Users' Experience with ADIFOR 2.0*

Argonne Preprint ANL/MCS-P589-0496
CRPC Technical Report CRPC-TR96642

C. Bischof† and A. Carle‡

Abstract

In July 1995, the ADIFOR 2.0 system for automatic differentiation of Fortran was made available to the academic and commercial communities via the World Wide Web. By January 1996, we had received and processed over one hundred requests for ADIFOR 2.0. In this paper, we describe some of the experiences of users of the system that should be interesting to developers of automatic differentiation tools.

1 Introduction

ADIFOR 1.0 was completed in June 1993 and initially made available to the public through accounts at Argonne and Rice or direct collaborations with the developers of the system. Even with such a limited distribution, it was successfully employed in many different areas of science and engineering, including aeronautical multidisciplinary design optimization [2, 31], aeronautical computational fluid dynamics [6, 7, 15, 19, 25, 26], weather modeling [12, 14, 28, 29], groundwater contaminant transport [11, 32], aquifer modeling [17, 21], structural engineering [16], statistics [13], mechanical system design [20], power networks [23], reaction modeling [27], and large-scale numerical optimization [1, 8, 30].

In July 1995, the ADIFOR 2.0 system, a substantial reimplementation and extension of ADIFOR 1.0, was made available for distribution via the World Wide Web. To retrieve the software, requestors filled out and submitted an electronic request form, downloaded and signed a license permitting academic use and commercial evaluation of the system, and either faxed or mailed the signed license back to us. We then gave the requestors access to password-protected Web pages with links to Unix tar files containing all of the components of the ADIFOR 2.0 system. Requests could be processed at either Argonne or Rice, and files could be downloaded from either site. By January 1996, we had received and processed over one hundred requests for ADIFOR from users in a wide range of areas including the following: bifurcation analysis in dynamical systems, maximum likelihood fitting of stochastic models, solution of differential algebraic equations, solution of ordinary differential equations, solution of

*This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38, by the National Aerospace Agency under Purchase Order L25935D and Cooperative Agreement No. NCC 1 212, and by the National Science Foundation, through the Center for Research on Parallel Computation, under Cooperative Agreement No. CCR-9120008.

†Mathematics and Computer Science Division, Argonne National Laboratory, 9700 S. Cass Avenue, Argonne, IL 60439-4844, bischof@mcs.anl.gov.

‡Center for Research on Parallel Computation, Rice University MS 41, 6100 S. Main Street, Houston, TX 77005, carle@cs.rice.edu.

partial differential equations, real-time control and optimization, parameter estimation for groundwater flow and transport, nonlinear structural analysis and optimization, multidisciplinary design optimization, inverse modeling in hydrodynamics, radioactive waste site characterization, cyclotron modeling, low-energy nuclear physics reaction theory, numerical weather prediction, reactive flow modeling, earthquake ground motion modeling, atmospheric pollution modeling, chemical reactor modeling, transistor models in circuit simulation, and aquifer parameter estimation.

2 An Overview of ADIFOR 2.0

The ADIFOR 2.0 system provides automatic differentiation (AD) of Fortran 77 programs for first-order derivatives [4]. It implements AD by using a source code transformation approach; that is, given a Fortran subroutine (or collection of subroutines) for a function f, ADIFOR produces Fortran 77 subroutines for the computation of the derivatives of this function. Derivatives are computed by using a "statement-level hybrid mode" in which the forward mode is used to propagate derivatives globally through the program, and the reverse mode is used to propagate derivatives within each assignment statement. This statement-level hybrid mode tends to be more efficient than the normal forward mode or a finite-difference approximation when derivatives are computed for multiple independent variables. In addition, this mode has the predictable storage and runtime requirements expected from the forward mode.

The ADIFOR 2.0 system offers the following features:

Full Fortran 77 Support. The ADIFOR 2.0 preprocessor supports all of Fortran 77 plus common extensions such as `DOUBLE COMPLEX`, `INCLUDE` statements, `IMPLICIT NONE`, and `NAMELIST`. In addition, codes that mix single-precision and double-precision real-valued or complex-valued data are now fully supported.

Flexible Intrinsic Handler. The ADIntrinsics 1.0 system provides for various reporting levels in response to exceptions such as the differentiation of `sqrt(x)` when `x` is zero, and can easily be customized through the use of template files. Exception-reporting levels range from "performance mode," which never reports exceptions, to "verbose mode," which reports every exception that occurs. Intermediate levels of exception reporting each generate more concise summaries of exceptions that have occurred.

Transparent Sparsity Support. Code generated with ADIFOR 2.0 can perform derivative computations by using the SparsLinC (Sparse Linear Combination) library. This sparse flavor of ADIFOR 2.0 allows transparent exploitation of sparsity arising in large sparse Jacobian computations or gradients of functions that have a sparse Hessian [3, 10].

Interprocedural Activity Analysis. ADIFOR 2.0 performs an analysis of the entire program that the user wants to differentiate to determine which real-valued and complex-valued variables depend on the user's specified independent variables and which are used to compute the user's specified dependent variables. The analysis traces the use of floating-point data throughout the program being differentiated. The goal of this analysis is to reduce the cost of computing derivatives by identifying variables whose values are irrelevant to the computation of the requested derivatives.

To support interprocedural activity analysis, ADIFOR 2.0 insists that the user provide the Fortran 77 source code for a *complete* and *consistent* program. To be *complete*, a program must have no missing entry points; that is, the program must link without undefined external references. To be *consistent*, the number of arguments and the types of those arguments must agree between a call site and the called procedure. In our experience, the interprocedural analysis phase of ADIFOR 2.0 processing is the most memory-consuming and time-consuming part of the process.

The derivative code generated by ADIFOR 2.0 provides, as expected from the forward mode of AD [9], the ability to compute directional derivatives. Instead of simply producing code to compute the Jacobian J, ADIFOR 2.0 produces code to compute $J * S$, where the "seed matrix" S is initialized by the user. Thus, if S is the identity, ADIFOR 2.0 computes the full Jacobian, whereas if S is just a vector, ADIFOR 2.0 computes the product of the Jacobian by a vector.

The seed matrix mechanism allows for flexible use of the code generated by ADIFORTWO. For example, it can be employed to compute compressed versions of large sparse Jacobians [1], to chain derivatives generated by programs running on different platforms [6, 15], or to decrease turnaround time for derivative computations through a parallel stripmining approach [7].

The benefit from proper initialization of the seed matrix is substantial, since the cost of derivative computation is more or less proportional to the number p of directional derivatives (equal to the number of columns of S) that are computed in one run. Hence, computing a Jacobian-vector product is much less expensive than computing the Jacobian itself. Typically, but not always, the code generated by ADIFOR 2.0 runs two to four times faster than one-sided divided difference approximations when one computes more than 5–10 derivatives at one time. This advantage comes from coupling the hybrid mode with interprocedural activity analysis.

3 Users' Experiences

As software developers providing software for academic use and commercial evaluation, we have been curious about users' experiences with our software. In early January 1996, we sent out an informal questionnaire to all of the people who had requested ADIFOR 2.0. Here we summarize the responses we received.

Question: Was it sufficient to provide ADIFOR 2.0 executables for only Sun Sparc, IBM RS/6000, and SGI Iris workstations?

Answer: Apparently not. Five people requested the software and then determined that ADIFOR 2.0 was unavailable for the platforms that they had available to them—a 386-class PC or an HP workstation or DEC Alpha Vax workstation. In response, we are developing ports to Windows 95 and Windows NT for the PC and HPUX for HP workstations.

Question: Have people been able to download the software using their Web browser?

Answer: One person acknowledged being unable to download the software over the Web due to limited available memory on the machine on which the Web browser was being run. In several other cases, European requestors were unable to download files from one of the download sites, but were able to retrieve them from the other.

Question: Did people have problems with the terms of the license that we required that they have signed?

Answer: The vast majority of requestors said that getting the license signed and submitted was not a problem. Some said that it did take some time to get someone to sign the license on behalf of their company or university. Unfortunately, we have no way of knowing whether our insistence on receiving a signed license has scared away potential users of the software from even requesting it.

Question: Did people read the manual [5]? Did they try the examples provided in the manual? Was it helpful?

Answer: The majority of the users claimed to have looked primarily at the examples in the manual. Several commented that the examples were very useful, and several commented that they needed more examples. At least one person commented (negatively) that the manual looked like it had been written by a mathematician or a computer scientist.

Question: Is ADIFOR 2.0 performing robustly?

Answer: Yes. Prior to the survey we had received only two reports of errors in the system. In their response to the survey, three users noted had encountered errors in ADIFOR 2.0 that they had been able to work around. No users claimed to have encountered problems that kept them from using ADIFOR 2.0.

Question: Do people understand the concept of the seed matrix?

Answer: Yes, for simple examples having a single array of independent variables and a single array of dependent variables. For more complex examples, users seemed to struggle to determine the appropriate dimensions and initial values for the seed matrix.

Question: Has the availability of ADIFOR 2.0 changed the way that people are doing their work?

Answer: Without doubt! See the following sections.

3.1 Engineering Codes at NASA Langley

Katherine Young and Joanne Walsh of NASA Langley Research Center have been applying ADIFOR 2.0 to typical engineering analysis codes and then evaluating the resulting sensitivity analysis codes for use in design and optimization. They have applied ADIFOR 2.0 to CAMRAD/JA [24] (a comprehensive rotorcraft analysis code), HOVT (a hover analysis code), and WOPWOP (a rotor acoustics code). CAMRAD/JA is the largest code ever processed by ADIFOR (250,000 lines after processing) and includes complex arithmetic, complicated trim logic, and huge amounts of input and output data. Derivatives of horsepower, hub shear, drag, vibratory frequency, thickness, and loading noise were computed with respect to rotor blade planform design variables such as blade twist, taper ratio, and root chord length. The derivatives produced by the preprocessed versions of these codes are consistent with finite difference derivatives.

Larry Green and Perry Newman of NASA Langley and Art Taylor of Old Dominion University also are using ADIFOR 2.0 to compute derivatives of CFD codes [15].

3.2 Neural Networks at DuPont

Aaron Owens, in the Engineering Research Laboratory of DuPont, is using ADIFOR 2.0 in several research projects to demonstrate the feasibility of using stiff ordinary differential equation solvers and gradient descent to solve difficult multivariate real function optimization problems: specifically, given a real function $F(P)$ with parameters P, minimize F w.r.t P by solving the ODE's $dP/dt = -dF/dP$, where P is a long vector, with hundreds or thousands of elements.

The first successful application of ADIFOR 2.0 was with a neural network having a combination of several kinds of squashing functions (not just sigmoidal). It was tedious to compute the gradient analytically to do neural net learning; hence, numerical differentiation had been used. With ADIFOR 2.0, however, researchers were able to accurately compute the required gradient an order of magnitude faster than by using numerical derivatives.

3.3 Maximum Likelihood Optimization at the Harvard School of Public Health

Mario Casella and Donna Spiegelman, in the Departments of Epidemiology and Biostatistics at the Harvard School of Public Health, are using ADIFOR 2.0 to perform maximum likelihood optimization on problems in nutritional epidemiology [22]. Two likelihood functions were considered, one with 17 parameters, the other with 33 parameters. For the 17-parameter case, analytic derivatives had been constructed by hand over a period of two years. Initial results using the derivative code generated by ADIFOR 2.0 exhibited roughly a linear increase in computational complexity for gradients and quadratic increase for Hessians (by applying ADIFOR 2.0 twice), as expected. Gradients and Hessians for the 17-parameter problem were computed in the time required to compute 17.07 and 515.17 original function evaluations, respectively. In comparison, the hand-coded analytic derivatives computed gradients and Hessians for the 17-parameter case in the time required to compute 3.35 and 16.98 original function evaluations.

Disappointed with the time required for ADIFOR 2.0 to compute the derivatives, additional effort was expended to see whether "interface contraction" could be used to reduce the cost of the derivative computations. Interface contraction can be applied whenever the number of variables that are passed as inputs to some subroutine is small compared with the number of independent variables. Computing derivatives of the subroutine with respect to the small number of inputs and then applying the chain rule at the subroutine level should be significantly cheaper than computing derivatives with respect to the larger number of independent variables. For the 17-parameter case, interface contraction was applied to one procedure invocation that took only two inputs. The resulting derivative code computed gradients and Hessians in the time required to compute 4.97 and 55.66 original function evaluations, a significant speedup requiring only a small number of actual source code modifications.

The derivatives computed by the original and modified derivative code agreed with those computed by both finite differences and hand-coded derivatives. Except for rare cases, full double precision was achieved.

Casella states that the ability to construct efficient derivative codes has added great flexibility to his research, since he can easily implement different models to analyze data

```
      SUBROUTINE DGEMV ( TRANS, M, N, ALPHA, A, LDA, X, INCX,
     $                   BETA, Y, INCY )
      ...
            DO 60, J = 1, N
               IF( X( JX ).NE.ZERO )THEN
                  TEMP = ALPHA*X( JX )
                  DO 50, I = 1, M
                     Y( I ) = Y( I ) + TEMP*A( I, J )
   50             CONTINUE
               END IF
               JX = JX + INCX
   60       CONTINUE
     ...
     END
```

FIG. 1. *Branch-problem arising in DGEMV BLAS routine*

and then rely on ADIFOR 2.0 for the needed derivatives. He estimates that he "crunches derivatives" almost weekly.

3.4 Chemical Process Simulation at the University of Texas

Kenneth Teague, Jr., in the Department of Chemical Engineering at the University of Texas at Austin, is using ADIFOR 2.0 to develop a computer simulation of the pressure swing adsorption air separation process. A subproblem is to simulate adsorption of oxygen and nitrogen by a zeolite-packed column. For example, when the column is saturated with oxygen feed and the feed is switched to pure nitrogen, he needs to know the effluent composition as a function of time.

The effluent composition vs. time can be determined by solving the system of nonlinear, coupled partial differential equations that describe conservation of oxygen and nitrogen in the gas and solid phases. Teague discretized the axial dimension of the column in these equations, using the Galerkin finite element technique to obtain a large system of ordinary differential equations, which he then solved using the differential algebraic system solver DASSL. The ODE system was presented to DASSL as $f(y, y', t) = 0$ by computing for DASSL $\delta = f(y, y', t)$, after giving it initial y, y', and t. To integrate this system, DASSL needs $df(y, y', t)/dy + \alpha * df(y, y', t)/dy'$ at each time step. Although DASSL can do this computation by using finite differences, it is preferable to use the sparse flavor of ADIFOR 2.0 and then provide the result to DASSL. The derivatives are computed not only more efficiently, but more accurately, making DASSL's solution more stable. This combined effect has reduced the solution time by orders of magnitude.

Teague encountered an instance of the "branch-problem of AD" [18] in applying ADIFOR 2.0 to compute the derivatives needed by DASSL. The Jacobian of $y = A * x$, for any x, should be A. If one uses the BLAS routine DGEMV to evaluate the term $A * x$ for $x = 0$, the derivative code generated by ADIFOR 2.0 yields a zero derivative. This situation happens, as shown in Figure 1, because DGEMV avoids computation of zero entries in y by the corresponding column vector in A and instead immediately returns a zero vector. The problem was solved by replacing the call to DGEMV with explicit unoptimized `DO` loops in the code that is processed by ADIFOR 2.0, while retaining the call to DGEMV in

the function evaluation code itself to get the maximum efficiency when derivatives are not being computed.

3.5 Real-time Optimization at Dynamic Matrix Control Corporation

Dynamic Matrix Control Corporation (DMCC), located in Houston, Texas, licenses a commercial real-time optimization system for refineries and chemical plants called [DMO], Dynamic Matrix Optimization. The [DMO] system uses a sparse-matrix SQP algorithm and fundamental chemical engineering models of the process to optimize the plant in real time. Typical model sizes are on the order of 150,000 to 250,000 equations with 1.5 to 3.0 million derivatives. The SQP algorithm requires first derivatives from the models, which are computed either by finite differences or analytically using ADIFOR 2.0.

Steve Hendon at DMCC is using ADIFOR 2.0 to generate derivative code using the "compressed" Jacobian computing scheme to capitalize on known sparsity information. SparsLinC has not yet been tried on this problem. Performance-mode exception handling is used exclusively in the code; however, the exception template for `ABS` was modified to return a derivative of 1.0 when the argument was equal to zero.

On average, a 50% improvement in differentiation time has been observed. The derivatives have been verified by comparison with numerical approximations with typical differences in the sixth or seventh decimal place.

The only significant difficulty encountered in processing code for [DMO] is ADIFOR 2.0's insistence on having access to all of the source code for all routines invoked by the routines for which derivatives are needed. More than 800 subroutines make up the optimization program, with only 143 subroutines actually required in the derivative computations. Many of these routines do not actively participate in floating-point computations; they perform input/output and define which equations and formulations are to be used. ADIFOR 2.0 currently takes over an hour to process the 143 subroutines and generate the differentiated code on an RS/6000. In doing so, ADIFOR 2.0 must process approximately 30,000 lines of Fortran 77 code.

In using ADIFOR 2.0, DMCC has encountered only three bugs: two were ADIFOR 2.0 problems, and one was a problem with the provided function code.

Hendon claims that the key advantage to using tools like ADIFOR 2.0 in software development efforts is the reduction in cost and effort required to generate and maintain analytical derivative code. Generating the derivatives from the original source during a pre-compilation step frees developers to concentrate on model development, rather than derivative code development.

4 Lessons Learned

Here are some of the lessons we learned from our survey:

- Users of ADIFOR 2.0 are not computer scientists (or mathematicians) and do not wish to be.
- Few users of the system have had any experience with AD prior to using ADIFOR 2.0.
- Users look at examples, and ignore most of the rest of the manual until absolutely stuck.
- Our users seem hesitant to form a users community. They also seem hesitant to report bugs. (Do people just *expect* bugs in free software?)

- Most users of ADIFOR 2.0 have taken the time to verify the answers computed by the generated derivative code.
- A good number of people who requested the ADIFOR 2.0 software found out about it by "Web surfing."
- Users of AD technology are working on a fairly large variety of hardware platforms.
- The larger the code to be processed, the more likely it is that the person applying ADIFOR 2.0 to the code was not the author of the code.

Acknowledgments

We thank everyone who requested ADIFOR 2.0 and who responded to our questionnaire, especially Katherine Young, Aaron Owens, Mario Casella, Kenneth Teague, Jr., and Steve Hendon. In all cases, comments made by those people who responded represent only their personal views, not the views of their employer.

References

[1] B. Averick, J. Moré, C. Bischof, A. Carle, and A. Griewank, *Computing large sparse Jacobian matrices using automatic differentiation*, SIAM Journal on Scientific Computing, 15 (1994), pp. 285–294.

[2] J.-F. Barthelemy and L. Hall, *Automatic differentiation as a tool in engineering design*, in Proceedings of the 4th AIAA/USAF/NASA/OAI Symp. on Multidisciplinary Analysis and Optimization, AIAA 92-4743, American Institute of Aeronautics and Astronautics, 1992, pp. 424–432.

[3] C. Bischof, A. Bouaricha, P. Khademi, and J. Moré, *Computing gradients in large-scale optimization using automatic differentiation*, Preprint MCS-P488-0195, Mathematics and Computer Science Division, Argonne National Laboratory, 1995.

[4] C. Bischof, A. Carle, P. Khademi, and A. Mauer, *The ADIFOR 2.0 system for the automatic differentiation of Fortran 77 programs.* Preprint MCS-P481-1194, Mathematics and Computer Science Division, Argonne National Laboratory, and Technical Report CRPC-TR94491, Center for Research on Parallel Computation, Rice University, 1994.

[5] C. Bischof, A. Carle, P. Khademi, A. Mauer, and P. Hovland, *ADIFOR 2.0 user's guide*, Technical Memorandum ANL/MCS-TM-192, Mathematics and Computer Science Division, Argonne National Laboratory, and Technical Report CRPC-TR95516-S, Center for Research on Parallel Computation, Rice University, 1994.

[6] C. Bischof, G. Corliss, L. Green, A. Griewank, K. Haigler, and P. Newman, *Automatic differentiation of advanced CFD codes for multidisciplinary design*, Journal on Computing Systems in Engineering, 3 (1992), pp. 625–638.

[7] C. Bischof, L. Green, K. Haigler, and T. Knauff, *Parallel calculation of sensitivity derivatives for aircraft design using automatic differentiation*, in Proceedings of the 5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA 94-4261, American Institute of Aeronautics and Astronautics, 1994, pp. 73–84.

[8] C. Bischof and A. Griewank, *Computational differentiation and multidisciplinary design*, in Inverse Problems and Optimal Design in Industry, H. Engl and J. McLaughlin, eds., Stuttgart, 1994, Teubner Verlag, pp. 187–211.

[9] C. Bischof and P. Hovland, *Using ADIFOR to compute dense and sparse Jacobians*, Technical Memorandum. ANL/MCS-TM-158, Mathematics and Computer Science Division, Argonne National Laboratory, 1991.

[10] C. Bischof, P. Khademi, A. Bouaricha, and A. Carle, *Computation of gradients and Jacobians by transparent exploitation of sparsity in automatic differentiation*, Preprint MCS-P519-0595, Mathematics and Computer Science Division, Argonne National Laboratory, 1995.

[11] C. Bischof, G. Whiffen, C. Shoemaker, A. Carle, and A. Ross, *Application of automatic differentiation to groundwater transport models*, in Computational Methods in Water Resources X, A. Peters, ed., Dordrecht, 1994, Kluwer Academic Publishers, pp. 173–182.

[12] C. H. Bischof, *Automatic differentiation, tangent linear models and pseudo-adjoints*, Preprint MCS-P472-1094, Mathematics and Computer Science Division, Argonne National Laboratory, 1994.

[13] B. W. Brown, F. M. Spears, L. B. Levy, J. Lovato, and K. Russell, *Algorithm LLDRLF: Log-likelihood and some derivatives for Log-F models*, Technical Report, Dept. of Biomathematics, The University of Texas M. D. Anderson Cancer Center, Houston, 1994.

[14] D. W. Byun, R. Dennis, D. Hwang, J. Carlie Coats, and M. T. Odman, *Computational modeling issues in next generation air quality models*, in Proceedings of IMACS'94, Atlanta, Georgia, 1994.

[15] A. Carle, L. Green, C. Bischof, and P. Newman, *Applications of automatic differentiation in CFD*, in Proceedings of the 25th AIAA Fluid Dynamics Conference, AIAA Paper 94-2197, American Institute of Aeronautics and Astronautics, 1994.

[16] S. Chinchalkar, *The application of automatic differentiation to problems in engineering analysis*, Computer Methods in Applied Mechanics and Engineering, 118 (1994), pp. 197–207.

[17] G. F. Corliss, C. Bischof, A. Griewank, S. Wright, and T. Robey, *Automatic differentiation for PDE's – unsaturated flow case study*, in Advances in Computer Methods for Partial Differential Equations – VII, R. Vichnevetski, D. Knight, and G. Richter, eds., New Brunswick, 1992, IMACS, pp. 150–156.

[18] H. Fischer, *Special problems in automatic differentiation*, in Automatic Differentiation of Algorithms: Theory, Implementation, and Application, A. Griewank and G. F. Corliss, eds., SIAM, Philadelphia, Penn., 1991, pp. 43 – 50.

[19] L. Green, P. Newman, and K. Haigler, *Sensitivity derivatives for advanced CFD algorithm and viscous modeling parameters via automatic differentiation*, in Proceedings of the 11th AIAA Computational Fluid Dynamics Conference, AIAA Paper 93-3321, American Institute of Aeronautics and Astronautics, 1993.

[20] U. Häußermann, *Automatische Differentiation zur Rekursiven Bestimmung von Partiellen Ableitungen*. STUD-102, Institut B für Mechanik, Universität Stuttgart, 1993.

[21] M. Heidari and S. Ranjithan, *A hybrid optimization approach to the estimation of distributed parameters in two dimensional confined aquifers under steady state conditions*. Draft manuscript, 1994.

[22] P. Hovland, C. Bischof, D. Spiegelman, and M. Casella, *Efficient derivative codes through automatic differentiation and interface contraction: An application in biostatistics*, Preprint MCS-P491-0195, Mathematics and Computer Science Division, Argonne National Laboratory, 1995.

[23] A. Ibsais and V. Ajjarapu, *The application of automatic differentiation in the continuation power flow*, in Proc. 26th North American Power Symposium, Part I, Manhattan, Kansas, 1994, pp. 329–337.

[24] W. Johnson, *CAMRAD/JA – a comprehensive analytical model of rotorcraft aerodynamics and dynamics - Johnson Aeronautics version*, Technical Report, Johnson Aeronautics, 1988.

[25] V. Korivi, L. Sherman, A. Taylor, G. Hou, L. Green, and P. Newman, *First- and second-order aerodynamic sensitivity derivatives via automatic differentiation with incremental iterative methods*, in Proceedings of the 5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA 94-4262, American Institute of Aeronautics and Astronautics, 1994, pp. 87–120.

[26] V. Korivi, A. Taylor, and P. Newman, *Aerodynamic optimization studies using a 3-D supersonic Euler code with efficient calculation of sensitivity derivatives*, in Proceedings of the 5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA 94-4270, American Institute of Aeronautics and Astronautics, 1994, pp. 170–194.

[27] D. Muir, *Description of covariance data in ENDF-6 format*, in Proc. on Nuclear Data Evaluation Methodology, C. L. Dunford, ed., World Scientific, 1993.

[28] S. K. Park and K. Droegemeier, *Effect of a microphysical parameterization on the evolution of*

linear perturbations in a convective cloud model, in Preprints, Conference on Cloud Physics, January 1995, Dallas, Texas, American Meteorological Society, 1995.

[29] S. K. Park, K. Droegemeier, C. Bischof, and T. Knauff, *Sensitivity analysis of numerically-simulated convective storms using direct and adjoint methods*, in Preprints, 10th Conference on Numerical Weather Prediction, Portland, Oregon, American Meteorological Society, 1994, pp. 457–459.

[30] M. Rosemblun, *Automatic differentiation: Overview and application to systems of parameterized nonlinear equations*, Technical Report CRPC-TR92267, Center for Research on Parallel Computation, Rice University, 1992.

[31] E. R. Unger and L. E. Hall, *The use of automatic differentiation in an aircraft design problem*, in Proceedings of the 5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA 94-4260, American Institute of Aeronautics and Astronautics, 1994, pp. 64–73.

[32] G. Whiffen, C. Shoemaker, C. Bischof, A. Ross, and A. Carle, *Application of automatic differentiation to groundwater transport codes*, Preprint MCS-P441-0594, Mathematics and Computer Science Division, Argonne National Laboratory, 1994.