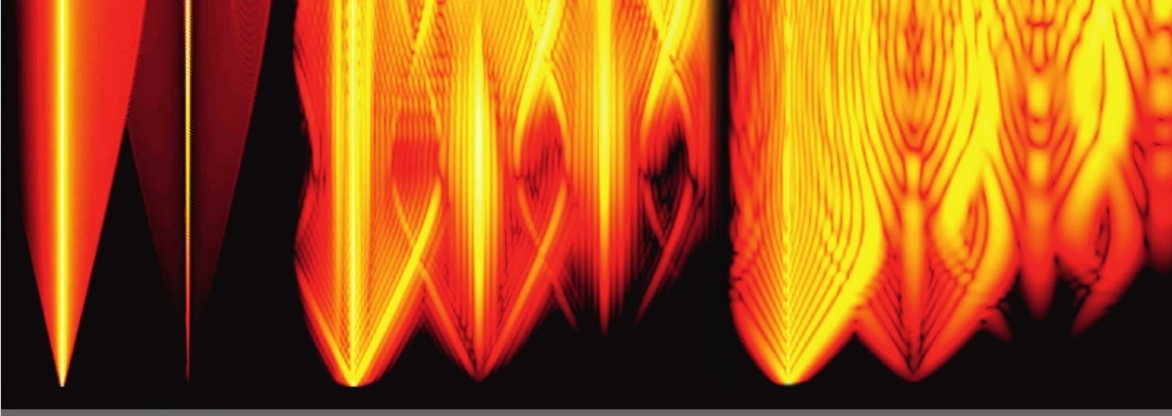

LECTURE NOTES IN COMPUTATIONAL SCIENCE AND ENGINEERING 87

Shaun Forth · Paul Hovland · Eric Phipps Jean Utke · Andrea Walther *Editors*

Lecture Notes
in Computational Science
and Engineering

87

Editors:

For further volumes:
http://www.springer.com/series/3527

Shaun Forth • Paul Hovland • Eric Phipps
Jean Utke • Andrea Walther
Editors

Recent Advances in Algorithmic Differentiation

Springer

Editors
Shaun Forth
Applied Mathematics
and Scientific Computing
Cranfield University
Shrivenham
Swindon
United Kingdom

Paul Hovland
Mathematics and Computer Science
Division
Argonne National Laboratory
Argonne
Illinois
USA

Eric Phipps
Sandia National Laboratory
Albuquerque
New Mexico
USA

Jean Utke
Mathematics and Computer Science
Division
Argonne National Laboratory
Argonne
Illinois
USA

Andrea Walther
Department of Mathematics
University of Paderborn
Paderborn
Germany

ISSN 1439-7358
ISBN 978-3-642-30022-6 ISBN 978-3-642-30023-3 (eBook)
DOI 10.1007/978-3-642-30023-3
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012942187

Mathematics Subject Classification (2010): 65D25, 90C30, 90C31, 90C56, 65F50, 68N20, 41A58, 65Y20

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The Sixth International Conference on Automatic Differentiation (AD2012) held July 23–27, 2012, in Fort Collins, Colorado (USA), continued this quadrennial conference series. While the fundamental idea of differentiating numerical programs is easy to explain, the practical implementation of this idea for many nontrivial numerical computations is not. Our community has long been aware of the discrepancy between the aspiration of an *automatic* process suggested by the name automatic differentiation and the reality of its practical use, which often requires substantial effort from the user. New algorithms and methods implemented in differentiation tools improve their usability and reduce the need for user intervention. On the other hand, the demands to compute derivatives for numerical models on parallel hardware, using a wide variety of libraries and having components implemented in different programming languages, pose new challenges, particularly for the efficiency of the derivative computation. These challenges, as well as new applications, have been driving research for the past four years and will continue to do so. Despite retaining automatic differentiation in the conference name, the editors purposely switched to *algorithmic differentiation* (AD) in the proceedings title. Thus, the conference proceedings follow somewhat belatedly the more appropriate naming chosen by Andreas Griewank for the first edition of his seminal monograph covering our subject area. This name better reflects the reality of AD usage and the research results presented in the papers collected here.

The 31 contributed papers cover the application of AD to many areas of science and engineering as well as aspects of AD theory and its implementation in tools. For all papers the referees, selected from the program committee and the wider AD community, as well as the editors have emphasized accessibility of the presented ideas also to non-AD experts.

In the AD tools arena new implementations are introduced covering, for example, Java and graphical modeling environments, or join the set of existing tools for Fortran. New developments in AD algorithms target: efficient derivatives for matrix-operation, detection and exploitation of sparsity, partial separability, the treatment of nonsmooth functions, and other high-level mathematical aspects of the numerical computations to be differentiated.

Applications stem from the Earth sciences, nuclear engineering, fluid dynamics, and chemistry, to name just a few. In many cases the applications in a given area of science or engineering share characteristics that require specific approaches to enable AD capabilities or provide an opportunity for efficiency gains in the derivative computation. The description of these characteristics and of the techniques for successfully using AD should make the proceedings a valuable source of information for users of AD tools.

The image on the book cover shows the high-harmonic emission spectrum of a semiconductor quantum dot for different excitation conditions. To favor specific frequencies one has to find an appropriate input pulse within a large parameter space. This was accomplished by combining a gradient-based optimization algorithm with AD. The data plots were provided by Matthias Reichelt.

Algorithmic differentiation draws on many aspects of applied mathematics and computer science and ultimately is useful only when users in the science and engineering communities become aware of its capabilities. Furthering collaborations outside the core AD community, the AD2012 program committee invited leading experts from diverse disciplines as keynote speakers. We are grateful to Lorenz Biegler (Carnegie Mellon University, USA), Luca Capriotti (Credit Suisse, USA), Don Estep (Colorado State University, USA), Andreas Griewank (Humboldt University, Germany), Mary Hall (University of Utah, USA), Barbara Kaltenbacher (University of Klagenfurt, Austria), Markus Püschel (ETH Zurich, Switzerland), and Bert Speelpenning (MathPartners, USA) for accepting the invitations.

We want to thank SIAM and the NNSA and ASCR programs of the US Department of Energy for their financial support of AD2012.

Albuquerque, Chicago
Paderborn, Shrivenham
April 2012

Shaun Forth
Paul Hovland
Eric Phipps
Jean Utke
Andrea Walther

Program Committee AD2012

Brad Bell, University of Washington (USA)
Martin Berz, Michigan State University (USA)
Christian Bischof, TU Darmstadt (Germany)
Martin Bücker, RWTH Aachen (Germany)
Bruce Christianson, University of Hertfordshire (UK)
David Gay, AMPL Optimization Inc. (USA)
Andreas Griewank, Humboldt University Berlin (Germany)
Laurent Hascoët, INRIA (France)
Patrick Heimbach, Massachusetts Institute of Technology (USA)
Koichi Kubota, Chuo University (Japan)
Kyoko Makino, Michigan State University (USA)
Jens-Dominik Müller, Queen Mary University of London (UK)
Uwe Naumann, RWTH Aachen (Germany)
Boyana Norris, Argonne National Laboratory (USA)
Trond Steihaug, University of Bergen (Norway)

Contents

Contributors

Hany S. Abdel-Khalik Department of Nuclear Engineering, North Carolina State University, Raleigh, NC, USA, abdelkhalik@ncsu.edu

Johan Åkesson Department of Automatic Control, Faculty of Engineering, Lund University, Lund, Sweden, johan.akesson@control.lth.se

Juan J. Alonso Department of Aeronautics and Astronautics, Stanford University, Stanford, CA, USA, jjalonso@stanford.edu

Joel Andersson Electrical Engineering Department (ESAT) and Optimization in Engineering Center (OPTEC), K.U. Leuven, Heverlee, Belgium, joel.andersson@esat.kuleuven.be

Mihai Anitescu Mathematics and Computer Science, Argonne National Laboratory, Argonne, IL, USA, anitescu@mcs.anl.gov

Frédéric Baret INRA, Avignon, France, baret@avignon.inra.fr

Paul I. Barton Process Systems Engineering Laboratory, Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA, pib@mit.edu

Markus Beckers STCE, RWTH Aachen University, Aachen, Germany, beckers@stce.rwth-aachen.de

Christian H. Bischof Scientific Computing Group, TU Darmstadt, Darmstadt, Germany, christian.bischof@sc.tu-darmstadt.de

Torsten Bosse Humboldt-Universität zu Berlin, Berlin, Germany, bosse@math.hu-berlin.de

H. Martin Bücker Institute for Scientific Computing, RWTH Aachen University, Aachen, Germany, buecker@sc.rwth-aachen.de

David C. Carothers James Madison University, Harrisonburg, USA, carothdc@jmu.edu

Jieqiu Chen Mathematics and Computer Science, Argonne National Laboratory, Argonne, IL, USA, jieqchen@mcs.anl.gov

Bruce Christianson School of Computer Science, University of Hertfordshire, Hatfield, UK, b.christianson@herts.ac.uk

Thomas F. Coleman Department of Combinatorics and Optimization, University of Waterloo, Ontario, Canada, tfcoleman@uwaterloo.ca

Heather Cole-Mullen Argonne National Laboratory, The University of Chicago, Chicago, IL, USA, hlcm@mcs.anl.gov

Benoit Delinchant Grenoble Electrical Engineering Laboratory, Saint-Martin d'Hères, France, benoit.delinchant@grenoble-inp.fr

Moritz Diehl Electrical Engineering Department (ESAT) and Optimization in Engineering Center (OPTEC), K.U. Leuven, Heverlee, Belgium, moritz.diehl@esat.kuleuven.be

Beth Drewniak Environmental Science, Argonne National Laboratory, Argonne, IL, USA, bbye@anl.gov

Jeffrey A. Fike Department of Aeronautics and Astronautics, Stanford University, Stanford, CA, USA, jfike@stanford.edu

Shaun A. Forth Applied Mathematics and Scientific Computing, Cranfield University, Swindon, UK, S.A.Forth@cranfield.ac.uk

Nicolas R. Gauger Computational Mathematics Group, CCES, RWTH Aachen University, Aachen, Germany, gauger@mathcces.rwth-aachen.de

Assefaw Gebremedhin Department of Computer Science, Purdue University, West Lafayette, IN, USA, agebreme@purdue.edu

Ralf Giering FastOpt GmbH, Lerchenstrasse 28a, 22767 Hamburg, Germany, Ralf.Giering@FastOpt.com

Vaibhava Goel IBM, TJ Watson Research Center, Yorktown Heights, NY, USA, vgoel@us.ibm.com

Andreas Griewank Humboldt-Universität zu Berlin, Berlin, Germany, griewank@math.hu-berlin.de

Abhishek Kr. Gupta Department of Electrical Engineering, IIT Kanpur, Kanpur, India, g.kr.abhishek@gmail.com

Laurent Hascoët INRIA, Sophia-Antipolis, France, Laurent.Hascoet@inria.fr

Paul Hovland Mathematics and Computer Science, Argonne National Laboratory, Argonne, IL, USA, hovland@mcs.anl.gov

John Junkins Aerospace Engineering, Texas A&M University, College Station, TX, USA, junkins@tamu.edu

Kamil A. Khan Process Systems Engineering Laboratory, Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA, kamil@mit.edu

Kshitij Kulshreshtha Institut für Mathematik, Universität Paderborn, Paderborn, Germany, kshitij@math.upb.de

Claire Lauvernet Irstea, Lyon, France, claire.lauvernet@irstea.fr

François-Xavier Le Dimet Université de Grenoble, Grenoble, France, Francois-Xavier.Le-Dimet@imag.fr

Benjamin Letschert Universität Paderborn, Institut für Mathematik, Paderborn, Germany, Benny.Letschert@web.de

Johannes Lotz STCE, RWTH Aachen University, Aachen, Germany, lotz@stce.rwth-aachen.de

Stephen K. Lucas James Madison University, Harrisonburg, VA, USA, lucassk@jmu.edu

Andrew Lyons Dartmouth College, Hanover, NH, USA, lyonsam@gmail.com

Manoranjan Majji Mechanical and Aerospace Engineering, University at Buffalo, Buffalo, NY, USA, majjineo@gmail.com

Azamat Mametjanov Mathematics and Computer Science, Argonne National Laboratory, Argonne, IL, USA, azamat.mametjanov@gmail.com

Jan Marburger Fraunhofer-Institut für Techno- und Wirtschaftsmathematik, Kaiserslautern, Germany, jan.marburger@itwm.fhg.de

Viktor Mosenkis STCE, RWTH Aachen University, Aachen, Germany, mosenkis@stce.rwth-aachen.de

Todd Munson Mathematics and Computer Science, Argonne National Laboratory, Argonne, IL, USA, tmunson@mcs.anl.gov

Sri Hari Krishna Narayanan Mathematics and Computer Science, Argonne National Laboratory, Argonne, IL, USA, snarayan@mcs.anl.gov

Uwe Naumann STCE, RWTH Aachen University, Aachen, Germany, naumann@stce.rwth-aachen.de

Marco Nehmeier Institute of Computer Science, University of Würzburg, Würzburg, Germany, nehmeier@informatik.uni-wuerzburg.de

Anil Nemili Computational Mathematics Group, CCES, RWTH Aachen University, Aachen, Germany, nemili@mathcces.rwth-aachen.de

Duc Nguyen Department of Computer Science, Purdue University, West Lafayette, IN, USA, nguyend@purdue.edu

Boyana Norris Mathematics and Computer Science, Argonne National Laboratory, Argonne, IL, USA, norris@mcs.anl.gov

Peder A. Olsen IBM, TJ Watson Research Center, Yorktown Heights, NY, USA, pederao@us.ibm.com

Emre Özkaya Computational Mathematics Group, CCES, RWTH Aachen University, Aachen, Germany, ozkaya@mathcces.rwth-aachen.de

G. Edgar Parker James Madison University, Harrisonburg, VA, USA, parkerge@jmu.edu

Valérie Pascual INRIA, Sophia-Antipolis, Sophia-Antipolis, France, Valerie.Pascual@inria.fr

Roger Pawlowski Sandia National Laboratories, Multiphysics Simulation Technologies Department, Albuquerque, NM, USA, rppawlo@sandia.gov

Barak A. Pearlmutter Department of Computer Science and Hamilton Institute, National University of Ireland, Maynooth, Ireland, barak@cs.nuim.ie

Phuong Pham-Quang CEDRAT S.A., Meylan Cedex, France, phuong.phamquang@cedrat.com

Eric Phipps Sandia National Laboratories, Optimization and Uncertainty Quantification Department, Albuquerque, NM, USA, etphipp@sandia.gov

Alex Pothen Department of Computer Science, Purdue University, West Lafayette, IN, USA, apothen@purdue.edu

Alexey Radul Hamilton Institute, National University of Ireland, Maynooth, Ireland, alexey.radul@nuim.ie

James A. Reed Department of Nuclear Engineering, North Carolina State University, Raleigh, NC, USA, jareed3@ncsu.edu

Steven J. Rennie IBM, TJ Watson Research Center, Yorktown Heights, NY, USA, sjrennie@us.ibm.com

Daniel R. Reynolds Mathematics, Southern Methodist University, Dallas, TX, USA, reynolds@smu.edu

Joseph D. Rudmin James Madison University, Harrisonburg, VA, USA, rudminjd@jmu.edu

Ravi Samtaney Mechanical Engineering, Division of Physical Scienceand Engineering, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia, Ravi.Samtaney@kaust.edu.sa

Jeffrey Mark Siskind Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, qobi@purdue.edu

James S. Sochacki James Madison University, Harrisonburg, VA, USA, sochacjs@jmu.edu

Hamdi A. Tchelepi Department of Energy Resources Engineering, Stanford University, Stanford, CA, USA, tchelepi@stanford.edu

Roger J. Thelwell James Madison University, Harrisonburg, VA, USA, thelwerj@jmu.edu

Anthony Tongen James Madison University, Harrisonburg, VA, USA, tongenal@jmu.edu

James Turner Aerospace Engineering, Texas A&M University, College Station, TX, USA, turner@aero.tamu.edu

Jörn Ungermann Institute of Energy and Climate Research – Stratosphere (IEK-7), Research Center Jülich GmbH, Jülich, Germany, j.ungermann@fz-juelich.de

Jean Utke Argonne National Laboratory, The University of Chicago, Chicago, IL, USA, utke@mcs.anl.gov

Michael Voßbeck FastOpt GmbH, Lerchenstrasse 28a, 22767 Hamburg, Germany, Michael.Vossbeck@FastOpt.com

Andrea Walther Institut für Mathematik, Universität Paderborn, Paderborn, Germany, andrea.walther@uni-paderborn.de

Paul G. Warne James Madison University, Harrisonburg, VA, USA, warnepg@jmu.edu

Johannes Willkomm Scientific Computing Group, TU Darmstadt, Darmstadt, Germany, johannes.willkomm@sc.tu-darmstadt.de

Xin Xiong Department of Combinatorics and Optimization, University of Waterloo, Ontario, Canada, x6xiong@uwaterloo.ca

Wei Xu Department of Mathematics, Tongji University, Shanghai, China, wdxu@tongji.edu.cn

Ahmad Bani Younes Aerospace Engineering, Texas A&M University, College Station, TX, USA, olalahmad@gmail.com

Rami M. Younis Department of Energy Resources Engineering, Stanford University, Stanford, CA, USA, ryounis@stanford.edu

Xiaoyan Zeng Mathematics and Computer Science, Argonne National Laboratory, Argonne, IL, USA, zeng@mcs.anl.gov

A Leibniz Notation for Automatic Differentiation

Bruce Christianson

Abstract Notwithstanding the superiority of the Leibniz notation for differential calculus, the dot-and-bar notation predominantly used by the Automatic Differentiation community is resolutely Newtonian. In this paper we extend the Leibniz notation to include the reverse (or adjoint) mode of Automatic Differentiation, and use it to demonstrate the stepwise numerical equivalence of the three approaches using the reverse mode to obtain second order derivatives, namely forward-over-reverse, reverse-over-forward, and reverse-over-reverse.

Keywords Leibniz • Newton • Notation • Differentials • Second-order • Reverse mode

1 Historical Background

Who first discovered differentiation?[1] Popular European[2] contenders include Isaac Barrow, the first Lucasian Professor of Mathematics at Cambridge [5]; Isaac Newton, his immediate successor in that chair [21]; and Godfrey Leibniz, a librarian employed by the Duke of Brunswick [19]. The matter of priority was settled in Newton's favour by a commission appointed by the Royal Society. Since the report

[1] Archimedes' construction for the volume of a sphere probably entitles him to be considered the first to discover integral calculus.

[2] Sharaf al-Din al-Tusi already knew the derivative of a cubic in 1209 [1], but did not extend this result to more general functions.

B. Christianson (✉)
School of Computer Science, University of Hertfordshire, College Lane, Hatfield, England Europe
e-mail: b.christianson@herts.ac.uk

S. Forth et al. (eds.), *Recent Advances in Algorithmic Differentiation*, Lecture Notes in Computational Science and Engineering 87, DOI 10.1007/978-3-642-30023-3_1,

of the commission [2] was written by none other than Isaac Newton himself[3] we may be assured of its competence as well as its impartiality. Cambridge University thenceforth used Newton's notation exclusively, in order to make clear where its loyalties lay.

However, if instead we ask, who first discovered *automatic* differentiation, then Leibniz has the best claim. In contrast with Newton's geometric and dynamical interpretation, Leibniz clearly envisaged applying the rules of differentiation to the numerical values which the coefficients represented, ideally by a mechanical means, as the following excerpts [18, 19] respectively show:

> Knowing thus the *Algorithm* (as I may say) of this calculus, which I call *differential* calculus, all other differential equations can be solved by a common method. . . . For any other quantity (not itself a term, but contributing to the formation of the term) we use its differential quantity to form the differential quantity of the term itself, not by simple substitution, but according to the prescribed Algorithm. The methods published before have no such transition.[4]

> When, several years ago, I saw for the first time an instrument which, when carried, automatically records the number of steps taken by a pedestrian, it occurred to me at once that the entire arithmetic could be subjected to a similar kind of machinery . . .

Although Leibniz did devise and build a prototype for a machine to perform some of the calculations involved in automatic differentiation [18], the dream of a mechanical device of sufficient complexity to perform the entire sequence automatically had to wait until 1837, when Charles Babbage completed the design of his programmable analytical engine [20]. Babbage, who was eventually to succeed to Newton's chair, had while still an undergraduate been a moving force behind the group of young turks[5] who forced the University of Cambridge to change from the Newton to the Leibniz notation for differentiation. Babbage described this as rescuing the University from its dot-age [3].

There is no doubt that by the time of Babbage the use of Newton's notation was very badly hindering the advance of British analysis,[6] so it is ironic to reflect that we in the automatic differentiation community continue to use the Newton notation almost exclusively, for example by using a dot to denote the second field of an active variable.

---

[3] Although this fact did not become public knowledge until 1761, nearly 50 years later.

[4] The word Algorithm derives from the eponymous eighth century mathematician Al-Khwarizmi, known in Latin as Algoritmi. Prior to Leibniz, the term referred exclusively to mechanical arithmetical procedures, such as the process for extraction of square roots, applied (by a human) to numerical values rather than symbolic expressions. The italics are in the Latin original: "Ex cognito hoc velut *Algorithmo*, ut ita dicam, calculi hujus, quem voco *differentialem*."

[5] The Analytical Society was founded by Babbage and some of his friends in 1812. So successful was their program of reform that 11 of the 16 original members subsequently became professors at Cambridge.

[6] Rouse Ball writes [4] "It would seem that the chief obstacle to the adoption of analytical methods and the notation of the differential calculus arose from the professorial body and the senior members of the senate, who regarded any attempt at innovation as a sin against the memory of Newton."

2 The Leibniz Notation

Suppose that we have independent variables w, x and dependent variables y, z given by the system

$$y = f(w, x) \qquad z = g(w, x)$$

2.1 The Forward Mode

In Newton notation we would write the forward derivatives as

$$\dot{y} = f'_w \dot{w} + f'_x \dot{x} \qquad \dot{z} = g'_w \dot{w} + g'_x \dot{x}$$

It is quite straightforward to turn this into a Leibniz notation by regarding the second field of an active variable as a differential, and writing dx, dy etc in place of $\dot{x}, \dot{y}$, etc.

In Leibniz notation the forward derivatives become[7]

$$dy = \frac{\partial f}{\partial w} dw + \frac{\partial f}{\partial x} dx \qquad dz = \frac{\partial g}{\partial w} dw + \frac{\partial g}{\partial x} dx$$

where dw, dx are independent and dy, dz are dependent differential variables.[8]

2.2 The Reverse Mode

For the reverse mode of automatic differentiation, the backward derivatives are written in a Newton style notation as

$$\bar{w} = \bar{y} f'_w + \bar{z} g'_w \qquad \bar{x} = \bar{y} f'_x + \bar{z} g'_x$$

This can be turned into a Leibniz form in a similar way to the forward case. We introduce a new notation, writing by, bz in place of the independent barred variables $\bar{y}, \bar{z}$, and bw, bx in place of the dependent barred variables $\bar{w}, \bar{x}$.

---

[7] Since $y \equiv f(w, x)$ we allow ourselves to write $\frac{\partial f}{\partial x}$ interchangeably with $\frac{\partial y}{\partial x}$.

[8] Actually the tradition of treating differentials as independent variables in their own right was begun by d'Alembert as a response to Berkeley's criticisms of the infinitesimal approach [6], but significantly he made no changes to Leibniz's original notation for them. Leibniz's formulation allows for the possibility of non-negligible differential values, referring [19] to "the fact, until now not sufficiently explored, that dx, dy, dv, dw, dz can be taken *proportional* [my italics] to the momentary differences, that is, increments or decrements, of the corresponding x, y, v, w, z", and Leibniz is careful to write $d(xv) = xdv + vdx$, without the term $dxdv$.

$$bw = by\frac{\partial f}{\partial w} + bz\frac{\partial g}{\partial x} \qquad\qquad bx = by\frac{\partial f}{\partial w} + bz\frac{\partial g}{\partial x}$$

We refer to quantities such as bx as *barientials*. Note that the bariential of a dependent variable is independent, and vice versa. Differentials and barientials will collectively be referred to as *varientials*.

The barientials depend on all the dependent underlying variables so, as always with the reverse mode, the full set of equations must be explicitly given before the barientials can be calculated.

2.3 Forward over Forward

Repeated differentiation in the forward mode (the so-called forward-over-forward approach) produces the Newton equation

$$\ddot{y} = f''_{ww}\dot{w}\dot{w} + 2f''_{wx}\dot{w}\dot{x} + f''_{xx}\dot{x}\dot{x} + f'_w\ddot{w} + f'_x\ddot{x}$$

and similarly for $\ddot{z}$. This has the familiar[9] Leibniz equivalent

$$d^2y = \frac{\partial^2 f}{\partial w^2}dw^2 + 2\frac{\partial^2 f}{\partial w \partial x}dwdx + \frac{\partial^2 f}{\partial x^2}dx^2 + \frac{\partial f}{\partial w}d^2w + \frac{\partial f}{\partial x}d^2x$$

and similarly for d^2z.

2.4 Forward over Reverse

Now consider what happens when we apply forward mode differentiation to the backward derivative equations (the so-called forward-over-reverse approach). Here are the results in Newton notation

$$\dot{\bar{w}} = \dot{\bar{y}}f'_w + \bar{y}\,f''_{ww}\dot{w} + \bar{y}\,f''_{wx}\dot{x} + \dot{\bar{z}}g'_w + \bar{z}g''_{ww}\dot{w} + \bar{z}g''_{wx}\dot{x}$$

and here is the Leibniz equivalent

[9]The familiarity comes in part from the fact that this is the very equation of which Hademard said [15] "que signifie ou que représente l'égalité? A mon avis, rien du tout." ["What is meant, or represented, by this equality? In my opinion, nothing at all."] It is good that the automatic differentiation community is now in a position to give Hadamard a clear answer: (y, dy, d^2y) is the content of an active variable.

$$dbw = dby\frac{\partial f}{\partial w} + by\frac{\partial^2 f}{\partial w^2}dw + by\frac{\partial^2 f}{\partial w \partial x}dx + dbz\frac{\partial g}{\partial w} + bz\frac{\partial^2 g}{\partial w^2}dw + bz\frac{\partial^2 f}{\partial w \partial x}dx$$

with similar equations for $\dot{\bar{x}}$ and dbx respectively.

What happens when we repeatedly apply automatic differentiation in other combinations?

3 Second Order Approaches Involving Reverse Mode

For simplicity, in this section we shall consider the case[10] of a single independent variable x and a single dependent variable $y = f(x)$.

3.1 Forward over Reverse

Here are the results in Newton notation for forward-over-reverse in the single variable case. The reverse pass gives

$$y = f(x) \qquad \bar{x} = \bar{y}f'$$

and then the forward pass, with independent variables x and $\bar{y}$, gives

$$\dot{y} = f'\dot{x} \qquad \dot{\bar{x}} = \dot{\bar{y}}f' + \bar{y}f''\dot{x}$$

The Leibniz equivalents are

$$y = f(x) \qquad bx = by\frac{\partial f}{\partial x}$$

and

$$dy = \frac{\partial f}{\partial x}dx \qquad dbx = dby\frac{\partial f}{\partial x} + by\frac{\partial^2 f}{\partial x^2}dx$$

[10]The variables x and y may be vectors: in this case the corresponding differential dx and bariential by are respectively a column vector with components dx^j and a row vector with components by_i; f' is the matrix $J^i_j = \partial_j f^i = \partial f^i/\partial x^j$, and f'' is the mixed third order tensor $K^i_{jk} = \partial^2_{jk} f^i = \partial^2 f^i/\partial x^j \partial x^k$.

3.2 Reverse over Forward

Next, the corresponding results for reverse-over-forward. First the forward pass in Newton notation

$$y = f(x) \qquad \dot{y} = f'\dot{x}$$

then the reverse pass, applying the rules already given, and treating both y and $\dot{y}$ as dependent variables. We use a long bar to denote ADOL-C style reverse mode differentiation [13], starting from $\dot{y}$ and y

$$\overline{x} = \overline{y}\, f' + \overline{\dot{y}}\, f''\dot{x} \qquad \overline{\dot{x}} = \overline{\dot{y}}\, f'$$

In Leibniz notation the forward pass gives

$$y = f(x) \qquad dy = \frac{\partial f}{\partial x} dx$$

and for the reverse pass we treat y and dy as the dependent variables. We denote the bariential equivalent of the long bar by the letter p for the moment, although we shall soon see that this notation can be simplified. This gives

$$px = py\frac{\partial f}{\partial x} + pdy\frac{\partial^2 f}{\partial x^2}dx \qquad pdx = pdy\frac{\partial f}{\partial x}$$

3.3 Reverse over Reverse

Finally we consider reverse over reverse. The first reverse pass gives

$$y = f(x) \qquad \bar{x} = \bar{y}\, f'$$

the dependent variables are y and $\bar{x}$. We denote the adjoint variables on the second reverse pass by a long bar

$$\overline{x} = \overline{y}\, f' + \bar{y}\, f''\overline{\bar{x}} \qquad \overline{\bar{y}} = f'\overline{\bar{x}}$$

and we shall see shortly that the use made here of the long bar is consistent with that of the previous subsection. In Leibniz notation, the first reverse pass corresponds to

$$y = f(x) \qquad bx = by\frac{\partial f}{\partial x}$$

with the dependent variables being y and bx. Denoting the barientials for the second reverse pass by the prefix p, we have

$$px = py\frac{\partial f}{\partial x} + by\frac{\partial^2 f}{\partial x^2}pbx \qquad\qquad pby = \frac{\partial f}{\partial x}pbx$$

In general we write differentials on the right and barientials on the left, but pbx is a bariential of a bariential, and so appears on the right.[11]

4 The Equivalence Theorem

By collating the equations from the three previous subsections, we can immediately see that all three of the second-order approaches involving reverse differentiation produce structurally equivalent sets of equations, in which certain pairs of quantities correspond. In particular, where v is any dependent or independent variable,

$$\overline{v} = \dot{\bar{v}} \qquad\qquad \overline{\dot{v}} = \bar{v} \qquad\qquad \overline{\bar{v}} = \dot{v}$$

or, in Leibniz notation

$$pv = dbv \qquad\qquad pdv = bv \qquad\qquad pbv = dv$$

allowing the use of p-barientials to be eliminated.

However, we can say more than this. Not only are the identities given above true for dependent and independent varientials,[12] the correspondences also hold for the varientials corresponding to all the intermediate variables in the underlying computation. Indeed, the three second-order derivative computations themselves are structurally identical.

This can be seen by defining the intermediate variables v_i in the usual way [14] by the set of equations

$$v_i = \phi_i(v_{j:j\prec i})$$

and then simulating the action of the automatic differentiation algorithm, by using the rules in the preceding subsections to successively eliminate the varientials corresponding to the intermediate variables, in the order appropriate to the algorithm being used.

In all three cases, we end up computing the varientials of each intermediate variable with exactly the same arithmetical steps

$$pbv_i = dv_i = \sum_{j:j\prec i}\frac{\partial \phi_i}{\partial v_j}dv_j \qquad\qquad pdv_i = bv_i = \sum_{k:i\prec k} bv_k\frac{\partial \phi_k}{\partial v_i}$$

[11] If x is a vector then pbx is a column vector.

[12] Recall that this term includes all combinations of differentials and barientials.

and

$$pv_i = dbv_i = \sum_{k:i \prec k} \left\{ dbv_k \frac{\partial \phi_k}{\partial v_i} + bv_k \sum_{j:j \prec k} \frac{\partial^2 \phi_k}{\partial v_i \partial v_j} dv_j \right\}$$

We therefore have established the following

Theorem 1. *The three algorithms forward-over reverse, reverse-over-forward, and reverse-over-reverse are all numerically stepwise identical, in the sense that they not only produce the same numerical output values, but at every intermediate stage perform exactly the same floating point calculations on the same intermediate variable values.*

Although the precise order in which these calculations are performed may depend on which of the three approaches is chosen, each of the three algorithms performs exactly the same floating point arithmetic. Strictly speaking, this statement assumes that an accurate inner product is available as an elemental operation to perform accumulations, such as those given above for dv_i, bv_i, dbv_i, in an order-independent way.

A final caveat is that the statement of equivalence applies only to the floating point operations themselves, and not to the load and store operations which surround them, since a re-ordering of the arithmetic operations may change the contents of the register set and cache.

Historically, all three of the second-order methods exploiting reverse were implemented at around the same time in 1989 [11]: reverse-over-reverse in PADRE2 by Iri and Kubota [16, 17]; reverse-over-forward in ADOL-C by Griewank and his collaborators [12, 13]; and forward-over-reverse by Dixon and Christianson in an Ada package [7, 10]. The stepwise equivalence of forward-over-reverse with reverse-over-reverse was noted in [9] and that of forward-over-reverse with reverse-over-forward in [8].

The stepwise equivalence of the three second order approaches involving the reverse mode nicely illustrates the new Leibniz notation advanced in this paper, but also deserves to be more widely known than is currently the case.

References

1. Al-Tusi, S.A.D.: Treatise on Equations. Manuscript, Baghdad (1209)
2. Anonymous: An account of the book entitled commercium epistolicum collinii et aliorum, de analysi promota; published by order of the Royal Society, in relation to the dispute between Mr. Leibnitz [sic] and Dr. Keill, about the right of invention of the method of fluxions, by some called the differential method. Philosophical Transaction of the Royal Society of London **342**, 173–224 (January and February 1714/5)
3. Babbage, C.: Passages from the Life of a Philosopher. London (1864)
4. Ball, W.W.R.: A History of the Study of Mathematics at Cambridge. Cambridge (1889)

5. Barrow, I.: Lectiones Opticae et Geometricae. London (1669)
6. Berkeley, G.: The Analyst; or, A Discourse Addressed to an Infidel Mathematician, Wherein it is examined whether the Object, Principles, and Inferences of the modern Analysis are more distinctly conceived, or more evidently deduced, than Religious Mysteries and Points of Faith. London (1734)
7. Christianson, B.: Automatic Hessians by reverse accumulation. Technical Report NOC TR228, Numerical Optimisation Centre, Hatfield Polytechnic, Hatfield, United Kingdon (1990)
8. Christianson, B.: Reverse accumulation and accurate rounding error estimates for Taylor series coefficients. Optimization Methods and Software **1**(1), 81–94 (1991). Also appeared as Tech. Report No. NOC TR239, The Numerical Optimisation Centre, University of Hertfordshire, U.K., July 1991
9. Christianson, B.: Automatic Hessians by reverse accumulation. IMA Journal of Numerical Analysis **12**(2), 135–150 (1992)
10. Dixon, L.C.W.: Use of automatic differentiation for calculating Hessians and Newton steps. In: Griewank and Corliss [11], pp. 114–125
11. Griewank, A., Corliss, G.F. (eds.): Automatic Differentiation of Algorithms: Theory, Implementation, and Application. SIAM, Philadelphia, PA (1991)
12. Griewank, A., Juedes, D., Srinivasan, J., Tyner, C.: ADOL-C: A package for the automatic differentiation of algorithms written in C/C++. Preprint MCS-P180-1190, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois (1990)
13. Griewank, A., Juedes, D., Utke, J.: Algorithm 755: ADOL-C: A package for the automatic differentiation of algorithms written in C/C++. ACM Transactions on Mathematical Software **22**(2), 131–167 (1996). URL http://doi.acm.org/10.1145/229473.229474
14. Griewank, A., Walther, A.: Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, 2nd edn. No. 105 in Other Titles in Applied Mathematics. SIAM, Philadelphia, PA (2008). URL http://www.ec-securehost.com/SIAM/OT105.html
15. Hadamard, J.: La notion de différentiel dans l'enseignement. Mathematical Gazette **XIX**(236), 341–342 (1935)
16. Kubota, K.: PADRE2 Version 1 Users Manual. Research Memorandum RMI 90-01, Department of Mathematical Engineering and Information Physics, Faculty of Engineering, University of Tokyo, Tokyo, Japan (1990)
17. Kubota, K.: PADRE2, a Fortran precompiler yielding error estimates and second derivatives. In: Griewank and Corliss [11], pp. 251–262
18. Leibniz, G.W.: Machina arithmetica in qua non additio tantum et subtractio sedet multiplicatio nullo, division veropaene nullo animi labore peragantur. [An arithmetic machine which can be used to carry out not only addition and subtraction but also multiplication with no, and division with really almost no, intellectual exertion.]. Manuscript, Hannover (1685). A translation by Mark Kormes appears in 'A Source Book in Mathematics' by David Eugene Smith, Dover (1959)
19. Leibniz, G.W.: Nova methodvs pro maximis et minimis, itemque tangentibus, quae nec fractas, nec irrationales quantitates moratur, et singulare pro illis calculi genus. [A new method for maxima and minima as well as tangents, which is impeded neither by fractional nor irrational quantities, and a remarkable type of calculus for them.]. Acta Erutitorium (October 1684)
20. Menabrea, L.F.: Sketch of the analytical engine invented by Charles Babbage, with notes by the translator Augusta Ada King, Countess of Lovelace. Taylor's Scientific Memoirs **3**, 666–731 (1842)
21. Newton, I.: Philosophiae Naturalis Principia Mathematica. London (1687)

Sparse Jacobian Construction for Mapped Grid Visco-Resistive Magnetohydrodynamics

Daniel R. Reynolds and Ravi Samtaney

Abstract We apply the automatic differentiation tool OpenAD toward constructing a preconditioner for fully implicit simulations of mapped grid visco-resistive magnetohydrodynamics (MHD), used in modeling tokamak fusion devices. Our simulation framework employs a fully implicit formulation in time, and a mapped finite volume spatial discretization. We solve this model using inexact Newton-Krylov methods. Of critical importance in these iterative solvers is the development of an effective preconditioner, which typically requires knowledge of the Jacobian of the nonlinear residual function. However, due to significant nonlinearity within our PDE system, our mapped spatial discretization, and stencil adaptivity at physical boundaries, analytical derivation of these Jacobian entries is highly nontrivial. This paper therefore focuses on Jacobian construction using automatic differentiation. In particular, we discuss applying OpenAD to the case of a spatially-adaptive stencil patch that automatically handles differences between the domain interior and boundary, and configuring AD for reduced stencil approximations to the Jacobian. We investigate both scalar and vector tangent mode differentiation, along with simple finite difference approaches, to compare the resulting accuracy and efficiency of Jacobian construction in this application.

Keywords Forward mode • Iterative methods • Sparse Jacobian construction

D.R. Reynolds (✉)
Department of Mathematics, Southern Methodist University, Dallas, TX 75275, USA
e-mail: reynolds@smu.edu

R. Samtaney
Department of Mechanical Engineering, Division of Physical Science and Engineering, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
e-mail: Ravi.Samtaney@kaust.edu.sa

S. Forth et al. (eds.), *Recent Advances in Algorithmic Differentiation*, Lecture Notes in Computational Science and Engineering 87, DOI 10.1007/978-3-642-30023-3_2,

1 Introduction

In this paper, we examine application of the Automatic Differentiation (AD) tool OpenAD [12–14] toward fully implicit simulations of mapped grid visco-resistive magnetohydrodynamics (MHD). These simulations are used to study tokamak devices for magnetically-confined fusion plasmas. However, such problems are indicative of a much more expansive class of large-scale simulations involving multi-physics systems of partial differential equations (PDEs), and most of the work described herein will apply in that larger context. We note that similar efforts have been made in Jacobian construction within the context of compressible fluid dynamics [6, 11], and the current study complements that work through our investigation of an increasingly complex PDE system with more significant nonzero Jacobian structure.

This paper addresses using OpenAD to generate Jacobian components required within iterative solvers for nonlinear implicit equations arising from our PDE model. We begin by describing the model (Sect. 1.1), and our discretization and solver framework (Sect. 1.2). We then describe three competing approaches for Jacobian construction (Sect. 2): scalar mode AD, vector mode AD, and simple finite difference approximation, as well as the variety of code modifications that were required to enable these techniques. We then describe our experimental tests on these approaches and the ensuing results (Sect. 3), and conclude with some proposed optimizations for Jacobian construction in similar applications.

1.1 Model

We study visco-resistive MHD in cylindrical, (r, φ, z), coordinates [15],

$$\partial_t \mathbf{U} + \frac{1}{r}\partial_r (r\mathbf{F}(\mathbf{U})) + \partial_z \mathbf{H}(\mathbf{U}) + \frac{1}{r}\partial_\varphi \mathbf{G}(\mathbf{U}) = \mathbf{S}(\mathbf{U}) + \nabla \cdot \mathbf{F}_d(\mathbf{U}), \tag{1}$$

where $\mathbf{U} = \left(\rho, \rho u_r, \rho u_\varphi, \rho u_z, B_r, B_\varphi, B_z, e\right)$, with plasma density ρ, velocity $\mathbf{u} = \left(u_r, u_\varphi, u_z\right)$, magnetic induction $\mathbf{B} = \left(B_r, B_\varphi, B_z\right)$, total energy e, and radial location r. Here, the hyperbolic fluxes are given by

$$\mathbf{F} = \begin{pmatrix} \rho u_r, & \rho u_r^2 + \tilde{p} - B_r^2, & \rho u_r u_\varphi - B_r B_\varphi, & \rho u_r u_z - B_r B_z, & 0, \\ u_r B_\varphi - u_\varphi B_r, & u_r B_z - u_z B_r, & (e + \tilde{p}) u_r - (\mathbf{B}\cdot\mathbf{u}) B_r \end{pmatrix}, \tag{2}$$

$$\mathbf{G} = \begin{pmatrix} \rho u_\varphi, & \rho u_r u_\varphi - B_r B_\varphi, & \rho u_\varphi^2 + \tilde{p} - B_\varphi^2, & \rho u_z u_\varphi - B_z B_\varphi, \\ u_\varphi B_r - u_r B_\varphi, & 0, & u_\varphi B_z - u_z B_\varphi, & (e + \tilde{p}) u_\varphi - (\mathbf{B}\cdot\mathbf{u}) B_\varphi \end{pmatrix}, \tag{3}$$

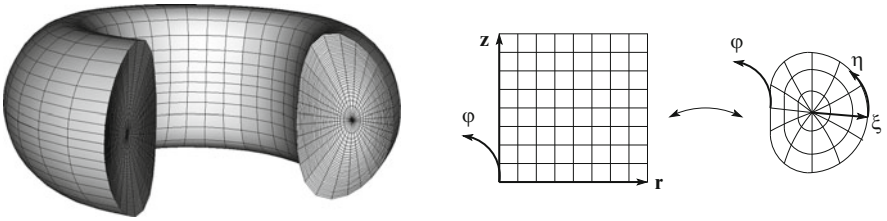

Fig. 1 *Left*: tokamak domain (a slice has been removed to show the poloidal cross-section). Note the coordinate singularity at the torus core. Cells near the core exhibit a loss of floating-point accuracy in evaluation of $\mathscr{J}$ in (6). *Right*: mapping between cylindrical and shaped domains

$$\mathbf{H} = \begin{pmatrix} \rho u_z, & \rho u_r u_z - B_r B_z, & \rho u_z u_\varphi - B_z B_\varphi, & \rho u_z^2 + \tilde{p} - B_z^2, \\ & u_z B_r - u_r B_z, & u_z B_\varphi - u_\varphi B_z, & 0, \quad (e + \tilde{p}) u_z - (\mathbf{B} \cdot \mathbf{u}) B_z \end{pmatrix}, \tag{4}$$

where $\tilde{p} = p + \frac{\mathbf{B}\cdot\mathbf{B}}{2}$ and pressure $p = \frac{2e}{3} - \frac{\rho \mathbf{u}\cdot\mathbf{u}}{3} - \frac{\mathbf{B}\cdot\mathbf{B}}{3}$. In this model, $\mathbf{S}(\mathbf{U})$ is a local source term resulting from the cylindrical coordinate system,

$$\mathbf{S} = \begin{pmatrix} 0, & B_z^2 - \rho u_z^2 - \tilde{p}, & 0, & \rho u_r u_z - B_r B_z, & 0, & 0, & u_z B_r - u_r B_z, & 0 \end{pmatrix} / r. \tag{5}$$

A similar cylindrical divergence is applied to the diffusive terms $\nabla \cdot \mathbf{F}_d(\mathbf{U})$,

$$\nabla \cdot \mathbf{F}_d(\mathbf{U}) = \Big(0, \ \nabla \cdot \overline{\tau}, \ \nabla \cdot (\overline{\tau}\mathbf{u} + \kappa \nabla T + \mathbf{B} \times (\eta(\nabla \times \mathbf{B}))) \, , \ -\nabla \times (\eta(\nabla \times \mathbf{B})) \, , \ 0 \Big),$$

where stress tensor $\overline{\tau} = \mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^T) - \frac{2}{3}\mu(\nabla \cdot \mathbf{u})\overline{I}$, temperature $T = \frac{2p}{\rho}$, μ, η and κ are input parameters for the plasma viscosity, resistivity and heat conductivity.

We map (1) to a shaped grid corresponding to the toroidal tokamak geometry (see Fig. 1). These mappings are encoded in the functions

$$\begin{aligned} \xi = \xi(r,z), \quad \eta = \eta(r,z), \quad \varphi = \varphi \qquad & \text{(cylindrical} \rightarrow \text{mapped)}, \\ r = r(\xi,\eta), \quad z = z(\xi,\eta), \quad \varphi = \varphi \qquad & \text{(mapped} \rightarrow \text{cylindrical)}, \\ \mathscr{J} = (\partial_\xi r)(\partial_\eta z) - (\partial_\eta r)(\partial_\xi z), \qquad & \mathscr{J}^{-1} = (\partial_r \xi)(\partial_z \eta) - (\partial_r \eta)(\partial_z \xi). \end{aligned} \tag{6}$$

Under this mapping, we rewrite the visco-resistive MHD system as

$$\partial_t \mathbf{U} + \tfrac{1}{\mathscr{J} r} \Big[\partial_\xi (r \tilde{\mathbf{F}}(\mathbf{U})) + \partial_\eta (r \tilde{\mathbf{H}}(\mathbf{U})) + \partial_\varphi (\tilde{\mathbf{G}}(\mathbf{U})) \Big] = \mathbf{S}(\mathbf{U}) + \nabla \cdot \tilde{\mathbf{F}}_d(\mathbf{U}), \tag{7}$$

$$\begin{aligned} \text{where} \quad \tilde{\mathbf{F}} &= \mathscr{J} \, (\partial_r \xi \, \mathbf{F} + \partial_z \xi \, \mathbf{H}) \ = \ \partial_\eta z \, \mathbf{F} - \partial_\eta r \, \mathbf{H}, \qquad \tilde{\mathbf{G}} \ = \ \mathscr{J} \mathbf{G}, \\ \tilde{\mathbf{H}} &= \mathscr{J} \, (\partial_r \eta \, \mathbf{F} + \partial_z \eta \, \mathbf{H}) \ = \ -\partial_\xi z \, \mathbf{F} + \partial_\xi r \, \mathbf{H}. \end{aligned}$$

Similar transformations are performed on the diffusive fluxes $\nabla \cdot \tilde{\mathbf{F}}_d(\mathbf{U})$. We also employ a 2D version of this model for simulations within the poloidal, (ξ, η), plane.