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 Leibnitz 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.

Key words: 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 succcessor 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 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

Bruce Christianson
School of Computer Science. University of Hertfordshire, College Lane, Hatfield England Europe, b.christianson@herts.ac.uk

[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.

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

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 [19, 18] 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.

---

[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 programme of reform that eleven of the sixteen 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 $\bar{w}, \bar{x}$.

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

[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 infinitisimal approach [6], but significantly he made no changes to Leibniz's original notation for them. Leibnitz's formulation allows for the possibility of non-negligable 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 Leibnitz is careful to write $d(xv) = xdv + vdx$, without the term $dxdv$.

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

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

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

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

---

[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.

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

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

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 Leibnitz notation the forward pass gives

[10] The variables x and y may be vectors: in this case the corresponding differential dx and barrential 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 $K^i_{jk} = \partial^2_{jk} f^i = \partial^2 f^i / \partial x^j \partial x^k$.

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

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

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

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,

---

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

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

or, in Leibniz notation

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

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 same 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

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

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

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

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: 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 Leibnitz 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)