development of the iterated reverse-mode scheme, where only the log files for the final forward-pass iteration are stored and used during the subsequent iterative solution for the derivatives. With the ADII-AV approach, however, the required disk space is not as restrictive an issue because it remains fixed and does not accumulate during the iterative solution process. In the present example the total storage requirement for log files with the ADII-AV method is only 64% of that required for a single iteration of the ADBB-AV method.

In addition to the log-file disk memory, required only for the new reverse-mode capability of ADIFOR, there are substantial additional core memory requirements. For the forward mode the core memory increase of the AD-enhanced code is approximately NDV times the core memory requirement of the original (undifferentiated) code. For the reverse mode the corresponding increase is NOF time that of the original code.

Table 1 Relative CPU timing comparison, model problem

Reverse-mode method tested	Relative timing[a]
HDII-AV	1.0
ADII-AV	4.7
ADBB-AV	7.9

[a]CPU time/iteration/grid-point/output function.

B. Second-Order Sensitivity Derivatives, SO Method 3, Airfoil Example

Results are presented subsequently from the successful verification of the proposed efficient noniterative SO method 3 [Eq. (22)] for computing SO SDs. The example problem is steady transonic inviscid flow over a NACA 0012 airfoil with freestream Mach number M_∞ 0.80 and angle of attack α 1 deg. The two-dimensional Euler equations are solved on a Sun workstation in double precision using a conventional finite volume upwind flux-vector-splitting scheme. A C-mesh computational grid is used with dimensions 129 × 33 grid points. High-quality lift-corrected boundary conditions are used at the far-field boundary, which is placed approximately five chord lengths from the surface of the airfoil.

In the present example derivatives of three aerodynamic output functions are considered: C_L, C_D, and C_M (i.e., coefficients of lift, wave drag, and pitching moment, respectively). The computed steady-state values of these aerodynamic force coefficients are given in Table 2. Note that the number of digits shown in Table 2 (and also Tables 3 and 4) is to illustrate consistency rather than accuracy. In addition, derivatives with respect to three aerodynamic input variables are considered. They are g (a geometric-shape variable), α, and M_∞. The geometric-shape variable g is a single arbitrarily selected y coordinate of the computational grid on the surface of the airfoil: simple, but one that is sufficient for verifying geometric-shape SDs.

Calculation of SO SDs by SO method 3 requires that all FO SDs are calculated first using both the forward-mode [Eqs. (3) and (4)] and the reverse-mode [Eqs. (7) and (8)] approaches. The calculated FO SDs from a hand-differentiated incremental-iterative (HDII) implementation of these two approaches are presented in Table 3, where the results are seen to agree, as expected. The FO SDs presented in Table 3 have been thoroughly verified for consistency through a meticulous implementation of the method of central finite differences, where agreement to six significant digits or greater is noted in all comparisons.

Table 2 Aerodynamic force coefficients, airfoil example

Coefficient	Value
C_L	$+0.2830659E+00$
C_D	$+0.2070493E-01$
C_M	$-0.2876639E-01$

Table 3 First-order sensitivity derivatives, HDII method, airfoil example

Derivative	b_j	Forward mode	Reverse mode
$\frac{dC_L}{db_j}$	g	$+0.1405406E+00$	$+0.1405406E+00$
	α	$-0.1087323E-01$	$-0.1087323E-01$
	M_∞	$-0.4672729E-01$	$-0.4672729E-01$
$\frac{dC_D}{db_j}$	g	$+0.1761807E+02$	$+0.1761807E+02$
	α	$+0.1158625E+01$	$+0.1158625E+01$
	M_∞	$-0.2382580E+01$	$-0.2382580E+01$
$\frac{dC_M}{db_j}$	g	$+0.3171492E+01$	$+0.3171492E+01$
	α	$+0.5955598E+01$	$+0.5955598E+01$
	M_∞	$-0.1603002E+01$	$-0.1603002E+01$

Table 4 Second-order sensitivity derivatives, SO method 3, airfoil example

Derivative	b_j/b_k	g	α	M_∞
$\frac{d^2C_L}{db_j db_k}$	g	$+0.248807E+03$	$+0.250402E+03$	$+0.205825E+03$
	α	$+0.250402E+03$	$+0.184277E+05$	$+0.160858E+05$
	M_∞	$+0.205825E+03$	$+0.160858E+05$	$+0.133087E+05$
$\frac{d^2C_D}{db_j db_k}$	g	$+0.71777E+02$	$+0.134379E+02$	$+0.101304E+02$
	α	$+0.134379E+02$	$+0.959310E+03$	$+0.804021E+03$
	M_∞	$+0.101304E+02$	$+0.804021E+03$	$+0.662088E+03$
$\frac{d^2C_M}{db_j db_k}$	g	$-0.663590E+02$	$-0.602441E+02$	$-0.490399E+02$
	α	$-0.6024415+02$	$-0.449198E+04$	$-0.386943+04$
	M_∞	$-0.490399E+02$	$-0.386943E+04$	$-0.320512E+04$

Table 5 Relative CPU timings: complete SO method 3, airfoil example

Computational procedure	% of total
Nonlinear flow, Eqs. (1) and (2)	5.4
Forward-mode FO SDs, Eqs. (3) and (4)	25.5
Reverse-mode FO SDs, Eqs. (7) and (9)	69.0
SO SDs, Eq. (22)	0.1
Total	100.0

The SO method 3 is implemented by application (in the forward mode) of ADIFOR to appropriate pieces of the FORTRAN code used earlier for hand-differentiated forward-mode calculation of the FO SDs. The calculated SO SDs from this implementation of SO method 3 are presented in Table 4. The SO SDs of Table 4 have been thoroughly verified for consistency through a meticulous application of central finite differences applied to FO SDs obtained by the hand-differentiated methods already described. Agreement to five significant digits or better is noted in this verification study for all SO SDs reported in Table 4. This verification study was not conducted using finite differences applied to the original nonlinear flow code; that approach has been documented to be vulnerable to severe numerical inaccuracy when SO SDs are calculated.[3] The symmetry of the calculated SO SDs shown in Table 4 is expected and results from the computations performed, that is, no derivative symmetry was explicitly imposed on the problem.

For the present airfoil example problem Table 5 illustrates (in terms of percentages of the total) the breakdown of relative CPU timings for the computational steps of SO method 3 procedure for calculating the SO SDs. Not included in Table 5 is the CPU time for the grid generation and the grid-sensitivity derivatives, negligible for this particular two-dimensional example. Table 5 illustrates clearly the computational efficiency of the SO method 3 for SO SDs. Recall that results of the present example are for three aerodynamic output functions and three input (design) variables, where the computational work of the forward-mode and reverse-mode procedures for FO SDs should be approximately equal (in theory, for hand-differentiated code, as used here). In this example, however, Table 5 reveals that the reverse mode was much more costly than the forward mode; apparently the three linear systems for the reverse mode are stiffer than the three for the forward mode. This characteristic of the adjoint equations has been observed by others.[7] As expected, Table 5 shows that using an ADIFOR-assisted second differentiation SO SDs can be obtained extremely fast, if one already has both the forward-mode and reverse-mode FO SDs.

IV. Conclusions

An efficient incremental-iterative approach for differentiating advanced CFD flow codes has been successfully demonstrated on a two-dimensional inviscid model problem. The method employs the reverse-mode capability of the automatic-differentiation software tool ADIFOR 3.0 and has been shown to yield consistent first-order aerodynamic sensitivity derivatives. A substantial reduction in CPU time and computer memory has been demonstrated by comparison with results from a straightforward, black-box reverse-mode application of ADIFOR 3.0 to the same flow code.

A computationally efficient ADIFOR-assisted procedure for consistent second-order aerodynamic sensitivity derivatives has been successfully verified on an inviscid transonic lifting airfoil example problem. Accurate second derivatives (i.e., the complete Hessian matrices) of lift, wave drag, and pitching-moment coefficients with respect to geometric shape, angle of attack, and freestream Mach number have been calculated. With the present procedure second-order derivatives are now computationally feasible, at least in two dimensions. The computation of second-order derivatives in three dimensions appears to be within reach, but remains to be investigated.

This second-order method requires that first-order derivatives be calculated using both the forward (direct) and reverse (adjoint) procedures; then, second-order derivatives can be obtained in a noniterative calculation that is computationally very efficient. An ADIFOR differentiation is used to generate a number of required second-order terms in this noniterative calculation. If one already has either forward (NDV solutions) or reverse (NOF solutions) FO SDs, then upon obtaining the other FO SDs (NOF or NDV additional solutions, respectively) one calculates all of the SO SDs ($NOF \times NDV^2$ derivatives) very efficiently.

Appendix: Expansion of Terms

In this Appendix the terms DF'_{ij}/Db_k, DR'_{lj}/Db_k, and DG_{im}/Db_k are expanded using the index notation already established. The expansion of DF'_{ij}/Db_k is

$$\begin{aligned}\frac{DF'_{ij}}{Db_k} &= \frac{\partial F'_{ij}}{\partial Q_n}Q'_{nk} + \frac{\partial F'_{ij}}{\partial X_q}X'_{qk} + \frac{\partial F'_{ij}}{\partial b_k}\\ &= \frac{\partial^2 F_i}{\partial Q_n \partial Q_m}Q'_{mj}Q'_{nk} + \frac{\partial^2 F_i}{\partial Q_n \partial X_p}X'_{pj}Q'_{nk} + \frac{\partial^2 F_i}{\partial Q_n \partial b_j}Q'_{nk}\\ &\quad + \frac{\partial^2 F_i}{\partial X_q \partial Q_m}Q'_{mj}X'_{qk} + \frac{\partial^2 F_i}{\partial X_q \partial X_p}X'_{pj}X'_{qk} + \frac{\partial^2 F_i}{\partial X_q \partial b_j}X'_{qk}\\ &\quad + \frac{\partial^2 F_i}{\partial b_k \partial Q_m}Q'_{mj} + \frac{\partial^2 F_i}{\partial b_k \partial X_p}X'_{pj} + \frac{\partial^2 F_i}{\partial b_k \partial b_j}\end{aligned} \tag{A1}$$

In Eq. (A1), the indices i, j, and k are free and repeated indices n, m, p, and q are summed. The terms of DR'_{lj}/Db_k are obtained from Eq. (A1) by replacing everywhere F'_{ij} with R'_{lj} and F_i with R_l (and thus l replaces i as a free index in the resulting expressions).

Finally, the expansion of the terms for DG_{im}/Db_k is

$$\begin{aligned}\frac{DG_{im}}{Db_k} &= \frac{\partial G_{im}}{\partial Q_n}Q'_{nk} + \frac{\partial G_{im}}{\partial X_q}X'_{qk} + \frac{\partial G_{im}}{\partial b_k}\\ &= \lambda_{il}\frac{\partial^2 R_l}{\partial Q_n \partial Q_m}Q'_{nk} + \frac{\partial^2 F_i}{\partial Q_n \partial Q_m}Q'_{nk} + \lambda_{il}\frac{\partial^2 R_l}{\partial X_q \partial Q_m}X'_{qk}\\ &\quad + \frac{\partial^2 F_i}{\partial X_q \partial Q_m}X'_{qk} + \lambda_{il}\frac{\partial^2 R_l}{\partial b_k \partial Q_m} + \frac{\partial^2 F_i}{\partial b_k \partial Q_m}\end{aligned} \tag{A2}$$

In Eq. (A2) the indices i, m, and k are free, and repeated indices l, n, and q are summed.

Acknowledgments

The first author was partially supported by an American Society for Engineering Education grant during the summer of 2000 at NASA Langley Research Center and by NASA Cooperative Agreement NCC1-01-015. The authors thank Thomas A. Zang of the Multidisciplinary Optimization Branch of NASA Langley Research Center for his encouragement and long-term support of this work. Gratitude is expressed to Mike Fagan of Rice University, Houston, Texas, for helpful discussions and advice on the use of ADIFOR 3.0.

References

[1]Newman, J. C., III, Taylor, A. C., III, Barnwell, R. W., Newman, P. A., and Hou, G. J.-W., "Overview of Sensitivity Analysis and Shape Optimization for Complex Aerodynamic Configurations," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 87–96.

[2]Newman, P. A., Hou, G. W., Jones, H. E., Taylor, A. C., III, and Korivi, V. M., "Observations on Computational Methodologies for Use in Large-Scale Gradient-Based Multidisciplinary Design Incorporating Advanced CFD Codes," *Proceedings of the Fourth AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization*, Pt. 1, AIAA, Washington, DC, 1992, pp. 531–542.

[3]Sherman, L. L., Taylor, A. C., III, Green, L. L., Newman, P. A., Hou, G. J.-W, and Korivi, V. M., "First- and Second-Order Aerodynamic Sensitivity Derivatives via Automatic Differentiation with Incremental Iterative Method," *Journal of Computational Physics*, Vol. 129, Dec. 1996, pp. 307–331.

[4]Taylor, A. C., III, and Oloso, A., "Aerodynamics Design Optimization Using Advanced CFD Codes, Automatic Differentiation, and Parallel Computing," *Computational Fluid Dynamics Review 1998*, Vol. 1, edited by M. Hafez and K. Oshima, World Scientific, Singapore, 1998, Article 31, pp. 560–571.

[5]Taylor, A. C., III, and Oloso, A., "Aerodynamic Design Sensitivities by Automatic Differentiation," AIAA Paper 98-2536, June 1998.

[6]Carle, A., Fagan, M., and Green, L., "Preliminary Results from the Application of Automatic Adjoint Code Generation to CFL3D," *Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Pt. 2, AIAA, Reston, VA, 1998, pp. 807–817.

[7]Carle, A., and Fagan, M., "Overview of ADIFOR 3.0," Dept. of Computational and Applied Mathematics, Rice Univ., CAAM-TR 00-02, Houston, TX, Jan. 2000.

[8]Oloso, A. O., and Taylor, A. C., III, "Aerodynamic Shape-Sensitivity Analysis and Design Optimization on the IBM-SP2," *Proceedings of the 15th AIAA Applied Aerodynamics Conference*, Pt. 1, AIAA, Reston, VA, 1997, pp. 481–488.

[9]Taylor, A. C., III, Oloso, A., and Newman, J. C., III, "CFL3D.ADII (Version 2.0): An Efficient Accurate General-Purpose Code for Flow Shape-Sensitivity Analysis," *Proceedings of the 15th AIAA Applied Aerodynamics Conference*, Pt. 1, AIAA, Reston, VA, 1997, pp. 12–16.

[10]Park, M. M., Green, L. L., Montgomery, R. C., and Raney, D. L., "Determination of Stability and Control Derivatives Using Computational Fluid Dynamics and Automatic Differentiation," *Proceedings of the 17th AIAA Applied Aerodynamics Conference*, AIAA, Reston, VA, 1999, pp. 268–285.

[11]Putko, M. M., Taylor, A. C., III, Newman, P. A., and Green, L. L., "Approach for Input Uncertainty Propagation and Robust Design in CFD Using Sensitivity Derivatives," *Journal of Fluids Engineering*, Vol. 124, No. 2, 2002, pp. 60–69.

[12]Lewis, R. M., "The Adjoint Approach in a Nutshell," *SIAM Activity Group on Optimization Views-and-News*, Vol. 11, No. 2, 2000, pp. 9–12.

[13]Korivi, V. M., Taylor, A. C., III, Newman, P. A., Hou, G. J.-W., and Jones, H. E., "An Approximately Factored Incremental Strategy for Calculating Consistent Discrete Aerodynamic Sensitivity Derivatives," *Journal of Computational Physics*, Vol. 113, No. 2, 1994, pp. 336–346.

A. Plotkin
Associate Editor