AIAA 2001-2529

Some Advanced Concepts in Discrete Aerodynamic Sensitivity Analysis

Arthur C. Taylor III *

Lawrence L. Green**

Perry A. Newman***

Michele M. Putko****

Department of Mechanical Engineering
Old Dominion University
Norfolk, VA 23529

Multidisciplinary Optimization Branch
NASA Langley Research Center
Hampton, VA 23681

Presented at the AIAA 15th Computational Fluid Dynamics Conference
June 11-14, 2001, Anaheim, CA

*Associate Professor, Old Dominion University, Norfolk, VA 23529
**Research Scientist, NASA Langley Research Center, Hampton VA 23681, Senior Member AIAA, l.l.green@nasa.larc.gov
***Senior Research Scientist, NASA Langley Research Center, Hampton VA 23681
****LTC, US Army, PhD Candidate, Old Dominion University, Norfolk, VA 23529

AIAA 2001-2529

SOME ADVANCED CONCEPTS IN DISCRETE AERODYNAMIC SENSITIVITY ANALYSIS

Arthur C. Taylor III [*] **Lawrence L. Green** [†]

Perry A. Newman [‡] **Michele M. Putko** [§]

Department of Mechanical Engineering
Old Dominion University
Norfolk, VA 23529

Multidisciplinary Optimization Branch
NASA Langley Research Center
Hampton, VA 23681

Abstract

An efficient incremental-iterative approach for differentiating advanced flow codes is successfully demonstrated on a 2D 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 straight-forward, 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 non-iterative 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

1.0 Introduction

Computing sensitivity derivatives (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 present study will essentially build on earlier studies[3,6] in an effort to exploit the full potential of the latest version of ADIFOR 3.0[7] for obtaining SDs from CFD codes.

In Ref. 2, a strategy was first proposed and later successfully demonstrated in Ref. 3, whereby AD was applied to a CFD code in incremental-iterative (I-I) form. This hybrid scheme (known as the ADII method) was designed to achieve in part the computational efficiency of a hand-differentiated (HD) approach, and at the same time capture identical accuracy while maintaining (at least in part) the ease-of-implementation of a straightforward black-box (BB) application of AD. The 2D effort of Ref. 3 was later extended to the 3D code CFL3D, including "in-parallel" computation of the de-

[*] Associate Professor, Old Dominion University, Norfolk, VA 23529
[†] Research Scientist, NASA Langley Research Center, Hampton, VA 23681, Senior Member AIAA
[‡] Senior Research Scientist, NASA Langley Research Center, Hampton, VA 23681
[§] LTC, US Army, PhD Candidate, Old Dominion University, Norfolk, VA 23529

rivatives[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 with 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 by application[6] to an "in-parallel" version of CFL3D and application[10] to a sequential linear aerodynamics code, resulting 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 over only the last function iteration tree.

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.

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. 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 2D 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; it should also be noted here that 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; therefore, with the availability of ADIFOR 3.0 and the new ADII-AV scheme, the door has been opened for implementation and testing of the most efficient SO SD scheme for CFD codes. The results of this effort to date are reported herein. In a companion study[11] these efficiently computed SO SDs have been used to demonstrate an approach for CFD input uncertainty propagation and robust design optimization for a quasi-1D flow application.

2.0 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 (Ref. 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

$$\boldsymbol{F} = \boldsymbol{F}(\boldsymbol{Q}(\boldsymbol{b}), \boldsymbol{X}(\boldsymbol{b}), \boldsymbol{b}) \qquad (1)$$

(aerodynamic output functions)

$$\boldsymbol{R} = \boldsymbol{R}(\boldsymbol{Q}(\boldsymbol{b}), \boldsymbol{X}(\boldsymbol{b}), \boldsymbol{b}) = \boldsymbol{0} \qquad (2)$$

(nonlinear state equations)

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.

2.1 First-Order Sensitivity Derivatives

By the following preliminary definitions, index (summation) notation is now introduced. (This notation will be necessary to avoid subsequent ambiguity when the SO SD methods are presented.)

$$F'_{ij} \equiv \frac{dF_i}{db_j} = \left(\frac{d\boldsymbol{F}}{d\boldsymbol{b}}\right)_{ij} \qquad R'_{lj} \equiv \frac{dR_l}{db_j} = \left(\frac{d\boldsymbol{R}}{d\boldsymbol{b}}\right)_{lj}$$

$$Q'_{mj} \equiv \frac{dQ_m}{db_j} = \left(\frac{d\boldsymbol{Q}}{d\boldsymbol{b}}\right)_{mj} \qquad X'_{pj} \equiv \frac{dX_p}{db_j} = \left(\frac{d\boldsymbol{X}}{d\boldsymbol{b}}\right)_{pj}$$

The forward-mode (direct) approach for calculating first-order (FO) SDs is developed by differentiation of Eq. (1) and (2) with respect to the design variables; the result is

$$F'_{ij} = \frac{dF_i}{db_j} = \frac{\partial F_i}{\partial Q_m} Q'_{mj} + \frac{\partial F_i}{\partial X_p} X'_{pj} + \frac{\partial F_i}{\partial b_j} \qquad (3)$$

$$R'_{lj} = \frac{dR_l}{db_j} = \frac{\partial R_l}{\partial Q_m} Q'_{mj} + \frac{\partial R_l}{\partial X_p} X'_{pj} + \frac{\partial R_l}{\partial b_j} = 0_{lj} \quad (4)$$

In the above, 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 in a non-conventional manner starting with application of the chain rule

$$\frac{\partial F_i}{\partial R_l} \frac{\partial R_l}{\partial Q_m} = \frac{\partial F_i}{\partial Q_m} \quad (5)$$

With Eq. (5), it then follows from eq. (4) that

$$\begin{aligned} \frac{\partial F_i}{\partial Q_m} Q'_{mj} &= \frac{\partial F_i}{\partial R_l} \frac{\partial R_l}{\partial Q_m} Q'_{mj} \\ &= -\frac{\partial F_i}{\partial R_l} \left(\frac{\partial R_l}{\partial X_p} X'_{pj} + \frac{\partial R_l}{\partial b_j} \right) \end{aligned} \quad (6)$$

and Eq. (3) becomes

$$\begin{aligned} F'_{ij} = \frac{dF_i}{db_j} &= -\frac{\partial F_i}{\partial R_l} \left(\frac{\partial R_l}{\partial X_p} X'_{pj} + \frac{\partial R_l}{\partial b_j} \right) \\ &+ \frac{\partial F_i}{\partial X_p} X'_{pj} + \frac{\partial F_i}{\partial b_j} \end{aligned} \quad (7)$$

Comparison of Eq. (5) and (7) with a more conventional development of the reverse-mode method yields the identity

$$-\frac{\partial F_i}{\partial R_l} = \lambda_{il} \quad (8)$$

where λ_{il} is more commonly called the "adjoint variable." Using Eq. (8), the more conventional presentation of Eq. (5) is

$$G_{im} \equiv \lambda_{il} \frac{\partial R_l}{\partial Q_m} + \frac{\partial F_i}{\partial Q_m} = 0_{im} \quad (9)$$

One objective of this particular development of the AV method for aerodynamic SDs is to ensure that the relationship given by Eq. (8) is clearly understood.

The I-I strategies for solving the preceding equations are reviewed here; additional detail is found in Ref. 3 and elsewhere. The I-I method for solving the nonlinear flow of Eq. (2) is

$$Q_m^{N+1} = Q_m^N - P_{ml}^N R_l^N \quad (10)$$

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

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

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 Eq. (4) is

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

where superscript M is the FO SD iteration index, and

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

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

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

Clearly ADII (Eq. (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 due to user intervention in the application of AD. With ADII

1. The operator P_{ml}^N can be evaluated only once 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

$$\lambda_{il}^{M+1} = \lambda_{il}^{M} - P_{ml} G_{im}^{M} \tag{15}$$

where

$$G_{im}^{M} \equiv \lambda_{il}^{M} \frac{\partial R_l}{\partial Q_m} + \frac{\partial F_i}{\partial Q_m} \tag{16}$$

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 that explained previously 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 the particular aerodynamic inputs and outputs of interest.

The forward-mode application of ADIFOR has always been very effective in constructing FORTRAN source code for the computationally efficient, repeated calculation of the vector (or matrix) product that results from the post-multiplication 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 $\frac{\partial R_l}{\partial Q_m} Q'^{M}_{mlj}$ and $\frac{\partial R_l}{\partial X_p} X'_{pj}$ of Eq. (13) are of this type. In contrast, however, the forward-mode application of ADIFOR constructs source code which is prohibitively inefficient for calculating the pre-multiplication 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 construct (automatically) the source code required for efficient evaluation of the term $\lambda_{il}^{M} \frac{\partial R_l}{\partial Q_m}$ in Eq. (16).

2.2 Second-Order Sensitivity Derivatives

The SO SD methods are presented subsequently using the index notation and beginning with the following preliminary definitions:

$$F''_{ijk} \equiv \frac{d^2 F_i}{db_k db_j} = \left(\frac{d^2 \boldsymbol{F}}{d\boldsymbol{b}^2} \right)_{ijk}$$

$$R''_{ljk} \equiv \frac{d^2 R_l}{db_k db_j} = \left(\frac{d^2 \boldsymbol{R}}{d\boldsymbol{b}^2} \right)_{ljk}$$

$$Q''_{mjk} \equiv \frac{d^2 Q_m}{db_k db_j} = \left(\frac{d^2 \boldsymbol{Q}}{d\boldsymbol{b}^2} \right)_{mjk}$$

$$X''_{pjk} \equiv \frac{d^2 X_p}{db_k db_j} = \left(\frac{d^2 \boldsymbol{X}}{d\boldsymbol{b}^2} \right)_{pjk}$$

$$\lambda'_{ilk} \equiv \frac{d\lambda_{il}}{db_k} = \left(\frac{d\boldsymbol{\lambda}}{d\boldsymbol{b}} \right)_{ilk} = -\frac{\partial^2 F_i}{\partial b_k \partial R_l}$$

$$G'_{imk} = \frac{dG_{im}}{db_k} = \left(\frac{d\boldsymbol{G}}{d\boldsymbol{b}} \right)_{imk}$$

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 Eq. (3) and (4) with respect to the design variables yields SO Method 1

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

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

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 2D CFD code in Ref. 3 by ADBB differentiation of the code's existing HDII scheme (Eq. (12) and (13)) for the FO SDs.

Alternatively, differentiation of the FO reverse-mode, Eq. (7), (8), and (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 Eq. (4) for Q'_{mj} plus NOF solutions of the FO Eq. (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. It is important to note that 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 Eq. (4) and (9) 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 Method 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 very much larger than NOF, and the advantage in favor of Method 3 or 4 for SO SDs is then all the greater. 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"; i.e., the SO SD are obtained through an explicit, non-iterative calculation. The source code for implementation of Methods 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'_{lj} / 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.

3.0 Results

3.1 First-Order Sensitivity Derivatives, ADII-AV Method, Model Problem

The proposed ADII-AV method (Eq. (15) and (16)) has been successfully implemented in a CFD code and verified for accuracy on a 2D inviscid internal flow

problem. This CFD code solves the 2D Euler equations by a conventional upwind finite-volume approach on a very coarse grid; but, one that is sufficient for 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 Eq. (15) and (16) (i.e., the HDII-AV approach), the results are the same, iteration-by-iteration. 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. 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. Relative CPU Timing Comparison

Reverse-Mode Method Tested	Relative Timing*
ADII-AV / HDII-AV	4.7
ADBB-AV / HDII-AV	7.9
*CPU Time/iteration/grid-point/output-function	

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 straight-forward black-box procedure (i.e., ADBB-AV is about eight times slower than HDII-AV).

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 percent of that required for a single iteration of the ADBB-AV method.

3.2 Second-Order Sensitivity Derivatives, SO Method 3, Airfoil Example

Results are presented subsequently from the successful verification of the proposed efficient non-iterative 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) 1.0 degree. The 2D 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.

Table 2. Aerodynamic Force Coefficients

C_L	+0.2830659 E+00
C_D	+0.2070493 E-01
C_M	-0.2876639 E-01

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 simply a single arbitrarily-selected "y" coordinate of the computational grid on the surface of the airfoil.

Calculation of SO SDs by SO Method 3 requires that all FO SDs are calculated first using both the forward-mode (Eq. (3) and (4)) and the reverse-mode (Eq. (6) and (7)) 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.

Table 3. First-Order Sensitivity Derivatives

	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

	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

Computational Procedure	% of Total
Nonlinear Flow, Eq. (1) and (2)	5.4
Forward-Mode FO SDs, Eq. (3) and (4)	25.5
Reverse-Mode FO SDs, Eq. (7) and (9)	69.0
SO SDs, Eq. (22)	0.1
Total	100.0

The FO SDs presented in Table 3 have been thoroughly verified for accuracy through a meticulous implementation of the method of central finite-differences, where agreement to six significant digits or greater is noted in all comparisons.

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 accuracy through a meticulous application of central finite-differences applied to FO SDs obtained by the hand-differentiated methods previously 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 SD shown in Table 4 is expected and results from the computations performed; i.e., 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 important steps of SO Method 3 for the SO SDs. (Not included in Table 5 is the CPU time for the grid generation and the grid-sensitivity derivatives.) 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. As expected, Table 5 shows that using an ADIFOR-assisted second differentiation, SO SD can be obtained extremely fast, if one already has both the forward-mode and reverse-mode FO SD.

4.0 Conclusions

An efficient incremental-iterative approach for differentiating advanced CFD flow codes has been successfully demonstrated on a 2D 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 accurate first-order aerodynamic sensitivity derivatives. A substantial reduction in CPU time and computer memory has been demonstrated by comparison with results from a straight-forward, black-box reverse-mode application of ADIFOR 3.0 to the same flow code.

A computationally efficient ADIFOR-assisted procedure for accurate 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. Second-order derivatives are now computationally feasible, at least in 2D.

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 non-iterative calculation that is computationally very efficient. An ADIFOR differentiation is used to generate a number of required second-order terms (see Appendix) in this non-iterative 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.

5.0 Future Direction

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, work is presently in-progress where the timing result for the ADII-AV scheme is projected to improve by an additional 30 percent over that reported herein. Thus the relative timing given in Table 1 for ADII-AV / HDII-AV is projected to drop from 4.7 to about 3.3, or better (close to the expected AD limit). This educated projection comes from a proposed 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.

Extracting accurate sensitivity derivatives from advanced flow codes is a challenging, computationally intensive task, particularly in 3D, even for the first-order derivatives. With the use of SO Method 3 and ADIFOR, however, accurate second-order derivatives are now computationally feasible, at least in 2D; per-

haps even 3D is within reach through future "in-parallel" implementation.

6.0 Acknowledgements

The authors wish to thank Dr. Thomas A. Zang of the MDO branch of NASA-Langley Research Center for his encouragement and long-term support of this work. Gratitude is expressed to Dr. Mike Fagan of Rice University for helpful discussions and advice on the use of ADIFOR 3.0. The first author was partially supported by an ASEE grant during the Summer 2000 at NASA Langley. The word-processing efforts of Mrs. Diane Mitchell and Mrs. Teresa Taylor are gratefully acknowledged.

7.0 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," AIAA Journal of Aircraft, Vol. 36, No. 1, Jan.-Feb. 1999, pp. 87-96 (an invited contribution).

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/OAL Symposium on Multidisciplinary Analysis and Optimization, pp. 531-542, Sept. 21-23 1992, Cleveland OH, AIAA Paper 92-4753.

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, Article 31 of Vol.1, pp. 560-571; Editors: M. Hafez and K. Oshima, published by World Scientific Publishing Co., 1998 (an invited contribution).

5. Taylor, A.C.III; and Oloso, A.; "Aerodynamic Design Sensitivities by Automatic Differentiation," AIAA Paper 98-2536, June 1998 (an invited paper).

6. Carle, A.; Fagan, M.; and Green, L.; "Preliminary Results from the Application of Automatic Adjoint Code Generation to CFL3D," Proceedings of the Seventh AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization; Sept. 1998, St. Louis MO; AIAA Paper 98-4807.

7. Carle, A.; and Fagan, M.; "Overview of ADIFOR 3.0," Department of Computational an Applied Mathematics, Rice University, CAAM-TR 00-02, Jan. 2000.

8. Oloso, A.; and Taylor, A.C. III; "Aerodynamic Shape-Sensitivity Analysis and Design Optimization on the IBM-SP2 Using the 3D Euler Equations," Proceedings of the 15th AIAA Applied Aerodynamics Conference; June 23-25 1997, Atlanta GA; AIAA Paper 97-2204.

9. Taylor, A.C. III; Oloso, A.; and Newman, J.C. III; "CFL3D.ADII (v. 2.0): An Efficient Accurate General-Purpose Code for Flow Shape-Sensitivity Analysis," Proceedings of the 15th AIAA Applied Aerodynamics Conference; June 23-25 1997, Atlanta GA; AIAA Paper 97-2275.

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; June 28-July 1 1999, Norfolk VA; AIAA Paper 99-3136.

11. Putko, M. M.; Newman, P. A.; Taylor, A. C. III; Green, L. L., "Approach for Uncertainty Propagation and Robust Design in CFD Using Sensitivity Derivatives," Proceedings of the 15th AIAA Computational Fluid Dynamics Conference; June 11-14 2001, Anaheim CA; AIAA Paper 2001-2528.

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

8.0 Appendix

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

$$\frac{DF'_{ij}}{Db_k} = \frac{\partial F'_{ij}}{\partial Q_n} Q'_{nk} + \frac{\partial F'_{ij}}{\partial X_q} X'_{qk} + \frac{\partial F'_{ij}}{\partial b_k}$$

$$= \frac{\partial^2 F_i}{\partial Q_n \partial Q_m} Q'_{mj} Q'_{nk} + \frac{\partial^2 F_i}{\partial Q_n \partial X_p} X'_{pj} Q'_{nk} + \frac{\partial^2 F_i}{\partial Q_n \partial b_j} Q'_{nk}$$

$$+ \frac{\partial^2 F_i}{\partial X_q \partial Q_m} Q'_{mj} X'_{qk} + \frac{\partial^2 F_i}{\partial X_q \partial X_p} X'_{pj} X'_{qk} + \frac{\partial^2 F_i}{\partial X_q \partial b_j} X'_{qk}$$

$$+ \frac{\partial^2 F_i}{\partial b_k \partial Q_m} Q'_{mj} + \frac{\partial^2 F_i}{\partial b_k \partial X_p} X'_{pj} + \frac{\partial^2 F_i}{\partial b_k \partial b_j} \qquad (24)$$

In Eq. (24), 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. (24) 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

$$\frac{DG_{im}}{Db_k} = \frac{\partial G_{im}}{\partial Q_n} Q'_{nk} + \frac{\partial G_{im}}{\partial X_q} X'_{qk} + \frac{\partial G_{im}}{\partial b_k}$$

$$= \lambda_{il} \frac{\partial^2 R_l}{\partial Q_n \partial Q_m} Q'_{nk} + \frac{\partial^2 F_i}{\partial Q_n \partial Q_m} Q'_{nk}$$

$$+ \lambda_{il} \frac{\partial^2 R_l}{\partial X_q \partial Q_m} X'_{qk} + \frac{\partial^2 F_i}{\partial X_q \partial Q_m} X'_{qk} \qquad (25)$$

$$+ \lambda_{il} \frac{\partial^2 R_l}{\partial b_k \partial Q_m} + \frac{\partial^2 F_i}{\partial b_k \partial Q_m}$$

In Eq. (25), the indices i, m, and k are free, and repeated indices l, n, and q are summed.