ALGEBRAIC SIMPLIFIER INCORPORATING NON-STANDARD ANALYSIS

BY EDGAR B. REEVES JR.

DECEMBER 2, 1996

COMMITTEE MEMBERS:

Dr. David E. Leasure (chair)

Dr. David R. Thomas

Dr. George D. Tintera
Contents

Abstract 01
Background and Rationale 02
Narrative 12
Environment 17
Procedure 18
End Results 30
Bibliography 33
Appendix 34
Abstract

This thesis discusses an algebraic simplifier program that can be used as either a subprogram in a more complete theorem prover, or to perform as a stand-alone program. The overall method uses natural deduction to implement Abraham Robinson's non-standard analysis for polynomial derivatives. The program also automates the symbolic calculation and algebraic simplification of polynomial, exponential, and trigonometric derivatives.
Background and Rationale

This algebraic simplifier is a program that can evaluate and simplify algebraic expressions, and can symbolically calculate many of the derivatives of elementary calculus. It calculates many elementary derivatives using the non-standard formulation of the definition of the derivative. A principal use of such an algebraic simplifiers is as a subprogram in a larger theorem prover, such as the one using non-standard analysis which was done at The University of Texas (Ballantyne, 1977).

A description of non-standard analysis follows in section A, and more on the background and rationale of this program is in section B.

Section A: Isaac Newton is generally credited with the first development of "infinitesimal calculus", although G. W. Leibniz seems to have discovered it independently not long afterward. "Infinitesimal numbers" were used in calculus by mathematicians, scientists, and engineers until the mid-nineteenth century, when Weierstrass seemingly banished infinitesimals from analysis forever. He formulated the calculus proofs using the epsilon-delta notation, which restricts them to Archimedean numbers in the standard number system. For a number x to be Archimedean means that it is not the case that $0 < x < 1/n$ for all positive integers n. On the other hand, if the number x were one of Robinson's non-standard infinitesimals, then the above inequality would hold. Also, infinitely large numbers can be introduced as reciprocals of
the infinitesimals.

Weierstrass’ methods of proof were designed to remove certain contradictions from analysis as it was formerly done with infinitesimals. Bishop Berkeley discussed these contradictions in his paper of 1734, addressed to "an infidel mathematician", thought to be astronomer Edmund Halley. It was believed that he had persuaded one of the Bishop’s friends of "the inconceivability of Christianity". Berkeley said that if a mathematician could attempt theology, then he would attempt mathematics. The resulting critique was not dealt with satisfactorily for more than one hundred years; that is, not until Weierstrass’ work.

One illustration of the type of contradiction under consideration is from Newton’s calculation of the instantaneous velocity of a falling body. For example, this turns out to be 32 feet per second at time $t = 1$ second. He obtained this by differentiating the position function $s = 16t^2$, which is a simplified form of $S = S_0 + V_0t + \frac{1}{2}gt^2$, with $g = 32$ fps*s, and is adequate for our purposes. One definition of the derivative is

$$\frac{ds}{dt} = \lim_{\Delta t \to 0} \frac{f(s+\Delta t) - f(s)}{\Delta t},$$

where the $\Delta t$ quantities are "infinitesimally small numbers". The result of the differentiation is $32 + 16\Delta t$ feet per second at $t = 1$ second. The "infinitesimal quantity", $16\Delta t$, is neglected in order to arrive at the 32 fps answer. However, in Newton’s time, the derivative could not be calculated without using infinitesimals, and, consequently, one of them actually occurs in the first formulation of the answer above. An infinitesimal cannot
be 0, since it occurs in the denominator of a fraction in the
calculations of the derivative, yet it was neglected as though it
were 0 in the final answer. This material is referenced in any
sufficiently detailed history of mathematics.

Such contradictions were removed by expunging infinitesimals
from the system. However, since the mid-twentieth century, they
have been reinstated in Abraham Robinson’s non-standard analysis.
For example, the non-standard number system is constructed so that
if the non-standard number x1 differs from the non-standard number
x2 by an infinitesimal amount, we write x1 - x2, and if this is so,
then the elementary calculus formula for the derivative becomes

\[(1) \quad \frac{dy}{dx} = \frac{f(x_2) - f(x_1)}{x_2 - x_1}, \text{ at } x_1.\]

This expression is easier to automate than the usual form, which is

\[(2) \quad \frac{dy}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}, \text{ at } x.\]

The foundations of non-standard analysis are found in modern
logic. The completeness theorem states that a set of sentences is
logically consistent (no contradiction can be deduced from them) if
and only if the sentences have a model; that is, if and only if
there is some "universe" in which they are all true. The
compactness theorem then proceeds as follows: Suppose that there
is a collection of sentences in the language L, such that every
finite subset of the collection L is true in the standard universe.
If this is the case, then there is a non-standard universe in which the entire collection is true at the same time. This result follows from the completeness theorem, because if every finite subset of the collection \( L \) is true in the standard universe, then every finite subset is logically consistent. Therefore, since any deduction can make use of only a finite number of premises, the entire collection is logically consistent. Then, by the completeness theorem, there is a (non-standard) universe which is a model for the entire collection, so that the entire collection is true in that universe. As a consequence, there is a sense in which the infinitesimals exist, though not in the sense that some of the subject matter of physics exists. Model theory does not address such ontological questions, but it does tell us that we should not get into any more trouble reasoning in the non-standard universe than we do in the standard one.

The compactness theorem states that if \( L \) is a formal language as before, and if \( K \) is the set of all true sentences of \( L \) in the standard universe, then each of the sentences in \( K \) is true, with suitable modifications, in the non-standard universe. Even though we do not know all of the elements of \( K \), we can still reason and draw conclusions about it. One consideration is that since all standard real numbers are Archimedean and some of the non-standard reals are not, the Archimedean property cannot be expressed in one of the sentences in \( L \), nor can Berkeley's contradiction. Of course, since \( K \) is a subset of \( L \), then sentences expressing either the Archimedean property or the contradiction are not in \( K \). The
"pseudoobjects" in the non-standard real number system behave formally like standard objects, yet differ with respect to important properties which are not formalized by L.

Even though the standard real number system R and the non-standard system R* are conceptually distinct, it is useful to think of the standard numbers as being embedded in the non-standard reals. Since the non-standard universe R* is a model for L, every sentence in K (which is true for the standard real numbers) has a true interpretation in R*. In particular, we will identify the ("pseudo") object "2" in R* with the ("real") object "2" in the standard reals, so that if we continue in this manner, R* will contain the standard real numbers, as well as the infinite collection of infinitesimal and infinite quantities in R*. R* is thought of as having an infinite collection of "pseudoreals" clustered around every standard real number, so that each of these infinite collections is within an infinitesimal neighborhood of a given standard number. This standard real number is called the standard part of each of the "pseudoreals" in its infinitesimal neighborhood. That is, if e is an infinitesimal number and r is a standard real number, then p = r + e is one of the infinitely many "pseudoreals" in the infinitesimal neighborhood of r, and r is called the standard part of p.

Note that the Newtonian falling body problem can now be formulated in the non-standard number system, so that instead of defining the instantaneous velocity as the ratio of infinitesimal increments, it is defined to be the standard part of that ratio.
and \( ds, \, dt, \) and the ratio of them, \( ds/dt, \) are now non-standard numbers. The careful distinguishing between the non-standard \( ds/dt \) and the standard real number \( v \) avoids the contradiction, which many of the mathematicians and scientists before Weierstrass simply ignored. We can conclude rigorously, without resorting to arguments involving limits, that \( v = 32 \) feet per second exactly, at \( t = 1 \) second.

In general, results such as proofs of theorems, which are obtained in one system, can be "imported" into the other without loss of validity. In this project, derivatives obtained in the non-standard number system are converted and returned to the standard number system.

The development of non-standard analysis is proceeding in many pure and applied areas. For example, work has been done in non-standard probability, non-standard dynamics, and a non-standard model of space and time has been used to provide new existence results for the Boltzman equation. In some areas of mathematical physics, such as those involving large finite collections of particles, the physical situation may be more accurately modeled by a hyperfinite set (A hyperfinite set is infinite, but is finite from the non-standard point of view, and therefore, inherits many of the properties of finite sets.). Also, heuristic reasoning with infinitesimals, which even the "purest" of pure mathematicians might do in a weak moment, can be made rigorous in a way that it could not before (Cutland, 1988).
Section B: Another advantage of this program is that it provides a way to obtain symbolic derivatives, rather than numerical approximation methods. This provides heuristic problem-solving advantages, and it is sometimes helpful to have a derivative function rather than a numerical answer.

Also, this simplifier employs a natural-deduction system, which is easier to trace than, for example, a resolution theorem-prover like J. A. Robinson's. The resolution theorem prover rewrites the hypotheses and the negation of the conclusion of a proposed theorem as a series of conjunctions of disjunctions, that is, in conjunctive normal form. It then tries to find a contradiction, which, if successful, proves the theorem. This rewriting of the original theorem completely obscures the original structure to human readers. Also, it is virtually impossible to trace intelligibly the deduction steps followed by the computer. This obviously would preclude its use in an interactive environment, and in any case tends to obfuscate rather than facilitate the research and solution of problems. For example, in a natural deduction system, the following might be displayed during a proof:

H1: P
H2: (P -> Q)
H3: (R & Q -> S)

C1: Q
C2: (R -> S).
This would indicate that each of the \( H(n) \) is an hypothesis, and that each of the \( C(m) \) is a goal. On the other hand, a resolution system would display the same situation with a set of disjunctive clauses:

1. \( P \)
2. \( \neg P \) or \( Q \)
3. \( \neg R \) or \( \neg Q \) or \( S \)
4. \( \neg Q \) or \( R \)
5. \( \neg Q \) or \( \neg S \).

Even though the two representations are logically equivalent, all information about goals, which is what we are trying to prove, has been lost in the second (Barr, 1981).

Some of the advantages of the Logistica programming language can be appreciated by comparing a program segment written in Scheme to a similar one in Logistica. Suppose that the following four differentiation rules were to be implemented in Scheme:

\[
dc/dx = 0 \text{ for } c \text{ a constant or a variable different from } x,
\]

\[
dx/dx = 1,
\]

\[
d/dx \ (u + v) = du/dx + dv/dx, \text{ and}
\]

\[
d/dx \ (uv) = u \ (dv/dx) + v \ (du/dx).
\]
About a dozen lines of Scheme code would be needed, and there would have to be eleven preliminary procedures to implement selectors, constructors, and predicates to do the following:

For e an expression, a(i) a term, m(j) a factor, and v(n) a variable,

- Determine whether e is a constant
- Determine whether e is a variable
- Determine whether v(1) = v(2)
- Determine whether e is a sum
- Determine whether e is a product
- Obtain the addend of the sum e
- Obtain the augend of the sum e
- Obtain the multiplier of the product e
- Obtain the multiplicand of the product e
- Construct the sum of a(1) and a(2)
- Construct the product of m(1) and m(2).

If the procedures to perform these functions were available, the Scheme code necessary to implement the four rules would be:

```
(define (deriv exp var)
  (cond ((constant? exp) 0)
        ((variable? exp)
         (if (same-variable? exp var) 1 0))
        ((sum? exp)
         (make-sum (deriv (addend exp) var)
                   (deriv (augend exp) var)))
```
((product? exp)
(make-sum
(make-product (multiplier exp)
   (deriv (multiplicand exp) var))
(make-product (deriv (multiplier exp) var)
   (multiplicand exp))))

This code would produce an unsimplified answer, and still more code would have to be written in order to simplify it. For example, the result of \( \frac{d}{dx} (xy) \) would be \( x*0 + 1*y \) instead of just \( y \), and more complex expressions, although answered correctly, would be much harder to read before simplification. The Logistica code necessary to produce the same unsimplified result would be essentially only four statements, which look very much like ordinary mathematical rules for differentiation.