The 30 Year Horizon

Manuel Bronstein  William Burge  Timothy Daly
James Davenport  Michael Dewar  Martin Dunstan
Albrecht Fortenbacher  Patrizia Gianni  Johannes Grabmeier
Jocelyn Guidry  Richard Jenks  Larry Lambe
Michael Monagan  Scott Morrison  William Sit
Jonathan Steinbach  Robert Sutor  Barry Trager
Stephen Watt  Jim Wen  Clifton Williamson

Volume Bibliography: Axiom Literature Citations
Portions Copyright (c) 2005 Timothy Daly

The Blue Bayou image Copyright (c) 2004 Jocelyn Guidry

Portions Copyright (c) 2004 Martin Dunstan
Portions Copyright (c) 2007 Alfredo Portes
Portions Copyright (c) 2007 Arthur Ralfs
Portions Copyright (c) 2005 Timothy Daly

Portions Copyright (c) 1991-2002,
The Numerical ALgorithms Group Ltd.
All rights reserved.

This book and the Axiom software is licensed as follows:

Redistribution and use in source and binary forms, with or
without modification, are permitted provided that the following
conditions are met:

- Redistributions of source code must retain the above
  copyright notice, this list of conditions and the
  following disclaimer.

- Redistributions in binary form must reproduce the above
  copyright notice, this list of conditions and the
  following disclaimer in the documentation and/or other
  materials provided with the distribution.

- Neither the name of The Numerical ALgorithms Group Ltd.
  nor the names of its contributors may be used to endorse
  or promote products derived from this software without
  specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND
CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,
BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.
Inclusion of names in the list of credits is based on historical information and is as accurate as possible. Inclusion of names does not in any way imply an endorsement but represents historical influence on Axiom development.

Michael Albaugh
Christian Aistleitner
S.J. Atkins
Stephen Balzac
Gerald Baumgartner
Nelson H. F. Beebe
Fred Blair
Raoul Bourquin
Peter A. Broadbery
Stephen Buchwald
William Burge
Robert Caviness
Tzu-Yi Chen
Gregory V. Chudnovsky
Jia Zhao Cong
Don Coppersmith
Gary Cornell
David Cyganski
Timothy Daly Jr.
James Demmel
Jack Dongarra
Claire DiCrescendo
Iain Duff
Brian Dupee
Heow Eide-Goodman
Bertfried Fauser
Brian Ford
Constantine Frangos
Marc Gaetano
Kathy Gerber
Holger Gollan
Stephen Gortler
Klaus Ebbe Grue
Oswald Gschnitzer
Gaetan Hache
Sven Hammarling
Richard Harke
Martin Hassner
Waldek Hebisch
Cyril Alberga
Richard Anderson
Henry Baker
Yuri Baransky
Gilbert Baumslag
Jay Belanger
Vladimir Bondarenko
Alexandre Bouyer
Martin Brock
Florian Bundschuh
Ralph Byers
Bruce Char
Cheekai Chin
Mark Clements
Josh Cohen
George Corliss
Meino Cramer
Nathaniel Daly
James H. Davenport
Didier Deshommes
Jean Della Dora
Sam Dooley
Lee Duhem
Dominique Duval
Lars Erickson
Stuart Feldman
Albrecht Fortenbacher
Timothy Freeman
Rudiger Gebauer
Patricia Gianni
Teresa Gomez-Diaz
Johannes Grabmeier
James Griesmer
Ming Gu
Steve Hague
Mike Hansen
Bill Hart
Arthur S. Hathaway
Karl Hegbloom
Roy Adler
George Andrews
Martin Baker
Michael Becker
David R. Barton
Karen Braman
Manuel Bronstein
Luanne Burns
Quentin Carpent
Ondrej Certik
David V. Chudnovsky
James Cloos
Christophe Coul
Robert Corless
Jeremy Du Croz
Timothy Daly Sr.
David Day
Michael Dewar
Gabriel Dos Reis
Lionel Ducos
Martin Dunstan
Robert Edwards
Richard Fateman
John Fletcher
George Frances
Korrinn Fu
Van de Geijn
Samantha Goldrich
Laureano Gonzalez-Vega
Matt Grayson
Vladimir Grinberg
Jocelyn Guidry
Satoshi Hamaguchi
Richard Hanson
Vilya Harvey
Dan Hatton
Ralf Hemmecke
<table>
<thead>
<tr>
<th>Henderson</th>
<th>Antoine Hersen</th>
<th>Roger House</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gernot Hueber</td>
<td>Pietro Iglio</td>
<td>Alejandro Jakubi</td>
</tr>
<tr>
<td>Richard Jenks</td>
<td>William Kahan</td>
<td>Kai Kaminski</td>
</tr>
<tr>
<td>Grant Keady</td>
<td>Wilfrid Kendall</td>
<td>Tony Kennedy</td>
</tr>
<tr>
<td>Ted Kosan</td>
<td>Paul Kosinski</td>
<td>Klaus Kusche</td>
</tr>
<tr>
<td>Bernhard Kutzler</td>
<td>Tim Lahey</td>
<td>Larry Lambe</td>
</tr>
<tr>
<td>Kaj Laurson</td>
<td>George L. Legendre</td>
<td>Franz Lehner</td>
</tr>
<tr>
<td>Frederic Lehobey</td>
<td>Michel Levaud</td>
<td>Howard Levy</td>
</tr>
<tr>
<td>Ren-Cang Li</td>
<td>Rudiger Loos</td>
<td>Michael Lucks</td>
</tr>
<tr>
<td>Richard Luczak</td>
<td>Cammi Maguire</td>
<td>Francois Maltey</td>
</tr>
<tr>
<td>Alasdair McAndrew</td>
<td>Bob McElrath</td>
<td>Michael McGirruck</td>
</tr>
<tr>
<td>Edi Meier</td>
<td>Ian Meikle</td>
<td>David Mentre</td>
</tr>
<tr>
<td>Victor S. Miller</td>
<td>Gerard Milmeister</td>
<td>Mohammed Mobarak</td>
</tr>
<tr>
<td>H. Michael Moeller</td>
<td>Michael Monagan</td>
<td>Marc Moreno-Maza</td>
</tr>
<tr>
<td>Scott Morrison</td>
<td>Joel Moses</td>
<td>Mark Murray</td>
</tr>
<tr>
<td>William Naylor</td>
<td>Patrice Naudin</td>
<td>C. Andrew Neff</td>
</tr>
<tr>
<td>John Nelder</td>
<td>Godfrey Nolan</td>
<td>Arthur Norman</td>
</tr>
<tr>
<td>Jinzhong Niu</td>
<td>Michael O'Connor</td>
<td>Summat Oemrawsingh</td>
</tr>
<tr>
<td>Kostas Dikonomou</td>
<td>Humberto Ortiz-Zuazaga</td>
<td>Julian A. Padget</td>
</tr>
<tr>
<td>Bill Page</td>
<td>David Parnas</td>
<td>Susan Pelzel</td>
</tr>
<tr>
<td>Michel Petitot</td>
<td>Didier Pinchon</td>
<td>Ayal Pinkus</td>
</tr>
<tr>
<td>Frederick H. Pitts</td>
<td>Jose Alfredo Portes</td>
<td>Gregorio Quintana-Orti</td>
</tr>
<tr>
<td>Claude Quitte</td>
<td>Arthur C. Ralfs</td>
<td>Norman Ramsey</td>
</tr>
<tr>
<td>Anatoly Raportirenko</td>
<td>Albert D. Rich</td>
<td>Michael Richardson</td>
</tr>
<tr>
<td>Guilherme Reis</td>
<td>Huan Ren</td>
<td>Reno Ruiooo</td>
</tr>
<tr>
<td>Jean Rivlin</td>
<td>Nicolas Robidoux</td>
<td>Simon Robinson</td>
</tr>
<tr>
<td>Raymond Rogers</td>
<td>Michael Rothstein</td>
<td>Martin Rubey</td>
</tr>
<tr>
<td>Philip Santas</td>
<td>Alfred Scherhorn</td>
<td>William Schelter</td>
</tr>
<tr>
<td>Gerhard Schneider</td>
<td>Martin Schoenert</td>
<td>Marshall Schor</td>
</tr>
<tr>
<td>Frithjof Schulze</td>
<td>Fritz Schwarz</td>
<td>Steven Segletes</td>
</tr>
<tr>
<td>V. Sina</td>
<td>Nick Simicich</td>
<td>William Sit</td>
</tr>
<tr>
<td>Elena Smirnova</td>
<td>Jonathan Steinbach</td>
<td>Fabio Stumbo</td>
</tr>
<tr>
<td>Christine Sundaresan</td>
<td>Robert Sutor</td>
<td>Moss E. Sweedler</td>
</tr>
<tr>
<td>Eugene Surowitz</td>
<td>Max Tegmark</td>
<td>T. Doug Telford</td>
</tr>
<tr>
<td>James Thatcher</td>
<td>Balbir Thomas</td>
<td>Mike Thomas</td>
</tr>
<tr>
<td>Dylan Thurston</td>
<td>Steve Toleque</td>
<td>Barry Trager</td>
</tr>
<tr>
<td>Themos T. Tsikas</td>
<td>Gregory Vanuxem</td>
<td>Bernhard Wall</td>
</tr>
<tr>
<td>Stephen Watt</td>
<td>Jaap Weel</td>
<td>Juergen Weiss</td>
</tr>
<tr>
<td>M. Weller</td>
<td>Mark Wegman</td>
<td>James Wen</td>
</tr>
<tr>
<td>Thorsten Werther</td>
<td>Michael Wester</td>
<td>R. Clint Whaley</td>
</tr>
<tr>
<td>John M. Wiley</td>
<td>Berhard Will</td>
<td>Clifton J. Williamson</td>
</tr>
<tr>
<td>Stephen Wilson</td>
<td>Shmuel Winograd</td>
<td>Robert Wisbauer</td>
</tr>
<tr>
<td>Sandra Wityak</td>
<td>Waldemar Wiwianka</td>
<td>Knut Wolf</td>
</tr>
<tr>
<td>Liu Xiaojun</td>
<td>Clifford Yapp</td>
<td>David Yun</td>
</tr>
<tr>
<td>Vadim Zhilyak</td>
<td>Richard Zippel</td>
<td>Evelyn Zoernack</td>
</tr>
<tr>
<td>Bruno Zuercher</td>
<td>Dan Zwilinger</td>
<td></td>
</tr>
</tbody>
</table>
# Contents

1.1 Axiom Citations in the Literature ................................................. 3
A ............................................. 3
B ............................................. 4
C ............................................. 9
D ............................................. 10
E ............................................ 16
F ............................................ 16
G ............................................ 17
H ............................................ 20
J ............................................ 21
K ............................................ 23
L ............................................ 24
M ............................................ 27
N ............................................ 28
O ............................................ 30
P ............................................ 30
R ............................................ 31
S ............................................ 32
T ............................................ 36
V ............................................ 36
W ............................................ 37
Y ............................................ 40
Z ............................................ 41

1.2 Axiom Citations of External Sources .............................................. 42
A ............................................. 42
B ............................................. 42
C ............................................. 46
D ............................................. 47
F ............................................. 50
G ............................................. 51
H ............................................. 54
I ............................................. 56
J ............................................. 56
K ............................................. 57
L ............................................. 58
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>60</td>
</tr>
<tr>
<td>N</td>
<td>62</td>
</tr>
<tr>
<td>O</td>
<td>62</td>
</tr>
<tr>
<td>P</td>
<td>62</td>
</tr>
<tr>
<td>Q</td>
<td>65</td>
</tr>
<tr>
<td>R</td>
<td>65</td>
</tr>
<tr>
<td>S</td>
<td>67</td>
</tr>
<tr>
<td>T</td>
<td>68</td>
</tr>
<tr>
<td>U</td>
<td>69</td>
</tr>
<tr>
<td>V</td>
<td>69</td>
</tr>
<tr>
<td>W</td>
<td>69</td>
</tr>
<tr>
<td>Y</td>
<td>71</td>
</tr>
<tr>
<td>1.3 Special Topics</td>
<td></td>
</tr>
<tr>
<td>Solving Systems of Equations</td>
<td>72</td>
</tr>
<tr>
<td>Numerical Algorithms</td>
<td>72</td>
</tr>
<tr>
<td>Special Functions</td>
<td>73</td>
</tr>
<tr>
<td>Polynomial GCD</td>
<td>74</td>
</tr>
<tr>
<td>Category Theory</td>
<td>76</td>
</tr>
<tr>
<td>Proving Axiom Correct</td>
<td>77</td>
</tr>
<tr>
<td>Interval Arithmetic</td>
<td>83</td>
</tr>
<tr>
<td>Numerics</td>
<td>84</td>
</tr>
<tr>
<td>Advanced Documentation</td>
<td>84</td>
</tr>
<tr>
<td>Differential Equations</td>
<td>85</td>
</tr>
<tr>
<td>Expression Simplification</td>
<td>89</td>
</tr>
<tr>
<td>Integration</td>
<td>89</td>
</tr>
<tr>
<td>Partial Fraction Decomposition</td>
<td>95</td>
</tr>
<tr>
<td>Ore Rings</td>
<td>95</td>
</tr>
</tbody>
</table>
New Foreword

On October 1, 2001 Axiom was withdrawn from the market and ended life as a commercial product. On September 3, 2002 Axiom was released under the Modified BSD license, including this document. On August 27, 2003 Axiom was released as free and open source software available for download from the Free Software Foundation’s website, Savannah.

Work on Axiom has had the generous support of the Center for Algorithms and Interactive Scientific Computation (CAISS) at City College of New York. Special thanks go to Dr. Gilbert Baumslag for his support of the long term goal.

The online version of this documentation is roughly 1000 pages. In order to make printed versions we’ve broken it up into three volumes. The first volume is tutorial in nature. The second volume is for programmers. The third volume is reference material. We’ve also added a fourth volume for developers. All of these changes represent an experiment in print-on-demand delivery of documentation. Time will tell whether the experiment succeeded.

Axiom has been in existence for over thirty years. It is estimated to contain about three hundred man-years of research and has, as of September 3, 2003, 143 people listed in the credits. All of these people have contributed directly or indirectly to making Axiom available. Axiom is being passed to the next generation. I’m looking forward to future milestones.

With that in mind I’ve introduced the theme of the “30 year horizon”. We must invent the tools that support the Computational Mathematician working 30 years from now. How will research be done when every bit of mathematical knowledge is online and instantly available? What happens when we scale Axiom by a factor of 100, giving us 1.1 million domains? How will we integrate theory with code? How will we integrate theorems and proofs of the mathematics with space-time complexity proofs and running code? What visualization tools are needed? How do we support the conceptual structures and semantics of mathematics in effective ways? How do we support results from the sciences? How do we teach the next generation to be effective Computational Mathematicians?

The “30 year horizon” is much nearer than it appears.

Tim Daly
CAISS, City College of New York
November 10, 2003 ((iHy))
A bibliography of Axiom references which are used throughout Axiom. The first section contains literature that mentions Axiom, initially derived with permission from Nelson Beebe’s collection. The second section contains references from Axiom to the literature. The third section sorts papers by topic.
Bibliography

1.1 Axiom Citations in the Literature

A

[ACM 89] ACM, editor
Proceedings of the ACM-SIGSAM 1989 International Symposium on Symbolic and
Algebraic Computation, ISSAC '89 ACM Press, New York, NY 10036, USA, 1989,

[ACM 94] ACM, editor
ISSAC '94. Proceedings of the International Symposium on Symbolic and Algebraic
LCCN QA76.95.I59 1994

[ACS 91] D. Augot; P. Charpin; N. Sendrier
“The minimum distance of some binary codes via the Newton’s identities”
In Cohen and Charpin [CC91], pages 65-73 ISBN 0-387-54303-1 (New York), 3-540-
54303-1 (Berlin). LCCN QA268.E95 1990

“An Introduction to Gröbner Bases”

[Andrews 84] George E. Andrews
“Ramanujan and SCRATCHPAD”
In Golden and Hussain [GH84], pages 383-??

“Application of Scratchpad to problems in special functions and combinatorics”
In Janssen [Jan88], pages 158-?? ISBN 3-540-18928-9, 0-387-18928-9 LCCN
QA155.7.E4T74 1988

[Anon 91] Anonymous editor
Programming environments for high-level scientific problem solving.
IFIP TC2/WG 2.5 working conference. IFIP Transactions. A Computer Science and Technology, A-2:??, 1992. CODEN ITATEC. ISSN 0926-5473

Anonymous
GAMM 94 annual meeting. Zeitschrift fur Angewandte Mathematik und Physik, 75 (suppl. 2), 1995, CODEN ZAMMAX, ISSN 0044-2267

B

“SPAD language type checker”
github.com/cahirwpz/phd
The project aims to deliver a new type checker for SPAD language. Several improvements over current type checker are planned.

- introduce better type inference
- introduce modern language constructs
- produce understandable diagnostic messages
- eliminate well known bugs in the type system
- find new type errors

“An interactive facility for symbolic mathematics”

“LPL: LISP programming language”
IBM Research Report, RC3062 Sept 1970

“On the implementation of dynamic evaluation”
pdf.aminer.org/000/449/014/on_the_implementation_of_dynamic_evaluation.pdf

Dynamic evaluation is a technique for producing multiple results according to a decision tree which evolves with program execution. Sometimes it is desired to produce results for all possible branches in the decision tree, while on other occasions, it may be sufficient to compute a single result which satisfies certain properties. This technique finds use in computer algebra where computing the correct result depends on recognizing and properly handling special cases of parameters. In previous work, programs using dynamic evaluation have explored all branches of decision trees by repeating the computations prior to decision points.
This paper presents two new implementations of dynamic evaluation which avoid recomputing intermediate results. The first approach uses Scheme “continuations” to record state for resuming program execution. The second implementation uses the Unix “fork” operation to form new processes to explore alternative branches in parallel.

[Boehm 89] Boehm, Hans-J.
“Type inference in the presence of type abstraction”
ACM SIGPLAN Notices, 24(7) pp192-206 July 1989 CODEN SINODQ ISSN 0362-1340
www.acm.org/pubs/citations/proceedings/pldi/73141/p192-boehm

A number of recent programming language designs incorporate a type checking system based on the Girard-Reynolds polymorphic \(\lambda\)-calculus. This allows the construction of general purpose, reusable software without sacrificing compile-time type checking. A major factor constraining the implementation of these languages is the difficulty of automatically inferring the lengthy type information that is otherwise required if full use is made of these languages. There is no known algorithm to solve any natural and fully general formulation of the “type inference” problem. One very reasonable formulation of the problem is known to be undecidable.

Here we define a restricted version of the type inference problem and present an efficient algorithm for its solution. We argue that the restriction is sufficiently weak to be unobtrusive in practice.

[Boulton 04] Boulton, Richard; Hardy, Ruth; Gottliebsen, Hanne; Martin, Ursula
“Design verification for control engineering”
Proc Fourth International Conference on Integrated Formal Methods, April 2004

We introduce control engineering as a new domain of application for formal methods. We discuss design verification, drawing attention to the role played by diagrammatic evaluation criteria involving numeric plots of a design, such as Nichols and Bode plots. We show that symbolic computation and computational logic can be used to discharge these criteria and provide symbolic, automated, and very general alternatives to these standard numeric tests. We illustrate our work with reference to a standard reference model drawn from military avionics.

[Boulanger 91] Boulanger, Jean-Louis
“Etude de la compilation de scratchpad 2”
Rapport de DEA Universite dl lille 1, Sept 1991

[Boulanger 93a] Boulanger, Jean-Louis
“Axiom, language fonctionnel à développement objet”
IT 255, Oct 1993

[Boulanger 93b] Boulanger, Jean-Louis
“AXIOM, A Functional Language with Object Oriented Development”
We present in this paper, a study about the computer algebra system Axiom, which gives us many very interesting Software engineering concepts. This language is a functional language with an Object Oriented Development. This feature is very important for modeling the mathematical world (Hierarchy) and provides a running with mathematical sense. (All objects are functions). We present many problems of running and development in Axiom. We can note that Aiom is the only system of this category.

[Boulanger 94] Boulanger, J.L.
“Object Oriented Method for Axiom”
ACM SIGPLAN Notices, 30(2) pp33-41 February 1995 CODEN SINODQ ISSN 0362-1340

Axiom is a very powerful computer algebra system which combines two language paradigms (functional and OOP). Mathematical world is complex and mathematicians use abstraction to design it. This paper presents some aspects of the object oriented development in Axiom. The Axiom programming is based on several new tools for object oriented development, it uses two levels of class and some operations such that coerce, retract, or convert which permit the type evolution. These notions introduce the concept of multi-view.

[Bronstein 87] Bronstein, Manuel
“Integration of Algebraic and Mixed Functions” in [Wit87], p18

[Bronstein 89] Bronstein, M.
“Simplification of real elementary functions”

We describe an algorithm, based on Risch’s real structure theorem, that determines explicitly all the algebraic relations among a given set of real elementary functions. We also provide examples from its implementation that illustrate the advantages over the use of complex logarithms and exponentials.

[Bronstein 91a] Bronstein, M.

[Bronstein 91b] Bronstein, M.
“The Risch differential equation on an algebraic curve”

[Bronstein 92] Bronstein, M.
“Linear Ordinary Differential Equations: breaking through the order 2 barrier”
www-sop.inria.fr/cafe/Manuel.Bronstein/publications/issac92.ps.gz

A major subproblem for algorithms that either factor ordinary linear differential equations or compute their closed form solutions is to find their solutions $y$ which satisfy $y'/y \in K(x)$ where $K$ is the constant field for the coefficients of the
equation. While a decision procedure for this subproblem was known in the 19th century, it requires factoring polynomials over \( \overline{K} \) and has not been implemented in full generality. We present here an efficient algorithm for this subproblem, which has been implemented in the AXIOM computer algebra system for equations of arbitrary order over arbitrary fields of characteristic 0. This algorithm never needs to compute with the individual complex singularities of the equation, and algebraic numbers are added only when they appear in the potential solutions. Implementation of the complete Singer algorithm for \( n = 2, 3 \) based on this building block is in progress.

[Bronstein 93] Bronstein, Manuel (ed)

[Brunelli 09] Brunelli, J.C.
“Streams and Lazy Evaluation Applied to Integrable Models”

[Bronstein 93] Bronstein, Manuel; Salvy, Bruno
“Full partial fraction decomposition of rational functions”
www.acm.org/pubs/citations/proceedings/issac/164081/p157-bronstein

[Bronstein 92a] Bronstein, Manuel
“Integration and Differential Equations in Computer Algebra”
We describe in this paper how the problems of computing indefinite integrals and solving linear ordinary differential equations in closed form are now solved by computer algebra systems. After a brief review of the mathematical history of those problems, we outline the two major algorithms for them (respectively the Risch and Singer algorithms) and the recent improvements on those algorithms which has allowed them to be implemented.

“Double-track into the future: MathCAD will gain new users with Standard and Plus versions”
Elektronik, 43(15) pp107-110, July 1994, CODEN EKRKAR ISSN 0013-5658

[Bronstein 97a] Bronstein, Manuel; Weil, Jacques-Arthur
“We present alternative algorithms for computing symmetric powers of linear ordinary differential operators. Our algorithms are applicable to operators with coefficients in arbitrary integral domains and become faster than the traditional methods for symmetric powers of sufficiently large order, or over sufficiently complicated coefficient domains. The basic ideas are also applicable to other computations involving cyclic vector techniques, such as exterior powers of differential or difference operators.”

www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html
[Borwein 00] Borwein, Jonathan
“Multimedia tools for communicating mathematics”
Springer-Verlag ISBN 3-540-42450-4 p58

[Brown 94] Brown, R.; Tonks, A.
“Calculations with simplicial and cubical groups in AXIOM”
Journal of Symbolic Computation 17(2) pp159-179 February 1994 CODEN JSYCEH
ISSN 0747-7171

[Brown 95] Brown, Ronald; Dreckmann, Winfried
“Domains of data and domains of terms in AXIOM”
The main new concept we wish to illustrate in this paper is a distinction between “domains of data” and “domains of terms”, and its use in the programming of certain mathematical structures. Although this distinction is implicit in much of the programming work that has gone into the construction of Axiom categories and domains, we believe that a formalisation of this is new, that standards and conventions are necessary and will be useful in various other contexts. We shall show how this concept may be used for the coding of free categories and groupoids on directed graphs.

[Buchberger 85] Buchberger, Bruno; Caviness, Bob F. (eds)

[Buhl 05] Buhl, Soren L.
“Some Reflections on Integrating a Computer Algebra System in R”

[Burge 91] Burge, W.H.
“Scratchpad and the Rogers-Ramanujan identities”

[Burge 87] Burge, W.; Watt, S.
“Infinite structures in SCRATCHPAD II”
Technical Report RC 12794 (#57573) IBM Thomas J. Watson Research Center, Box 218, Yorktown Heights, NY 10598, USA 1987

“Streams and Power Series”
in [Wit87], pp9-12

[Burge 89] Burge, W. H.; Watt, S. M.
“Infinite structures in Scratchpad II”
1.1. AXIOM CITATIONS IN THE LITERATURE

C

[Calmet 94] Calmet, J. (ed)
Rhine Workshop on Computer Algebra, Proceedings. Universität Karsruhe, Karlsruhe, Germany 1994

[Camion 92] Camion, Paul; Courteau, Bernard; Montpetit, Andre
“Un problème combinatoire dans les graphs de Hamming et sa solution en Scratchpad”
“A combinatorial problem in Hamming Graphs and its solution in Scratchpad”
Rapports de recherche 1586, Institut National de Recherche en Informatique et en Automatique, Le Chesnay, France, January 1992, 12pp

[Caprotti] Caprotti, Olga; Cohen, Arjeh M.; Riem, Manfred
“Java Phrasebooks for Computer Algebra and Automated Deduction”

[Caprriott 99] Capriotti, O.; Carlisle, D.
“OpenMath and MathML: Semantic Mark Up for Mathematics”
www.acm.org/crossroads/xrds6-2/openmath.html

[Capriotti (a)] Caprotti, Olga; Cohen, Arjeh M.; Cuypers, Hans; Sterk, Hans
“OpenMath Technology for Interactive Mathematical Documents”
www.win.tue.nl/~hansc/lisbon.pdf

[Carpent] Carpent, Quentin; Conil, Christophe
“Utilisation de logiciels libres pour la réalisation de TP MT26” (2004)

[Chudnovsky 85] Chudnovsky, D.V; Chudnovsky, G.V.
“Elliptic Curve Calculations in Scratchpad II”
Scratchpad II Newsletter 1 (1) (1985)

[Chudnovsky 87] Chudnovsky, D.V; Chudnovsky, G.V.
“New Analytic Methods of Polynomial Root Finding”
in [Wit87], p2

[Chudnovsky 89] Chudnovsky, D.V. and Chudnovsky, G.V.
“The computation of classical constants”

[Chudnovsky 86] Chudnovsky, David; Jenks, Richard
“Computers in Mathematics”

[Cohen] Cohen, Arjeh; Cuypers, M.; Barreiro, Hans; Reinaldo, Ernesto; Sterk, Hans
“Interactive Mathematical Documents on the Web”
Springer 978354002576-c1.pdf
[Cohen 91] Cohen, G.; Charpin, P.; (ed)

[Conrad (a)] Conrad, Marc; French, Tim; Maple, Carsten; Pott, Sandra
“Approaching Inheritance from a Natural Mathematical Perspective and from a Java Driven Viewpoint: a Comparative Review”

It is well-known that few object-oriented programming languages allow objects to change their nature at run-time. There have been a number of reasons presented for this, but it appears that there is a real need for matters to change. In this paper we discuss the need for object-oriented programming languages to reflect the dynamic nature of problems, particularly those arising in a mathematical context. It is from this context that we present a framework that realistically represents the dynamic and evolving characteristic of problems and algorithms.

[Conrad (b)] Conrad, Marc; French, Tim; Maple, Carsten; Pott, Sandra
“Mathematical Use Cases lead naturally to non-standard Inheritance Relationships: How to make them accessible in a mainstream language?”

Conceptually there is a strong correspondence between Mathematical Reasoning and Object-Oriented techniques. We investigate how the ideas of Method Renaming, Dynamic Inheritance and Interclassing can be used to strengthen this relationship. A discussion is initiated concerning the feasibility of each of these features.

[Cuypers] Cuypers, Hans; Hendriks, Maxim; Knopper, Jan Willem
“Interactive Geometry inside MathDox”
www.win.tue.nl/~hansc/MathDox_and_InterGeo_paper.pdf

D

[Dalmas] Dalmas, Stéphane, Gaétano, Marc, and Watt, Stephen
“An OpenMath 1.0 Implementation”
citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.116.4401.pdf

[Dalmas 92] Dalmas, S.
“A polymorphic functional language applied to symbolic computation”

[Daly 88] Daly, Timothy
“Axiom in an Educational Setting”
Axiom course slide deck January 1988
1.1. AXIOM CITATIONS IN THE LITERATURE

[Daly 02] Daly, Timothy
“Axiom as open source”
SIGSAM Bulletin (ACM Special Interest Group on Symbolic and Algebraic Manipulation) 36(1) pp28-?? March 2002 CODEN SIGSBZ ISSN 0163-5824

[Daly 03] Daly, Timothy
“The Axiom Wiki Website”
axiom.axiom-developer.org

[Daly 06] Daly, Timothy
“Axiom Volume 1: Tutorial”
Lulu, Inc. 860 Aviation Parkway, Suite 300, Morrisville, NC 27560 USA, 2006 ISBN 141166597X 287pp
www.lulu.com/content/190827

[Daly 09] Daly, Timothy
“The Axiom Literate Documentation”
axiom-developer.org/axiom-website/documentation.html

[Daly 13] Daly, Timothy “Literate Programming in the Large” April 8-9, 2013 Portland Oregon
conf.writethedocs.org
daly.axiom-developer.org
www.youtube.com/watch?v=Av0PQDVTP4A

[Davenport 79a] Davenport, J.H.
“What can SCRATCHPAD/370 do?”
VM/370 SPAD.SCRIPTS August 24, 1979 SPAD.SCRIPT

[Davenport 80] Davenport, J.H.; Jenks, R.D.
“MODLISP – an Introduction”
Proc LISP80, 1980, and IBM RC8357 Oct 1980

[Davenport 84] Davenport, J.; Gianni, P.; Jenks, R.; Miller, V.; Morrison, S.; Rothstein, M.; Sundaresan, C.; Sutor, R.; Trager, B.
“Scratchpad”
Mathematical Sciences Department, IBM Thomas Watson Research Center 1984

[Davenport 84a] Davenport, James H.
“A New Algebra System”

[Davenport 85] Davenport, James H.
“The LISP/VM Foundation of Scratchpad II”
The Scratchpad II Newsletter, Volume 1, Number 1, September 1, 1985 IBM Corporation, Yorktown Heights, NY

[Davenport 88] Davenport, J.H.; Siret, Y.; Tournier, E.
staff.bath.ac.uk/masjhd/masternew.pdf
Axiom is a computer algebra system superficially like many others, but fundamentally different in its internal construction, and therefore in the possibilities it offers to its users and programmers. In these lecture notes, we will explain, by example, the methodology that the author uses for programming substantial bits of mathematics in Axiom.
www.nag.co.uk/doc/TechRep/axiomtr.html

[Davenport 93] Davenport, J. H.
“Primality testing revisited”
Downer’s Grove, IL, USA and Oxford, UK, August 1993
www.nag.co.uk/doc/TechRep/axiomtr.html

[Davenport (a)] Davenport, James; Faure, Christèle
“The Unknown in Computer Algebra”

Computer algebra systems have to deal with the confusion between “programming variables” and “mathematical symbols”. We claim that they should also deal with “unknowns”, i.e. elements whose values are unknown, but whose type is known. For examples $x^p \neq x$ if $x$ is a symbol, but $x^p = x$ if $x \in GF(p)$. We show how we have extended Axiom to deal with this concept.

[Davenport 00] Davenport, James
“13th OpenMath Meeting” James H. Davenport “A New Algebra System” May 1984
xml.coverpages.org/openmath13.html

[Davenport 12] Davenport, J.H.
“Computer Algebra”
staff.bath.ac.uk/masjhd/JHD-CA.pdf

[Davenport (b)] Davenport, J. H.; Siret; Tournier
“Computer Algebra”
staff.bath.ac.uk/masjhd/masternew.pdf

[Dewar 94] Dewar, M. C.
“Manipulating Fortran Code in AXIOM and the AXIOM-NAG Link”
Proceedings of the Workshop on Symbolic and Numeric Computing, ed by Apiola, H.
and Laine, M. and Valkeila, E. pp1-12 University of Helsinki, Finland (1994)

[Dewar] Dewar, Mike
“OpenMath: An Overview”

[Dicrescenzo 89] Dicrescenzo, C.; Duval, D.
“Algebraic extensions and algebraic closure in Scratchpad II”
In Gianni [Gia89], pp440-446 ISBN 3-540-51084-2 LCCN QA76.95.I57 1998 Conference
held jointly with AAECC-6

[Dingle 94] Dingle, Adam; Fateman, Richard
“Branch Cuts in Computer Algebra”
1994 ISSAC, Oxford (UK), July 1994
www.cs.berkeley.edu/~fateman/papers/ding.ps
Many standard functions, such as the logarithms and square root functions, cannot be defined continuously on the complex plane. Mistaken assumptions about the properties of these functions lead computer algebra systems into various conundrums. We discuss how they can manipulate such functions in a useful fashion.

[DLMF].
“Digital Library of Mathematical Functions”
dlmf.nist.gov/software/#T1

[Dooley 99] Dooley, Sam editor.

[Dos Reis 12] Dos Reis, Gabriel
“A System for Axiomatic Programming”

We present the design and implementation of a system for axiomatic programming, and its application to mathematical software construction. Key novelties include a direct support for user-defined axioms establishing local equality between types, and overload resolution based on equational theories and user-defined local axioms. We illustrate uses of axioms, and their organization into concepts, in structured generic programming as practiced in computational mathematical systems.

[Doye 97] Doye, Nicolas James
“Order Sorted Computer Algebra and Coercions”
Ph.D. Thesis University of Bath 1997

Computer algebra systems are large collections of routines for solving mathematical problems algorithmically, efficiently and above all, symbolically. The more advanced and rigorous computer algebra systems (for example, Axiom) use the concept of strong types based on order-sorted algebra and category theory to ensure that operations are only applied to expressions when they “make sense”. In cases where Axiom uses notions which are not covered by current mathematics we shall present new mathematics which will allow us to prove that all such cases are reducible to cases covered by the current theory. On the other hand, we shall also point out all the cases where Axiom deviates undesirably from the mathematical ideal. Furthermore we shall propose solutions to these deviations. Strongly typed systems (especially of mathematics) become unusable unless the system can change the type in a way a user expects. We wish any change expected by a user to be automated, “natural”, and unique. “Coercions” are normally viewed as “natural type changing maps”. This thesis shall rigorously define the word “coercion” in the context of computer algebra systems.
1.1. AXIOM CITATIONS IN THE LITERATURE

We shall list some assumptions so that we may prove new results so that all coercions are unique. This concept is called “coherence”.

We shall give an algorithm for automatically creating all coercions in type system which adheres to a set of assumptions. We shall prove that this is an algorithm and that it always returns a coercion when one exists. Finally, we present a demonstration implementation of this automated coercion algorithm in Axiom.

[Doyle 99] Doyle, Nicolas J.
“Automated coercion for Axiom”
www.acm.org/pubs/contents/proceedings/issac/309831

[Dominguez 01] Domínguez, César; Rubio, Julio
“Modeling Inheritance as Coercion in a Symbolic Computation System” ISSAC 2001
ACM 1-58113-417-7/01/0007
In this paper the analysis of the data structures used in a symbolic computation system, called Kenzo, is undertaken. We deal with the specification of the inheritance relationship since Kenzo is an object-oriented system, written in CLOS, the Common Lisp Object System. We focus on a particular case, namely the relationship between simplicial sets and chain complexes, showing how the order-sorted algebraic specifications formalisms can be adapted, through the “inheritance as coercion” metaphor, in order to model this Kenzo fragment.

[Dunstan 97] Dunstan, Martin; Ursula, Martin; Linton, Steve
“Embedded Verification Techniques for Computer Algebra Systems”
Grant citation GR/L48256 Nov 1, 1997-Feb 28, 2001

[Dunstan 01] Dunstan, Martin; Gottliebsen, Hanne; Kelsey, Tom; Martin, Ursula
www.cs.st-andrews.ac.uk/~tom/pub/tphols.ps
www.cs.st-andrews.ac.uk/~tom/pub/dunstanetal.ps

[Duval 92] Duval D.; Jung, F.
“Examples of problem solving using computer algebra”
CODEN ITATEC. ISSN 0926-5473

[Duval 94] Duval, Dominique
“Symbolic or algebraic computation?”
Madrid Spain, NAG conference (private copy of paper)

[Duval 95] Duval, D.
“Évaluation dynamique et clôture algébrique en Axiom”.
BIBLIOGRAPHY

E

[Erocal 10] Eröcal, Burcin; Stein, William
“The Sage Project”
wstein.org/papers/icms/icms_2010.pdf

F

[Fateman 90] Fateman, R. J.
“Advances and trends in the design and construction of algebraic manipulation sys-
tems”
In Watanabe and Nagata [WN90], pp60-67 ISBN 0-89791-401-5 LCCN QA76.95.157 1990

[Fateman 05] Fateman, R. J.
“An incremental approach to building a mathematical expert out of software”
4/19/2005
www.cs.berkeley.edu/~fateman/papers/axiom.pdf

[Fateman 06] Fateman, R. J.
“Building Algebra Systems by Overloading Lisp”

Some of the earliest computer algebra systems (CAS) looked like overloaded
languages of the same era. FORMAC, PL/I FORMAC, Formula Algol, and others
each took advantage of a pre-existing language base and expanded the notion of a
numeric value to include mathematical expressions. Much more recently, perhaps
encouraged by the growth in popularity of C++, we have seen a renewal of the
use of overloading to implement a CAS.

This paper makes three points. 1. It is easy to do overloading in Common Lisp,
and show how to do it in detail. 2. Overloading per se provides an easy solu-
tion to some simple programming problems. We show how it can be used for a
“demonstration” CAS. Other simple and plausible overloadings interact nicely
with this basic system. 3. Not all goes so smoothly: we can view overloading as a
case study and perhaps an object lesson since it fails to solve a number of fairly-
well articulated and difficult design issues in CAS for which other approaches are
preferable.

[Faure 00a] Faure, Christèle; Davenport, James
“Parameters in Computer Algebra”

[Faure 00b] Faure, Christèle; Davenport, James; Naciri, Hanane
“Multi-values Computer Algebra”
ISSN 0249-6399 Institut National De Recherche en Informatique et en Automatique
Sept. 2000 No. 4001
1.1. AXIOM CITATIONS IN THE LITERATURE

[Fitch 84] Fitch, J. P. (ed) 

[Fitch 93] Fitch, J. (ed) 

[Fogus 11] Fogus, Michael 
“UnConj”
clojure.com/blog/2011/11/22/unconj.html

[Fortenbacher 90] Fortenbacher, A. 
“Efficient type inference and coercion in computer algebra”

[Fouche 90] Fouche, Francois 
“Une implantation de l’algorithme de Kovacic en Scratchpad”
Technical report, Institut de Recherche Mathématique Avancée” Strasbourg, France, 1990 31pp

[FSF 14] FSF 
“Free Software Directory”
directory.fsf.org/wiki/Axiom

[Frisco ] Frisco 
“Objectives and Results”
www.nag.co.uk/projects/frisco/frisco/node3.htm

G

[Gebauer 86] Gebauer, Rüdiger; Möller, H. Michael 
“Buchberger’s algorithm and staggered linear bases”

[Gebauer 88] Gebauer, R.; Möller, H. M. 
“On an installation of Buchberger’s algorithm”
Journal of Symbolic Computation, 6(2-3) pp275-286 1988 CODEN JSYCEH ISSN 0747-7171
[Geddes 92] Geddes, Keith; Czapor, O.; Stephen R.; Labahn, George
“Algorithms For Computer Algebra”

[Gianni 87] Gianni, Patrizia
“Primary Decomposition of Ideals”
in [Wit87], pp12-13

ISBN 3-540-51084-2 LCCN QA76.95.I57 1988 Conference held jointly with AAECC-6

“Algebraic solution of systems of polynomial equations using Gröbner bases.”

[Gil 92] Gil, I.
“Computation of the Jordan canonical form of a square matrix (using the Axiom programming language)”

[Gomez-Diaz 92] Gómez-Díaz, Teresa
“Quelques applications de l’évaluation dynamique” Ph.D. Thesis L’Universite De Limoges March 1992

[Gomez-Diaz 93] Gómez-Díaz, Teresa
“Examples of using Dynamic Constructible Closure” IMACS Symposium SC-1993

“Using MathCAD in teaching material and energy balance concepts”
In Anonymous [Ano91], pp345-349 (vol. 1) 2 vols.


[Gonnet 96] Gonnet, Gaston H.
“Official verion 1.0 of the Meta Content Dictionary”
www.inf.ethz.ch/personal/gonnet/ContDict/Meta

[Goodloe 93] Goodloe, A.; Loustaunau, P.

[Gottliebsen 05] Gottliebsen, Hanne; Kelsey, Tom; Martin, Ursula
“Hidden verification for computational mathematics”
1.1. AXIOM CITATIONS IN THE LITERATURE

“About the Polynomial System Solve Facility of Axiom, Macyma, Maple Mathematica, MuPAD, and Reduce”

We report on some experiences with the general purpose Computer Algebra Systems (CAS) Axiom, Macsyma, Maple, Mathematica, MuPAD, and Reduce solving systems of polynomial equations and the way they present their solutions. This snapshot (taken in the spring of 1996) of the current power of the different systems in a special area concentrates on both CPU-times and the quality of the output.

“Das ComputeralgebraSystem AXIOM bei kryptologischen und verkehrstheoretischen Untersuchungen des Forschungsinstituts der Deutschen Bundespost TELEKOM”
Technischer Report TR 75.91.20, IBM Wissenschaftliches Zentrum, Heidelberg, Germany, 1991

[Grabmeier 92] Grabmeier, J.; Scheerhorn, A.
“Finite fields in Axiom”
www.nag.co.uk/doc/TechRep/axiomtr.html
and Technical Report, IBM Heidelberg Scientific Center, 1992

[Grabmeier 03] Grabmeier, Johannes; Kaltofen, Erich; Weispfenning, Volker (eds)
www.springer.com/sgw/cda/frontpage/0,11855,1-102-22-1477871-0,00.html

“SCRATCHPAD/1 – an interactive facility for symbolic mathematics”
In Petrick [Pet71], pp42-58. LCCN QA76.5.S94 1971
delivery.acm.org/10.1145/810000/806266/p42-griesmer.pdf
SYMSAC’71 Proc. second ACM Symposium on Symbolic and Algebraic Manipulation pp45-48

The SCRATCHPAD/1 system is designed to provide an interactive symbolic computational facility for the mathematician user. The system features a user language designed to capture the style and succinctness of mathematical notation, together with a facility for conveniently introducing new notations into the language. A comprehensive system library incorporates symbolic capabilities provided by such systems as SIN, MATHLAB, and REDUCE.

[Griesmer 72a] Griesmer, J.; Jenks, R.
“Experience with an online symbolic math system SCRATCHPAD”
“SCRATCHPAD: A capsule view”
Two-dimensional man-machine communications. Mark B. Wells and James B. Morris
(eds.).

“SCRATCHPAD User’s Manual”
IBM Research Publication RA70 June 1975

“A Set of SCRATCHPAD Examples”
April 1976 (private copy)

“Introduction to Gauss”
SIGSAM Bulletin (ACM Special Interest Group on Symbolic and Algebraic Manipu-
lation), 28(3) pp3-19 August 1994 CODEN SIGSBZ ISSN 0163-5824

[Gruntz 96] Gruntz, Dominik
“On Computing Limits in a Symbolic Manipulation System”
Thesis, Swiss Federal Institute of Technology Zürich 1996 Diss. ETH No. 11432
www.cybertester.com/data/gruntz.pdf

This thesis presents an algorithm for computing (one-sided) limits within a sym-
boolic manipulation system. Computing limits is an important facility, as limits
are used both by other functions such as the definite integrator and to get directly
some qualitative information about a given function.

The algorithm we present is very compact, easy to understand and easy to imple-
ment. It overcomes the cancellation problem other algorithms suffer from. These
goals were achieved using a uniform method, namely by expanding the whole
function into a series in terms of its most rapidly varying subexpression instead
of a recursive bottom up expansion of the function. In the latter approach ex-
act error terms have to be kept with each approximation in order to resolve the
cancellation problem, and this may lead to an intermediate expression swell. Our
algorithm avoids this problem and is thus suited to be implemented in a symbolic
manipulation system.

H

[Hassner 87] Hassner, Martin; Burge, William H.; Watt, Stephen M.
“Construction of Algebraic Error Control Codes (ECC) on the Elliptic Riemann Sur-
face”
in [Wit87], pp5-8

[Heck 01] Heck, A.
“Variables in computer algebra, mathematics and science”
1.1. AXIOM CITATIONS IN THE LITERATURE

The International Journal of Computer Algebra in Mathematics Education Vol. 8 No. 3 pp195-210 (2001)


J

[Jacob 93] Jacob, G.; Oussous, N. E.; Steinberg, S. (eds)


[Jenks 69] Jenks, R. D.
“META/LISP: An interactive translator writing system”

[Jenks 71] Jenks, R. D.
“META/PLUS: The syntax extension facility for SCRATCHPAD”

[Jenks 74] Jenks, R. D.
“The SCRATCHPAD language”
ACM SIGPLAN Notices, 9(4) pp101-111 1974 CODEN SINODQ. ISSN 0362-1340

[Jenks 76] Jenks, Richard D.
“A pattern compiler”

[Jenks 79] Jenks, R. D.
“MODLISP: An Introduction”
Proc EUROSAM 79, pp466-480, 1979 and IBMRC8073 Jan 1980

[Jenks 81] Jenks, R.D.; Trager, B.M.
“A Language for Computational Algebra”
Proceedings of SYMSAC81, Symposium on Symbolic and Algebraic Manipulation, Snowbird, Utah August, 1981
[Jenks 81a] Jenks, R.D.; Trager, B.M.
“A Language for Computational Algebra”
SIGPLAN Notices, New York: Association for Computing Machinery, Nov 1981

[Jenks 81b] Jenks, R.D.; Trager, B.M.
“A Language for Computational Algebra”
IBM Research Report RC8930 IBM Yorktown Heights, NY

[Jenks 84a] Jenks, Richard D.
“The new SCRATCHPAD language and system for computer algebra”
In Golden and Hussain [GH84], pp409-??

[Jenks 84b] Jenks, Richard D.
“A primer: 11 keys to New Scratchpad”

“Scratchpad II: An Abstract Datatype System for Mathematical Computation”
Research Report RC 12327 (#55257), International Business Machines, Inc.,
Thomas J. Watson Research Center, Yorktown Heights, NY, USA, 1986 23pp

Scratchpad II is an abstract datatype language and system that is under develop-
ment in the Computer Algebra Group, Mathematical Sciences Department, at
the IBM Thomas J. Watson Research Center. Some features of APL that made
computation particularly elegant have been borrowed. Many different kinds of
computational objects and data structures are provided. Facilities for computa-
tion include symbolic integration, differentiation, factorization, solution of equa-
tions and linear algebra. Code economy and modularity is achieved by having
polymorphic packages of functions that may create datatypes. The use of cate-
gories makes these facilities as general as possible.

“Scratchpad II: an Abstract Datatype System for Mathematical Computation”
Proceedings Trends in Computer Algebra, Bad Neuenahr, LNCS 296, Springer Verlag,
(1987)

[Jenks 88] Jenks, R. D.; Sutor, R. S.; Watt, S. M.
“Scratchpad II: An abstract datatype system for mathematical computation”
In Janßen [Jan88], pp12-?? ISBN 3-540-18928-9, 0-387-18928-9 LCCN QA155.7.E4T74
1988

[Jenks 88a] Jenks, R. D.
“A Guide to Programming in BOOT”
Computer Algebra Group, Mathematical Sciences Department, IBM Research Draft
September 5, 1988

[Jenks 88b] Jenks, Richard
“The Scratchpad II Computer Algebra System Interactive Environment Users Guide”
Spring 1988
1.1. AXIOM CITATIONS IN THE LITERATURE

[Jenks 88c] Jenks, R. D.; Sutor, R. S.; Watt, S. M.
“Scratchpad II: an abstract datatype system for mathematical computation”

“AXIOM: The Scientific Computation System”
Springer-Verlag, Berlin, Germany / Heidelberg, Germany / London, UK / etc., 1992

[Jenks 94] Jenks, R. D.; Trager, B. M.
“How to make AXIOM into a Scratchpad”
In ACM [ACM94], pp32-40 ISBN 0-89791-638-7 LCCN QA76.95.159 1994

Joswig 03 Joswig, Michael; Takayama, Nobuki
“Algebra, geometry, and software systems”
Springer-Verlag ISBN 3-540-00256-1 p291

[Joyner 06] Joyner, David
“OSCAS - Maxima”
SIGSAM Communications in Computer Algebra, 157 2006
sage.math.washington.edu/home/wdj/sigsam/oscas-cca1.pdf

[Joyner 14] Joyner, David
“Links to some open source mathematical programs”
www.opensourcemath.org/opensource_math.html

K

[Kauers 08] Kauers, Manuel
“Integration of Algebraic Functions: A Simple Heuristic for Finding the Logarithmic Part”

“Production of Argument SubPrograms in the AXIOM – NAG link: examples involving nonlinear systems”
www.nag.co.uk/doc/TechRep/axiomtr.html

[Kelsey 99] Kelsey, Tom
Ph.D. Thesis, University of St Andrews, 1999
www.cs.st-andrews.ac.uk/research/publications/Kel00.php
We investigate the use of formal methods languages and tools in the design and development of computer algebra systems (henceforth CAS). We demonstrate that errors in CAS design can be identified and corrected by the use of (i) abstract specifications of types and procedures, (ii) automated proofs of properties of the specifications, and (iii) interface specifications which assist the verification of pre- and post conditions of implemented code.

Kelsey, Tom
“Formal specification of computer algebra”
University of St Andrews, 6th April 2000
www.cs.st-andrews.cs.uk/~tom/pub/fscbs.ps

Kelsey, Tom
“Formal specification of computer algebra”
(slides) University of St Andrews, Sept 21, 2000
www.cs.st-andrews.cs.uk/~tom/pub/fscbstalk.ps

Kendall, W.S.
“Itovsn3 in AXIOM: modules, algebras and stochastic differentials”
www2.warwick.ac.uk/fac/sci/statistics/staff/academic-research/kendall/personal/ppt/328.ps.gz

Kendall, W.S.
“Symbolic Itô calculus in AXIOM: an ongoing story”
www2.warwick.ac.uk/fac/sci/statistics/staff/academic-research/kendall/personal/ppt/327.ps.gz

P.-V. Koseleff
“Word games in free Lie algebras: several bases and formulas”
Theoretical Computer Science 79(1) pp241-256 Feb. 1991 CODEN TCSCDI ISSN 0304-3975

Kusche, K.; Kutzler, B.; Mayr, H.
“Implementation of a geometry theorem proving package in SCRATCHPAD II”

Lahey, Tim
“Sage Integration Testing”
github.com/tjl/sage_int_testing Dec. 2008

Lambe, L. A.
“Scratchpad II as a tool for mathematical research”
Notices of the AMS, February 1928 pp143-147
One way in which mathematicians deal with infinite amounts of data is symbolic representation. A simple example is the quadratic equation

\[ x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \]

a formula which uses symbolic representation to describe the solutions to an infinite class of equations. Most computer algebra systems can deal with polynomials with symbolic coefficients, but what if symbolic exponents are called for (e.g. \(1 + t^i\))? What if symbolic limits on summations are also called for, for example

\[ 1 + t + \ldots + t^i = \sum_j t^j \]

The “Scratchpad Concept” is a theoretical ideal which allows the implementation of objects at this level of abstraction and beyond in a mathematically consistent way. The Axiom computer algebra system is an implementation of a major part of the Scratchpad Concept. Axiom (formerly called Scratchpad) is a language with extensible parameterized types and generic operators which is based on the notions of domains and categories. By examining some aspects of the Axiom system, the Scratchpad Concept will be illustrated. It will be shown how some complex problems in homological algebra were solved through the use of this system.
[Lebedev 08] Lebedev, Yuri
“OpenMath Library for Computing on Riemann Surfaces”
PhD thesis, Nov 2008 Florida State University
www.math.fsu.edu/~ylebedev/research/HyperbolicGeometry.html

[LeBlanc 91] LeBlanc, S.E.
“The use of MathCAD and Theorist in the ChE classroom”
In Anonymous [Ano91], pp287-299 (vol. 1) 2 vols.

[Lecerf 96] Lecerf, Grégoire
“Dynamic Evaluation and Real Closure Implementation in Axiom”
June 29, 1996
www.math.uvsq.fr/~lecerf/software/drc/drc.ps

[Lecerf 96a] Lecerf, Grégoire
“The Dynamic Real Closure implemented in Axiom”
lecerf.perso.math.cnrs.fr/software/drc/drc.ps

[Levelt 95] Levelt, A. H. M. (ed)

[Li 06] Li, Xin; Maza, Moreno
“Efficient Implementation of Polynomial Arithmetic in a Multiple-Level Programming Environment”

[Li 10] Li, Yue; Dos Reis, Gabriel
“A Quantitative Study of Reductions in Algebraic Libraries”

[Li 11] Li, Yue; Dos Reis, Gabriel
“An Automatic Parallelization Framework for Algebraic Computation Systems”

This paper proposes a non-intrusive automatic parallelization framework for typeful and property-aware computer algebra systems.

[Linton 93] Linton, Steve
“Vector Enumeration Programs, version 3.04”
www.cs.st-andrews.ac.uk/~sal/nme/nme_toc.html#SEC1

[Liska 97] Liska, Richard; Drska, Ladislav; Limpouch, Jiri; Sinor, Milan; Wester, Michael; Winkler, Franz
“Computer Algebra - algorithms, systems and applications”
1.1. AXIOM CITATIONS IN THE LITERATURE

June 2, 1997
kfe.fjfi.cvut.cz/~liska/ca/all.html

[Lucks 86] Lucks, Michael
“A fast implementation of polynomial factorization”

[Lueken 77] Lueken, E.
”Ueberlegungen zur Implementierung eines Formelmanipulationssystems”
Master’s thesis, Technischen Universität Carolo-Wilhelmina zu Braunschweig, Braunschweig, Germany, 1977

[Lynch 91] Lynch, R.; Mavromatis, H. A.
“New quantum mechanical perturbation technique using an ’electronic scratchpad’ on an inexpensive computer”
American Journal of Physics, 59(3) pp270-273, March 1991. CODEN AJPIAS ISSN 0002-9505

M

[Mahboubi 05] Mahboubi, Assia
“Programming and certifying the CAD algorithm inside the coq system”

[Mathews 89] Mathews, J.
“Symbolic computational algebra applied to Picard iteration”
Mathematics and computer education, 23(2) pp117-122 Spring 1989 CODEN MCEDDA, ISSN 0730-8639

“Software Presentation Group – Common Lisp family”
www.softwarepreservation.org/projects/LISP/common_lisp_family

[Melachrinoudis 90] Melachrinoudis, E.; Rumpf, D. L.
“Teaching advantages of transparent computer software – MathCAD”
CoED, 10(1) pp71-76, January-March 1990 CODEN CWLJDP ISSN 0736-8607

“Design and Implementation of Symbolic Computation Systems”
“Design and Implementation of Symbolic Computation Systems”
International Symposium DISCO '93 Gmunden, Austria, September 15-17, 1993: Pro-

[Missura 94] Missura, Stephan A.; Weber, Andreas
“Using Commutativity Properties for Controlling Coercions”
cg.cs.uni-bonn.de/personal-pages/weber/publications/pdf/
WeberA/MissuraWeber94a.pdf

This paper investigates some soundness conditions which have to be fulfilled in systems with coercions and generic operators. A result of Reynolds on unrestricted generic operators is extended to generic operators which obey certain constraints. We get natural conditions for such operators, which are expressed within the theoretic framework of category theory. However, in the context of computer algebra, there arise examples of coercions and generic operators which do not fulfill these conditions. We describe a framework – relaxing the above conditions – that allows distinguishing between cases of ambiguities which can be resolved in a quite natural sense and those which cannot. An algorithm is presented that detects such unresolvable ambiguities in expressions.

[Monagan 87] Monagan, Michael B.
“Support for Data Structures in Scratchpad II”
in [Wit87], pp17-18

[Monagan 93] Monagan, M. B.
“Gauss: a parameterized domain of computation system with support for signature functions”


[Moses 71] Moses, Joel
“Algebraic Simplification: A Guide for the Perplexed”
CACM August 1971 Vol 14 No. 8 pp527-537

[N ]

[Naylor] Naylor, William; Padget, Julian
“From Untyped to Polymorphically Typed Objects in Mathematical Web Services”
OpenMath is a widely recognized approach to the semantic markup of mathematics that is often used for communication between OpenMath compliant systems. The Aldor language has a sophisticated category-based type system that was specifically developed for the purpose of modelling mathematical structures, while the system itself supports the creation of small-footprint applications suitable for deployment as web services. In this paper we present our first results of how one may perform translations from generic OpenMath objects into values in specific Aldor domains, describing how the Aldor interface domain Expression-Tree is used to achieve this. We outline our Aldor implementation of an OpenMath translator, and describe an efficient extension of this to the Parser category. In addition, the Aldor service creation and invocation mechanism are explained. Thus we are in a position to develop and deploy mathematical web services whose descriptions may be directly derived from Aldor’s rich type language.

[Naylor 95] Naylor, Bill
“Symbolic Interface for an advanced hyperbolic PDE solver”
www.sci.csd.uwo.ca/~bill/Papers/symbInterface2.ps

An Axiom front end is described, which is used to generate mathematical objects needed by one of the latest NAG routines, to be included in the Mark 17 version of the NAG Numerical library. This routine uses powerful techniques to find the solution to Hyperbolic Partial Differential Equations in conservation form and in one spatial dimension. These mathematical objects are non-trivial, requiring much mathematical knowledge on the part of the user, which is otherwise irrelevant to the physical problem which is to be solved. We discuss the individual mathematical objects, considering the mathematical theory which is relevant, and some of the problems which have been encountered and solved during the FORTRAN generation necessary to realise the object. Finally we display some of our results.

[Naylor 00b] Naylor, W.A.; Davenport, J.H.
“A Monte-Carlo Extension to a Category-Based Type System”
www.sci.csd.uwo.ca/~bill/Papers/monteCarCat3.ps

The normal claim for mathematics is that all calculations are 100% accurate and therefore one calculation can rely completely on the results of sub-calculations, however there exist Monte-Carlo algorithms which are often much faster than the equivalent deterministic ones where the results will have a prescribed probability (presumably small) of being incorrect. However there has been little discussion of how such algorithms can be used as building blocks in Computer Algebra. In this paper we describe how the computational category theory which is the basis of the type structure used in the Axiom computer algebra system may be extended to cover probabilistic algorithms, which use Monte-Carlo techniques. We follow this with a specific example which uses Straight Line Program representation.

[Norman 75] Norman, A. C.
“Computing with formal power series”

[Norman 75a] Norman, A.C.
 “The SCRATCHPAD Power Series Package”
 IBM T.J. Watson Research RC4998

[Ollivier 89] Ollivier, F.
 “Inversibility of rational mappings and structural identifiability in automatics”

[Online 72] .

[OpenMath] .
 “OpenMath Technical Overview”
 www.openmath.org/overview/technical.html

[P] Page, William S.
 “Axiom - Open Source Computer Algebra System”
 Poster ISSAC 2007 Proceedings Vol 41 No 3 Sept 2007 p114

[Petitot 90] Petitot, Michel
 “Types récursifs en scratchpad, application aux polynômes non commutatifs”
 LIFL, 1990

[Petitot 93] Petitot, M.
 “Experience with Axiom”
 In Jacob et al. [JOS93], page 240

[Petric 71] Petric, S. R. (ed)

[Pinch 93] Pinch, R.G.E.
 “Some Primality Testing Algorithms”
 Devlin, Keith (ed.) Computers and Mathematics November 1993, Vol 40, Number 9 pp1203-1210

[Poll (b)] Poll, Erik
 “The type system of Axiom”
1.1. AXIOM CITATIONS IN THE LITERATURE

[Purtilo 86] Purtilo, J.

[R]
[Rainer 14] Joswig, Rainer
"2014: 30+ Years Common Lisp the Language"
lispm.de/30ycltl

[Robidoux 93] Robidoux, Nicolas
"Does Axiom Solve Systems of O.D.E’s Like Mathematica?"
July 1993
If I were demonstrating Axiom and were asked this question, my reply would be “No, but I am not sure that this is a bad thing”. And I would illustrate this with the following example.
Consider the following system of O.D.E.’s
\[
\frac{dx_1}{dt} = \left(1 + \frac{\cos}{2 + \sin}\right)x_1 \\
\frac{dx_2}{dt} = x_1 - x_2
\]
This is a very simple system: $x_1$ is actually uncoupled from $x_2$

[Rioboo 92] Rioboo, R.
"Real algebraic closure of an ordered field, implementation in Axiom”
Real algebraic numbers appear in many Computer Algebra problems. For instance the determination of a cylindrical algebraic decomposition for a euclidean space requires computing with real algebraic numbers. This paper describes an implementation for computations with the real roots of a polynomial. This process is designed to be recursively used, so the resulting domain of computation is the set of all real algebraic numbers. An implementation for the real algebraic closure has been done in Axiom (previously called Scratchpad).

[Roesner 95] Roesner, K. G.
"Verified solutions for parameters of an exact solution for non-Newtonian liquids using computer algebra” Zeitschrift fur Angewandte Mathematik und Physik, 75 (suppl. 2):S435-S438, 1995 ISSN 0044-2267
S

[Sage 14] Stein, William
“Sage”
www.sagemath.org/doc/reference/interfaces/sage/interfaces/axiom.html

[Salvy 89] Salvy, B.
“Examples of automatic asymptotic expansions”

[Salvy 91] Salvy, B.
“Examples of automatic asymptotic expansions”
SIGSAM Bulletin (ACM Special Interest Group on Symbolic and Algebraic Manipulation), 25(2) pp4-17 April 1991 CODEN SIGSBZ ISSN 0163-5824

[Schu 92] Schü, J.
“Implementing des Cartan-Kuranishi-Theorems in AXIOM”
Master’s diploma thesis (in german), Institut für Algorithmen und Kognitive Systeme, Universität Karlsruhe 1992

[Schwarz 88] Schwarz, F.
“Programming with abstract data types: the symmetry package SPDE in Scratchpad”

[Schwarz 89] Schwarz, F.
“A factorization algorithm for linear ordinary differential equations”

[Schwarz 91] Schwarz, F.
“Monomial orderings and Gröbner bases”
SIGSAM Bulletin (ACM Special Interest Group on Symbolic and Algebraic Manipulation) 25(1) pp10-23 Jan. 1991 CODEN SIGSBZ ISSN 0163-5824

[Seiler 94] Seiler, Werner Markus
“Analysis and Application of the Formal Theory of Partial Differential Equations”
www.mathematik.uni-kassel.de/~seiler/Papers/Diss/diss.ps.gz

An introduction to the formal theory of partial differential equations is given emphasizing the properties of involutive symbols and equations. An algorithm to complete any differential equation to an involutive one is presented. For an involutive equation possible values for the number of arbitrary functions in its general solution are determined. The existence and uniqueness of solutions for analytic equations is proven. Applications of these results include an analysis of symmetry and reduction methods and a study of gauge systems. It is show that the Dirac algorithm for systems with constraints is closely related to the
1.1. AXIOM CITATIONS IN THE LITERATURE

completion of the equation of motion to an involutive equation. Specific examples treated comprise the Yang-Mills Equations, Einstein Equations, complete and Jacobian systems, and some special models in two and three dimensions. To facilitate the involved tedious computations an environment for geometric approaches to differential equations has been developed in the computer algebra system Axiom. The appendices contain among others brief introductions into Carten-Kähler Theory and Janet-Riquier Theory.

[Seiler 94a] Seiler, W.M.
“Completion to involution in AXIOM”
in Calmet [Cal94] pp103-104

[Seiler 94b] Seiler, W.M.
“Pseudo differential operators and integrable systems in AXIOM”
Computer Physics Communications, 79(2) pp329-340 April 1994 CODEN CPHCBZ ISSN 0010-4655
An implementation of the algebra of pseudo differential operators in the computer algebra system Axiom is described. In several examples the application of the package to typical computations in the theory of integrable systems is demonstrated.

[Seiler 95] Seiler, W.M.
“Applying AXIOM to partial differential equations”
Internal Report 95-17, Universität Karlsruhe, Fakultät für Informatik 1995
We present an Axiom environment called JET for geometric computations with partial differential equations within the framework of the jet bundle formalism. This comprises especially the completion of a given differential equation to an involutive one according to the Cartan-Kuranishi Theorem and the setting up of the determining system for the generators of classical and non-classical Lie symmetries. Details of the implementations are described and applications are given. An appendix contains tables of all exported functions.

[Seiler 95b] Seiler, W.M.; Calmet, J.
“JET – An Axiom Environment for Geometric Computations with Differential Equations”
JET is an environment within the computer algebra system Axiom to perform such computations. The current implementation emphasises the two key concepts involution and symmetry. It provides some packages for the completion of a given system of differential equations to an equivalent involutive one based on the Cartan-Kuranishi theorem and for setting up the determining equations for classical and non-classical point symmetries.

[Seiler 97] Seiler, Werner M.
“Computer Algebra and Differential Equations: An Overview”
www.mathematik.uni-kassel.de/~seiler/Papers/Postscript/CADERep.ps.gz
We present an informal overview of a number of approaches to differential equations which are popular in computer algebra. This includes symmetry and completion theory, local analysis, differential ideal and Galois theory, dynamical systems and numerical analysis. A large bibliography is provided.

[Seiler (a)] Seiler, W.M.
“DETools: A Library for Differential Equations”
iaks-www.ira.uka.de/iaks-calmet/werner/werner.html

[Shannon 88] Shannon, D.; Sweedler, M.
“Using Gröbner bases to determine algebra membership, split surjective algebra homomorphisms determine birational equivalence”

[Sit 89] Sit, W.Y.
“On Goldman’s algorithm for solving first-order multinomial autonomous systems”
In Mora [Mor89], pp386-395 ISBN 3-540-51083-4 LCCN QA268.A35 1998 Conference held jointly with ISSAC ’88

[Sit 92] Sit, W.Y.
“An algorithm for solving parametric linear systems”
Journal of Symbolic Computations, 13(4) pp353-394, April 1992 CODEN JSYCEH ISSN 0747-7171

[Sit 06] Sit, Emil
“Tools for Repeatable Research”
www.emilsit.net/blog/archives/tools-for-repeatable-research

[Smedley 92] Smedley, Trevor J.
“Using pictorial and object oriented programming for computer algebra”

[Smith 07] Smith, Jacob; Dos Reis, Gabriel; Jarvi, Jaakko
“Algorithmic differentiation in Axiom”

This paper describes the design and implementation of an algorithmic differentiation framework in the Axiom computer algebra system. Our implementation works by transformations on Spad programs at the level of the typed abstract syntax tree.

[SSC92] .
“Algorithmic Methods For Lie Pseudogroups” In N. Ibragimov, M. Torrisi and A.
1.1. AXIOM CITATIONS IN THE LITERATURE


iaks-www.ira.uka.de/iaks-calmet/werner/Papers/Acireale92.ps.gz

[SSV87] Senechaud, P.; Siebert, F.; Villard G.
“Scratchpad II: Présentation d’un nouveau langage de calcul formel”

[Steele] Steele, Guy L.; Gabriel, Richard P.
“The Evolution of Lisp”
www.dreamsongs.com/Files/HOPL2-Uncut.pdf

[Sutor 85] Sutor, R.S.
“The Scratchpad II computer algebra language and system”

[Sutor 87a] Sutor, R. S.; Jenks, R. D.
“The type inference and coercion facilities in the Scratchpad II interpreter” In Wexelblat [Wex87], pp56-63 ISBN 0-89791-235-7 LCCN QA76.7.S54 v22 n7 SIGPLAN Notices, v22 n7 (July 1987)

[Sutor 87b] Sutor, Robert S.
IBM Course presentation slide deck Spring 1987

[Sutor 87c] Sutor, Robert S.; Jenks, Richard
“The type inference and coercion facilities in the Scratchpad II interpreter”
Research report RC 12595 (#56575), IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA, 1987, 11pp

The Scratchpad II system is an abstract datatype programming language, a compiler for the language, a library of packages of polymorphic functions and parameterized abstract datatypes, and an interpreter that provides sophisticated type inference and coercion facilities. Although originally designed for the implementation of symbolic mathematical algorithms, Scratchpad II is a general purpose programming language. This paper discusses aspects of the implementation of the interpreter and how it attempts to provide a user friendly and relatively weakly typed front end for the strongly typed programming language.

[Sutor 88] Sutor, Robert S.
“A guide to programming in the scratchpad 2 interpreter”
IBM Manual, March 1988
[Thompson 00] Thompson, Simon

“Logic and dependent types in the Aldor Computer Algebra System”

We show how the Aldor type system can represent propositions of first-order logic, by means of the ‘propositions as types’ correspondence. The representation relies on type casts (using pretend) but can be viewed as a prototype implementation of a modified type system with type evaluation reported elsewhere. The logic is used to provide an axiomatisation of a number of familiar Aldor categories as well as a type of vectors.

[Thompson (a)] Thompson, Simon; Timochouk, Leonid

“The Aldor language”

This paper introduces the Aldor-- language, which is a functional programming language with dependent types and a powerful, type-based, overloading mechanism. The language is built on a subset of Aldor, the ‘library compiler’ language for the Axiom computer algebra system. Aldor-- is designed with the intention of incorporating logical reasoning into computer algebra computations.

The paper contains a formal account of the semantics and type system of Aldor--; a general discussion of overloading and how the overloading in Aldor-- fits into the general scheme; examples of logic within Aldor-- and notes on the implementation of the system.

[Touratier 98] Touratier, Emmanuel

“Etude du typage dans le système de calcul scientifique Aldor”

Université de Limoges 1998

[van der Hoeven 14] van der Hoeven, Joris

“Computer algebra systems and TeXmacs”

www.texmacs.org/tmweb/plugins/cas.en.html

[van Hoeij 94] van Hoeij, M.

“An algorithm for computing an integral basis in an algebraic function field”


[Vasconcelos 99] Vasconcelos, Wolmer

“Computational Methods in Commutative Algebra and Algebraic Geometry”

1.1. AXIOM CITATIONS IN THE LITERATURE

W

[Wang 89] Wang, D.
“A program for computing the Liapunov functions and Liapunov constants in Scratchpad II”
SIGSAM Bulletin (ACM Special Interest Group on Symbolic and Algebraic Manipulation), 23(4) pp25-31, Oct. 1989, CODEN SIGSBZ ISSN 0163-5824

[Wang 91] Wang, Dongming
“Mechanical manipulation for a class of differential systems”


[Watanabe 90] Watanabe, Shunro; Nagata, Morio; (ed)

[Watt 85] Watt, Stephen
“Bounded Parallelism in Computer Algebra”
PhD Thesis, University of Waterloo

[Watt 86] Watt, S.M.; Della Dora, J.
“Algebra Snapshot: Linear Ordinary Differential Operators”
Scratchpad II Newsletter: Vol 1 Num 2 (Jan 1986)

[Watt 87] Watt, Stephen
“Domains and Subdomains in Scratchpad II”
in [Wit87], pp3-5

“Mapping as First Class Objects”
in [Wit87], pp13-17

[Watt 89] Watt, S. M.
“A fixed point method for power series computation”
In Gianni [Gia89], pp206-217 ISBN 3-540-51084-2 LCCN QA76.95.I57 1988 Conference held jointly with AAECC-6

“The Scratchpad II type system: Domains and subdomains”

“A# User’s Guide”
Version 1.0.0 O(ε^1) June 8, 1994

“A First Report on the A# Compiler (including benchmarks)”
IBM Research Report RC19529 (85075) May 12, 1994

[Watt 94c] Watt, Stephen M.
“A# Language Reference Version 0.35”

“AXIOM Library Compiler Users Guide”
The Numerical Algorithms Group (NAG) Ltd, 1994

[Watt 01] Watt, Stephen M.; Broadbery, Peter A.; Iglio, Pietro; Morrison, Scott C.; Steinbach, Jonathan M.
“FOAM: A First Order Abstract Machine Version 0.35”
IBM T. J. Watson Research Center (2001)

“Type Systems for Computer Algebra”
An important feature of modern computer algebra systems is the support of a rich type system with the possibility of type inference. Basic features of such a type system are polymorphism and coercion between types. Recently the use of order-sorted rewrite systems was proposed as a general framework. We will give a quite simple example of a family of types arising in computer algebra whose coercion relations cannot be captured by a finite set of first-order rewrite rules.

“Structuring the Type System of a Computer Algebra System”
Most existing computer algebra systems are pure symbol manipulating systems without language support for the occurring types. This is mainly due to the fact that the occurring types are much more complicated than in traditional programming languages. In the last decade the study of type systems has become an active area of research. We will give a proposal for a type system showing that several problems for a type system of a symbolic computation system can
be solved by using results of this research. We will also provide a variety of examples which will show some of the problems that remain and that will require further research.

[Weber 93b] Weber, Andreas
“Type Systems for Computer Algebra”
\texttt{cg.cs.uni-bonn.de/personal-pages/weber/publications/pdf/WeberA/Weber93b.pdf}

We study type systems for computer algebra systems, which frequently correspond to the “pragmatically developed” typing constructs used in AXIOM. A central concept is that of type classes which correspond to AXIOM categories. We will show that types can be syntactically described as terms of a regular order-sorted signature if no type parameters are allowed. Using results obtained for the functional programming language Haskell we will show that the problem of type inference is decidable. This result still holds if higher-order functions are present and parametric polymorphism is used. These additional typing constructs are useful for further extensions of existing computer algebra systems: These typing concepts can be used to implement category theoretic constructs and there are many well known constructive interactions between category theory and algebra.

“Algorithms for Type Inference with Coercions”
ISSAC 94 ACM 0-89791-638-7/94/0007

This paper presents algorithms that perform a type inference for a type system occurring in the context of computer algebra. The type system permits various classes of coercions between types and the algorithms are complete for the precisely defined system, which can be seen as a formal description of an important subset of the type system supported by the computer algebra program Axiom.

Previously only algorithms for much more restricted cases of coercions have been described or the frameworks used have been so general that the corresponding type inference problems were known to be undecidable.

“On coherence in computer algebra”
\texttt{cg.cs.uni-bonn.de/personal-pages/weber/publications/pdf/WeberA/Weber94e.pdf}

Modern computer algebra systems (e.g. AXIOM) support a rich type system including parameterized data types and the possibility of implicit coercions between types. In such a type system it will be frequently the case that there are different ways of building coercions between types. An important requirement is that all coercions between two types coincide, a property which is called coherence. We will prove a coherence theorem for a formal type system having several possibilities of coercions covering many important examples. Moreover, we will give some informal reasoning why the formally defined restrictions can be satisfied by an actual system.
“Computing Radical Expressions for Roots of Unity”

We present an improvement of an algorithm given by Gauss to compute a radical expression for a $p$-th root of unity. The time complexity of the algorithm is $O(p^3m^6logp)$, where $m$ is the largest prime factor of $p - 1$.

“Solving Cyclotomic Polynomials by Radical Expressions”
cg.cs.uni-bonn.de/personal-pages/weber/publications/pdf/

We describe a Maple package that allows the solution of cyclotomic polynomials by radical expressions. We provide a function that is an extension of the Maple solve command. The major algorithmic ingredient of the package is an improvement of a method due to Gauss which gives radical expressions for roots of unity. We will give a summary for computations up to degree 100, which could be done within a few hours of cpu time on a standard workstation.

[Wei-Jiang 12] Wei-Jiang
“Top free algebra System”
wei-jiang.com/it/software/top-free-algebra-system-bye-mathematica-bye-maple

[Wester 99] Wester, Michael J.
“Computer Algebra Systems”


[Wityak 87] Wityak, Sandra
“Scratchpad II Newsletter”
Volume 2, Number 1, Nov 1987

[WWW1] .
Software Preservation Group
www.softwarepresentation.org/projects/LISP/common_lisp_family

Y

[Yap 00] Yap, Chee Keng
“Fundamental Problems of Algorithmic Algebra”
1.1. AXIOM CITATIONS IN THE LITERATURE

[Yapp 07] Yapp, Clifford; Hebisch, Waldek; Kaminski, Kai
“Literate Programming Tools Implemented in ANSI Common Lisp”
brlcad.org/~starseeker/cl-web-v0.8.lisp.pamphlet

[Yun 83] Yun, David Y.Y.
“Computer Algebra and Complex Analysis”
Computational Aspects of Complex Analysis pp379-393 D. Reidel Publishing Company
H. Werner et. al. (eds.)

[Z] Z

[Zen92] Zenger, Ch.
“Gröbnerbasen für Differentialformen und ihre Implementierung in AXIOM”
Diplomarbeit, Universität Karlsruhe, Karlsruhe, Germany, 1992

[Zip92] Zippel, Richard
“Algebraic Computation”

[Zwi92] Zwillinger, Daniel
“Handbook of Integration”
1.2 Axiom Citations of External Sources

A

[Ablamowicz 98] Ablamowicz, Rafal
“Spinor Representations of Clifford Algebras: A Symbolic Approach”

[Abramowitz 64] Abramowitz, Milton; Stegun, Irene A.
“Handbook of Mathematical Functions”

[Abramowitz 68] Abramowitz M; Stegun I A
“Handbook of Mathematical Functions”
Dover Publications. (1968)

[Altmann 05] Altmann, Simon L.
“Rotations, Quaternions, and Double Groups”

[Ames 77] Ames W F
“Nonlinear Partial Differential Equations in Engineering”

[Amos 86] Amos D E
“Algorithm 644: A Portable Package for Bessel Functions of a Complex Argument and
Nonnegative Order”

[Anderson 00] Anderson, Edward
“Discontinuous Plane Rotations and the Symmetric Eigenvalue Problem”
LAPACK Working Note 150, University of Tennessee, UT-CS-00-454, December 4,
2000.

[Anthony 82] Anthony G T; Cox M G; Hayes J G
“DASL - Data Approximation Subroutine Library”
National Physical Laboratory. (1982)

[Aubry 99] P. Aubry; D. Lazard; M. Moreno Maza
“On the Theories of Triangular Sets”
Journal of Symbolic Computation 1999 Vol 28 pp105-124

B

[Bailey 66] Bailey P B
“Sturm-Liouville Eigenvalues via a Phase Function”
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Baker 96] Baker, George A.; Graves-Morris, Peter
“Pade Approximants”

[Baker 10] Baker, Martin
“3D World Simulation”
www.euclideanspace.com

[Baker 14] Baker, Martin
“Axiom Architecture”
www.euclideanspace.com/prog/scratchpad/internals/ccode

[Banks 68] Banks D O; Kurowski I
“Computation of Eigenvalues of Singular Sturm-Liouville Systems”

[Bard 74] Bard Y
“Nonlinear Parameter Estimation”
Academic Press. 1974

[Barrodale 73] Barrodale I; Roberts F D K
“An Improved Algorithm for Discrete $ll_1$ Linear Approximation”

[Barrodale 74] Barrodale I; Roberts F D K
“Solution of an Overdetermined System of Equations in the $ll_1 − norm.$”

[Beauzamy 92] Beauzamy, Bernard
“Products of polynomials and a priori estimates for coefficients in polynomial decom-
positions: a sharp result”
J. Symbolic Computation (1992) 13, 463-472

[Beauzamy 93] Beauzamy, Bernard; Trevisan, Vilmar; Wang, Paul S.
“Polynomial Factorization: Sharp Bounds, Efficient Algorithms”
J. Symbolic Computation (1993) 15, 393-413

[Bertrand 95] Bertrand, Laurent
“Computing a hyperelliptic integral using arithmetic in the jacobian of the curve”

[Berzins 87] Berzins M; Brankin R W; Gladwell I.
“Design of the Stiff Integrators in the NAG Library”

[Berzins 90] Berzins M
“Developments in the NAG Library Software for Parabolic Equations”
BIBLIOGRAPHY

[Birkhoff 62] Birkhoff, G; Rota, G C
“Ordinary Differential Equations”

[Boyd93a] Boyd, David W.
“Bounds for the Height of a Factor of a Polynomial in Terms of Bombieri’s Norms: I. 
The Largest Factor”
J. Symbolic Computation (1993) 16, 115-130

[Boyd93b] Boyd, David W.
“Bounds for the Height of a Factor of a Polynomial in Terms of Bombieri’s Norms: II. 
The Smallest Factor”
J. Symbolic Computation (1993) 16, 131-145

[Braman 02a] Braman, K.; Byers, R.; Mathias, R.
“The Multi-Shift QR Algorithm Part I: Maintaining Well Focused Shifts, and Level 3 
Performance”

[Braman 02b] Braman, K.; Byers, R.; Mathias, R.
“The Multi-Shift QR Algorithm Part II: Aggressive Early Deflation”

[Brent 75] Brent, R. P.
“Multiple-Precision Zero-Finding Methods and the Complexity of Elementary Function 
Evaluation, Analytic Computational Complexity”

[Brent 78] Brent, R. P.; Kung, H. T.
“Fast Algorithms for Manipulating Formal Power Series”
Journal of the Association for Computing Machinery, Vol. 25, No. 4, October 1978, 
581-595

[Brigham 73] Brigham E O
“The Fast Fourier Transform”
Prentice-Hall. (1973)

[Brillhart 69] Brillhart, John
“On the Euler and Bernoulli polynomials”

[Brillhart 90] Brillhart, John
“Note on Irreducibility Testing”

“The Transcendental Risch Differential Equation”
J. Symbolic Computation (1990) 9, pp49-60 Feb 1988
IBM Research Report RC13460 IBM Corp. Yorktown Heights, NY
We present a new rational algorithm for solving Risch differential equations in towers of transcendental elementary extensions. In contrast to a recent algorithm by Davenport we do not require a progressive reduction of the denominators involved, but use weak normality to obtain a formula for the denominator of a possible solution. Implementation timings show this approach to be faster than a Hermite-like reduction.

[Bronstein 90a] Bronstein, Manuel
“Integration of Elementary Functions”

We extend a recent algorithm of Trager to a decision procedure for the indefinite integration of elementary functions. We can express the integral as an elementary function or prove that it is not elementary. We show that if the problem of integration in finite terms is solvable on a given elementary function field \( k \), then it is solvable in any algebraic extension of \( k(\theta) \), where \( \theta \) is a logarithm or exponential of an element of \( k \). Our proof considers an element of such an extension field to be an algebraic function of one variable over \( k \).

In his algorithm for the integration of algebraic functions, Trager describes a Hermite-type reduction to reduce the problem to an integrand with only simple finite poles on the associated Riemann surface. We generalize that technique to curves over liouvillian ground fields, and use it to simplify our integrands. Once the multiple finite poles have been removed, we use the Puiseux expansions of the integrand at infinity and a generalization of the residues to compute the integral. We also generalize a result of Rothstein that gives us a necessary condition for elementary integrability, and provide examples of its use.

[Bronstein 90c] Bronstein, M.
“On the integration of elementary functions”

[Bronstein 93] Bronstein, Manuel; Salvy, Bruno
“Full partial fraction decomposition of rational functions”
www.acm.org/pubs/citations/proceedings/issac/164081/

[Bronstein 98] Bronstein, M.
“The lazy hermite reduction”
Rapport de Recherche RR-3562, INRIA, 1998

[Bronstein 98b] Bronstein, Manuel
“Symbolic Integration Tutorial”
INRIA Sophia Antipolis ISSN 1998 Rostock
C

[Carlson 65] Carlson B C
“On Computing Elliptic Integrals and Functions”
J Math Phys. 44 36–51. (1965)

[Carlson 77a] Carlson B C
“Elliptic Integrals of the First Kind”

[Carlson 77b] Carlson B C
“Special Functions of Applied Mathematics”
Academic Press. (1977)

[Carlson 78] Carlson B C,
“Computing Elliptic Integrals by Duplication”
(Preprint) Department of Physics, Iowa State University. (1978)

[Carlson 88] Carlson B C,
“A Table of Elliptic Integrals of the Third Kind”

[Cauchy 1829] Augustin-Luc Cauchy
“Exercices de Mathématiques Quatrième Année. De Bure Frères”

[Childs 79] Childs B; Scott M; Daniel J W; Denman E; Nelson P (eds)
“Codes for Boundary-value Problems in Ordinary Differential Equations”
Lecture Notes in Computer Science. 76 (1979) Springer-Verlag

[Clausen 89] Clausen, M.; Fortenbacher, A.
“Efficient Solution of Linear Diophantine Equations”
JSC (1989) 8, 201-216

[Clenshaw 55] Clenshaw C W,
“A Note on the Summation of Chebyshev Series”
Math. Tables Aids Comput. 9 118–120. (1955)

[Clenshaw 60] Clenshaw C W
“Curve Fitting with a Digital Computer”

[Clenshaw 62] Clenshaw C W
“Mathematical Tables. Chebyshev Series for Mathematical Functions”
HMSO. (1962)

[Cline 84] Cline A K; Renka R L,
“A Storage-efficient Method for Construction of a Thiessen Triangulation”
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Conway 87] Conway, J.; Curtis, R.; Norton, S.; Parker, R.; Wilson, R.
“Atlas of Finite Groups”

[Conway 03] Conway, John H.; Smith, Derek, A.
“On Quaternions and Octonions”

[Cox 72] Cox M G
“The Numerical Evaluation of B-splines”

[CH 73] Cox M G; Hayes J G
“Curve fitting: a guide and suite of algorithms for the non-specialist user”

[Cox 74a] Cox M G
“A Data-fitting Package for the Non-specialist User”

[Cox 74b] Cox M G
“Numerical methods for the interpolation and approximation of data by spline functions”

[Cox 75] Cox M G
“An Algorithm for Spline Interpolation”

[Cox 77] Cox M G
“A Survey of Numerical Methods for Data and Function Approximation”

[Cox 78] Cox M G
“The Numerical Evaluation of a Spline from its B-spline Representation”

[Curtis 74] Curtis A R; Powell M J D; Reid J K
“On the Estimation of Sparse Jacobian Matrices”

D

[Dahlquist 74] Dahlquist G; Bjork A
“Numerical Methods”
Prentice- Hall. (1974)
[Dalmas 98] Dalmas, Stephane; Arsac, Olivier
“The INRIA OpenMath Library”
Projet SAFIR, INRIA Sophia Antipolis Nov 25, 1998

[Dantzig 63] Dantzig G B
“Linear Programming and Extensions”
Princeton University Press. (1963)

[Davenport] Davenport, James
“On Brillhart Irreducibility.”
To appear.

[Davenport 93] Davenport, J.H.
“Primality testing revisited”
Downer’s Grove, IL, USA and Oxford, UK, August 1993
www.nag.co.uk/doc/TechRep/axiomtr.html

[Davis 67] Davis P J; Rabinowitz P
“Numerical Integration”
Blaisdell Publishing Company. 33–52. (1967)

[Davis 75] Davis P J; Rabinowitz P
“Methods of Numerical Integration”
Academic Press. (1975)

[DeBoor 72] De Boor C
“On Calculating with B-splines”
J. Approx. Theory. 6 50–62. (1972)

[De Doncker 78] De Doncker E,
“An Adaptive Extrapolation Algorithm for Automatic Integration”
Signum Newsletter. 13 (2) 12–18. (1978)

[Demmel 89] Demmel J W
“On Floating-point Errors in Cholesky”

[Dennis 77] Dennis J E Jr; More J J
“Quasi-Newton Methods, Motivation and Theory”
SIAM Review. 19 46–89. 1977

[Dennis 81] Dennis J E Jr; Schnabel R B
“A New Derivation of Symmetric Positive-Definite Secant Updates”
Nonlinear Programming 4. (ed O L Mangasarian, R R Meyer and S M. Robinson)

[Dennis 83] Dennis J E Jr; Schnabel R B
“Numerical Methods for Unconstrained Optimixation and Nonlinear Equations”
Prentice-Hall.(1983)
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Dierckx 75] Dierckx P
“An Algorithm for Smoothing, Differentiating and Integration of Experimental Data Using Spline Functions”

[Dierckx 81] Dierckx P
“An Improved Algorithm for Curve Fitting with Spline Functions”

[Dierckx 82] Dierckx P
“A Fast Algorithm for Smoothing Data on a Rectangular Grid while using Spline Functions”

[Dongarra 79] Dongarra J J; Moler C B; Bunch J R; Stewart G W
“LINPACK Users’ Guide”
SIAM, Philadelphia. (1979)

[Dongarra 85] Dongarra J J; Du Croz J J; Hammarling S; Hanson R J
“A Proposal for an Extended set of Fortran Basic Linear Algebra Subprograms”
SIGNUM Newsletter. 20 (1) 2–18. (1985)

[Dongarra 88] Dongarra, Jack J.; Du Croz, Jeremy; Hammarling, Sven; Hanson, Richard J.
“An Extended Set of FORTRAN Basic Linear Algebra Subroutines”
ACM Transactions on Mathematical Software, Vol 14, No 1, March 1988, pp 1-17

[Dongarra 88a] Dongarra, Jack J.; Du Croz, Jeremy; Hammarling, Sven; Hanson, Richard J.
“ALGORITHM 656: An Extended Set of Basic Linear Algebra Subprograms: Model Implementation and Test Programs”

[Dongarra 90] Dongarra, Jack J.; Du Croz, Jeremy; Hammarling, Sven; Duff, Iain S.
“A Set of Level 3 Basic Linear Algebra Subprograms”
ACM Transactions on Mathematical Software, Vol 16, No 1, March 1990, pp 1-17

[Dongarra 90a] Dongarra, Jack J.; Du Croz, Jeremy; Hammarling, Sven; Duff, Iain S.
“ALGORITHM 679: A Set of Level 3 Basic Linear Algebra Subprograms: Model Implementation and Test Programs”

[Ducos 00] Ducos, Lionel
“Optimizations of the subresultant algorithm”

[Duff 77] Duff I S,
“MA28 – a set of Fortran subroutines for sparse unsymmetric linear equations”
F

[Fletcher 01] Fletcher, John P.
“Symbolic processing of Clifford Numbers in C++”

[Fletcher 09] Fletcher, John P.
“Clifford Numbers and their inverses calculated using the matrix representation.”
Chemical Engineering and Applied Chemistry, School of Engineering and Applied Science, Aston University, Aston Triangle, Birmingham B4 7 ET, U. K.
www.ceac.aston.ac.uk/research/staff/jpf/papers/paper24/index.php

[Fletcher 81] Fletcher R
“Practical Methods of Optimization”

[Floyd 63] Floyd, R. W.
“Semantic Analysis and Operator Precedence”
JACM 10, 3, 316-333 (1963)

[Forsythe 57] Forsythe G E,
“Generation and use of orthogonal polynomials for data fitting with a digital computer”

[Fortenbacher 90] Fortenbacher, A.
“Efficient type inference and coercion in computer algebra”
Design and Implementation of Symbolic Computation Systems (DISCO 90) A. Miola, (ed) vol 429 of Lecture Notes in Computer Science Springer-Verlag, pp56-60

Computer algebra systems of the new generation, like Scratchpad, are characterized by a very rich type concept, which models the relationship between mathematical domains of computation. To use these systems interactively, however, the user should be freed of type information. A type inference mechanism determines the appropriate function to call. All known models which allow to define a semantics for type inference cannot express the rich “mathematical” type structure, so presently type inference is done heuristically. The following paper defines a semantics for a subproblem thereof, namely coercion, which is based on rewrite rules. From this definition, and efficient coercion algorith for Scratchpad is constructed using graph techniques.

[Fox 68] Fox L.; Parker I B.
“Chebyshev Polynomials in Numerical Analysis”
Oxford University Press. (1968)

[Franke 80] Franke R.; Nielson G
“Smooth Interpolation of Large Sets of Scattered Data”
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Fritsch 82] Fritsch F N
“PCHIP Final Specifications”

[Fritsch 84] Fritsch F N.; Butland J.
“A Method for Constructing Local Monotone Piecewise Cubic Interpolants”

[Froberg 65] Froberg C E.
“Introduction to Numerical Analysis”
Addison-Wesley. 181–187. (1965)

G

[Garcia 95] Garcia, A.; Stichtenoth, H.
“A tower of Artin-Schreier extensions of function fields attaining the Drinfeld-Vladut bound”

[Gathen 90] Gathen, Joachim von zur
“Functional Decomposition Polynomials: the Tame Case”
Journal of Symbolic Computation (1990) 9, 281-299

[Gathen 99] Gathen, Joachim von zur; Gerhard, Jürgen
“Modern Computer Algebra”

[Gautschi 79a] Gautschi W.
“A Computational Procedure for Incomplete Gamma Functions”
ACM Trans. Math. Softw. 5 466–481. (1979)

[Gautschi 79b] Gautschi W.
“Algorithm 542: Incomplete Gamma Functions”

[Gentlemen 69] Gentlemen W M
“An Error Analysis of Goertzel’s (Watt’s) Method for Computing Fourier Coefficients”

[Gentleman 73] Gentleman W M.
“Least-squares Computations by Givens Transformations without Square Roots”

[Gentleman 74] Gentleman W M.
“Algorithm AS 75. Basic Procedures for Large Sparse or Weighted Linear Least-squares Problems”
[Gentlemen 74a] Gentleman W. M.; Marovich S. B.
“More on algorithms that reveal properties of floating point arithmetic units”

[Genz 80] Genz A C.; Malik A A.
“An Adaptive Algorithm for Numerical Integration over an N-dimensional Rectangular Region”

[Gill 72] Gill P E.; Miller G F.
“An Algorithm for the Integration of Unequally Spaced Data”

“Numerical Methods for Constrained Optimization”

[Gill 76a] Gill P E.; Murray W.
“Minimization subject to bounds on the variables”

[Gill 76b] Gill P E.; Murray W.
“Algorithms for the Solution of the Nonlinear Least-squares Problem”
NAC 71 National Physical Laboratory. (1976)

[Gill 78] Gill P E.; Murray W.
“Algorithms for the Solution of the Nonlinear Least-squares Problem”

[Gill 79] Gill P E.; Murray W.
“Conjugate-gradient Methods for Large-scale Nonlinear Optimization”

[Gill 81] Gill P E.; Murray W.; Wright M H.
“Practical Optimization”
Academic Press. 1981

[Gill 82] Gill P E.; Murray W.; Saunders M A.; Wright M H.
“The design and implementation of a quadratic programming algorithm”

[Gill 84a] Gill P E.; Murray W.; Saunders M A.; Wright M H

[Gill 84b] Gill P E.; Murray W.; Saunders M A.; Wright M H
“Procedures for Optimization Problems with a Mixture of Bounds and General Linear Constraints”
[Gill 86a] Gill P. E.; Hammarling S.; Murray W.; Saunders M. A.; Wright M. H.
“User’s Guide for LSSOL (Version 1.0)”
Report SOL 86-1. Department of Operations Research, Stanford University. 1986

[Gill 86b] Gill P. E.; Murray W.; Saunders M. A.; Wright M. H.
“Some Theoretical Properties of an Augmented Lagrangian Merit Function”
Report SOL 86-6R. Department of Operations Research, Stanford University. 1986

“Initial Value Routines in the NAG Library”
ACM Trans Math Softw. 5 386–400. (1979)

[Gladwell 80] Gladwell I.; Sayers D. K.
“Computational Techniques for Ordinary Differential Equations”
Academic Press. 1980

“Vectorisation of one dimensional quadrature codes”

[Gladwell 87] Gladwell I.
“The NAG Library Boundary Value Codes”

[Goedel 40] Goedel
“The consistency of the continuum hypothesis”

[Gollan 90] H. Gollan; J. Grabmeier
“Algorithms in Representation Theory and their Realization in the Computer Algebra System Scratchpad”
Bayreuther Mathematische Schriften, Heft 33, 1990, 1-23

[Golub 89] Golub, Gene H.; Van Loan, Charles F.
“Matrix Computations”

“Matrix Computations”

[Grabmeier 87] Grabmeier, J.
“On Plesken’s root finding algorithm”
in preparation

The automatic computation of limits can be reduced to two main sub-problems. The first one is asymptotic comparison where one must decide automatically which one of two functions in a specified class dominates the other one asymptotically. The second one is asymptotic cancellation and is usually exemplified by
\[ e^x \left[ \exp\left(\frac{1}{x} + e^{-x}\right) - \exp\left(\frac{1}{x}\right) \right], \quad x \to \infty \]

In this example, if the sum is expanded in powers of $1/x$, the expansion always yields $O(x^{-k})$, and this is not enough to conclude.

In 1990, J. Shackell found an algorithm that solved both these problems for the case of $\exp - \log$ functions, i.e. functions build by recursive application of exponential, logarithm, algebraic extension and field operations to one variable and the rational numbers. D. Gruntz and G. Gonnet propose a slightly different algorithm for exp-log functions. Extensions to larger classes of functions are also discussed.

**H**

[Hache 95] Haché, G.; Le Brigand, D.
“Effective construction of algebraic geometry codes”

[Hache 95a] Haché, G.
“Computation in algebraic function fields for effective construction of algebraic-geometric codes”

[Hache 96] Haché, G.
“Construction effective des codes géométriques”
Thèse de doctorat de l’Université Pierre et Marie Curie (Paris 6), Septembre 1996.

Clarendon Press. (1976)
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Hamdy 04] Hamdy, S.
“LiDIA A library for computational number theory”
Reference manual Edition 2.1.1 May 2004
www.cdc.informatik.tu-darmstadt.de/TI/LiDIA

[Hammarling 85] Hammarling, S.
“The Singular Value Decomposition in Multivariate Statistics”
ACM Signum Newsletter. 20, 3 2–25. (1985)

[Hammersley 67] Hammersley J M; Handscomb D C.
“Monte-Carlo Methods”
Methuen. (1967)

[Hathway 1896] Hathway, Arthur S.
“A Primer Of Quaternions”
(1896)

[Hayes 70] Hayes J G.
“Curve Fitting by Polynomials in One Variable”
Numerical Approximation to Functions and Data. (ed J G Hayes) Athlone Press, Lon-
don. (1970)

[Hayes 74] Hayes J G.
“Numerical Methods for Curve and Surface Fitting”

[Hayes 74a] Hayes J G.; Halliday J.
“The Least-squares Fitting of Cubic Spline Surfaces to General Data Sets”

[Henrici 56] Henrici, Peter
“Automatic Computations with Power Series”
Journal of the Association for Computing Machinery, Volume 3, No. 1, January 1956,
10-15

[Higham 88] Higham, N.J.
“FORTRAN codes for estimating the one-norm of a real or complex matrix, with
applications to condition estimation”

[Higham 02] Higham, Nicholas J.
“Accuracy and stability of numerical algorithms”

[Hock 81] Hock W.; Schittkowski K.
“Test Examples for Nonlinear Programming Codes”
[Householder 70] Householder A S.
“The Numerical Treatment of a Single Nonlinear Equation”

[Householder 81] Householder, Alston S.
“Principles of Numerical Analysis”

[Huang 96] Huang, M.D.; Ierardi, D.
“Efficient algorithms for Riemann-Roch problem and for addition in the jacobian of a curve”

[IBM] .

[Itoh 88] Itoh, T.; Tsujii, S.
“A fast algorithm for computing multiplicative inverses in $GF(2^m)$ using normal bases”
Inf. and Comp. 78, pp.171-177, 1988

[Iyanaga 77] Iyanaga, Shokichi; Iyanaga, Yukiyosi Kawada
“Encyclopedic Dictionary of Mathematics”
1977

[J] [Jacobson 68] Jacobson, N.
“Structure and Representations of Jordan Algebras”
AMS, Colloquium Publications Volume 39

[James 81] James, Gordon; Kerber, Adalbert
“The Representation Theory of the Symmetric Group”

[Jaswon 77] Jaswon, M A.; Symm G T.
“Integral Equation Methods in Potential Theory and Elastostatics”
Academic Press. (1977)

[Jeffrey 04] Jeffrey, Alan
“Handbook of Mathematical Formulas and Integrals”
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Jenning 66] Jennings A
“A Compact Storage Scheme for the Solution of Symmetric Linear Simultaneous Equations”

K

[Kalkbrener 91] Kalkbrener, M.
“Three contributions to elimination theory”
Ph. D. Thesis, University of Linz, Austria, 1991

[Kalkbrener 98] Kalkbrener, M.
“Algorithmic properties of polynomial rings”
Journal of Symbolic Computation 1998

[Kantor 89] Kantor, I.L.; Solodovnikov, A.S.
“Hypercomplex Numbers”

[Kaufmann 00] Kaufmann, Matt; Manolios, Panagiotis; Moore J Strother
“Computer-Aided Reasoning: An Approach”

[Knuth 71] Knuth, Donald
“The Art of Computer Programming”

[Knuth 84] Knuth, Donald
The TeXbook.

[Knuth 92] Knuth, Donald E.
“Literate Programming”
Center for the Study of Language and Information ISBN 0-937073-81-4 Stanford CA (1992)

[Knu98] Donald Knuth
“The Art of Computer Programming” Vol. 3 (Sorting and Searching) Addison-Wesley 1998

[Kolchin 73] Kolchin, E.R.
“Differential Algebra and Algebraic Groups”

[Koutschan 10] Koutschan, Christoph
“Axiom / FriCAS”
www.risc.jku.at/education/courses/ws2010/cas/axiom.pdf
[Kozen 86] Kozen, Dexter; Landau, Susan
“Polynomial Decomposition Algorithms”

L

[Lamport 86] Lamport, Leslie
LaTeX: A Document Preparation System,

[Lautrup 71] Lautrup B.
“An Adaptive Multi-dimensional Integration Procedure”
Proc. 2nd Coll. on Advanced Methods in Theoretical Physics, Marseille. (1971)

[Lawson 77] Lawson C L.
“Software for C Surface Interpolation”

[Lawson 74] Lawson C L.; Hanson R J.
“Solving Least-squares Problems”
Prentice-Hall. (1974)

“Algorithm 539: Basic linear algebra subprograms for FORTRAN usage”
ACM Transactions on Mathematical Software, Vol 5 No 3 September 1979 pp 308-323

[Lawson 79] Lawson C L; Hanson R J; Kincaid D R; Krogh F T
“Basic Linear Algebra Subprograms for Fortran Usage”
ACM Trans. Math. Softw. 5 308–325. (1979)

[Lazard 91] Lazard, D.
“A new method for solving algebraic systems of positive dimension”

[Lazard 92] Lazard, D.
“Solving Zero-dimensional Algebraic Systems”

[Lazard 90] Lazard, Daniel; Rioboo, Renaud
“Integration of rational functions: Rational computation of the logarithmic part”

[Le Brigand 88] Le Brigand, D.; Risler, J.J.
“Algorithm de Brill-Noether et codes de Goppa”

[Legendre 11] Legendre, George L.; Grazini, Stefano
“Pasta by Design”
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Lenstra 87] Lenstra, H. W.; Schoof, R. J.
“Primitive Normal Bases for Finite Fields”

[Lewis 77] Lewis J G,
“Algorithms for sparse matrix eigenvalue problems”

[Lidl 83] Lidl, R.; Niederreiter, H.
“Finite Field, Encyclopaedia of Mathematics and Its Applications”

“Structured Programming: Theory and Practice”
Addison-Wesley (March 1979) ISBN 0201144611

[Lipson 81] Lipson, D.
“Elements of Algebra and Algebraic Computing”

[Loetzsch 09] Loetzsch, M.
“GTFL - A graphical terminal for Lisp”
martin-loetzsch.de/gtfl/

[Lösch 60] Lösch, Friedrich
“Tables of Higher Functions”
McGraw-Hill Book Company 1960

[LTU10] .
“Lambda the Ultimate”
lambda-the-ultimate.org/node/3663#comment-62440

“The Special Functions and their Approximations”

“The Special Functions and their Approximations”

[Lyness 83] Lyness J N.
“When not to use an automatic quadrature routine”
SIAM Review. 25 63-87. (1983)
M

[Mac Lane 79] Mac Lane, Saunders; Birkhoff, Garret
“Algebra”
AMS Chelsea Publishing ISBN 0821816462

[Malcolm 72] Malcolm M. A.

[Malcolm 76] Malcolm M A.; Simpson R B.
“Local Versus Global Strategies for Adaptive Quadrature”

[Marden 66] Marden M.
“Geometry of Polynomials”

[Marshak 07] Marshak, U.
“HT-AJAX - AJAX framework for Hunchentoot”
common-lisp.net/project/ht-ajax/ht-ajax.html

[Maza 95] Maza, M. Moreno; Rioboo, R.
“Computations of gcd over algebraic towers of simple extensions”

[Maza 97] Maza, M. Moreno
“Calculs de pgcd au-dessus des tours d’extensions simples et resolution des systemes d’équations algebriques”

[Maza 98] Maza, M. Moreno
“A new algorithm for computing triangular decomposition of algebraic varieties”

[Mignotte 82] Mignotte, Maurice
“Some Useful Bounds”
Computing, Suppl. 4, 259-263 (1982), Springer-Verlag

[McCarthy 83] McCarthy G J.
“Investigation into the Multigrid Code MGD1”

[Mie97] Mielenz, Klaus D.
“Computation of Fresnel Integrals”

[Mie00] Mielenz, Klaus D.
“Computation of Fresnel Integrals II”
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Millen 68] Millen, J. K.
“CHARYBDIS: A LISP program to display mathematical expressions on typewriter-like devices”

[Minc 79] Henryk Minc
“Evaluation of Permanents”

“User Guide for Minpack-1”
ANL-80-74 Argonne National Laboratory. (1974)

[Mikhlin 67] Mikhlin S G.; Smolitsky K L.
“Approximate Methods for the Solution of Differential and Integral Equations”
Elsevier. (1967)

“The Finite Difference Method in Partial Differential Equations”
Wiley. (1980)

[Moler 73] Moler C B.; Stewart G W.
“An Algorithm for Generalized Matrix Eigenproblems”

[Mulders 97] Mulders. Thom
“A note on subresultants and a correction to the hazard/rioboo/trager formula in rational function integration”

[Munksgaard 80] Munksgaard N.
“Solving Sparse Symmetric Sets of Linear Equations by Pre-conditioned Conjugate Gradients”

[Murray 72] Murray W., (ed)
“Numerical Methods for Unconstrained Optimization”
Academic Press. (1972)

[Murtagh 83] Murtagh B A.; Saunders M A
“MINOS 5.0 User’s Guide”

[Musser 78] Musser, David R.
“On the Efficiency of a Polynomial Irreducibility Test”
N

[Nijenhuis 78] Nijenhuis and Wilf
“Combinatorial Algorithms”

[Nikolai 79] Nikolai P J.
“Algorithm 538: Eigenvectors and eigenvalues of real generalized symmetric matrices
by simultaneous iteration”
ACM Trans. Math. Softw. 5 118–125. (1979)

O

[NIST10] Olver, Frank W.; Lozier, Daniel W.; Boisvert, Ronald F.; Clark, Charles W. (ed)
“NIST Handbook of Mathematical Functions”

[OpenM] .
“OpenMath Technical Overview”
www.openmath.org/overview/technical.html

[Ortega 70] Ortega J M.; Rheinboldt W C.
“Iterative Solution of Nonlinear Equations in Several Variables”

[Ostrogradsky 1845] Ostrogradsky, M.W.
“De l’intégration des fractions rationelles.”
Bulletin de la Classe Physico-Mathématiques de l’Académie Impériale des Sciences de
St. Pétersbourg, IV:145-167,286-300, 1845

P

[Paige 75] Paige C C.; Saunders M A.
“Solution of Sparse Indefinite Systems of Linear Equations”

[Paige 82a] Paige C C.; Saunders M A.
“LSQR: An Algorithm for Sparse Linear Equations and Sparse Least-squares”

[Paige 82b] Paige C C.; Saunders M A.
“ALGORITHM 583 LSQR: Sparse Linear Equations and Least-squares Problems”

[Parker 84] Parker, R. A.
“The Computer Calculation of Modular Characters (The Meat-Axe)”
1984
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Parlett 80] Parlett B N.
“The Symmetric Eigenvalue Problem”
Prentice-Hall. 1980

[Parnas 10] Parnas, David Lorge; Jin, Ying
“Defining the meaning of tabular mathematical expressions”
Science of Computer Programming V75 No.11 Nov 2010 pp980-1000 Elsevier

[Parnas 95] Parnas, David Lorge; Madey, Jan
“Functional Documents for Computer Systems”

[Paul 81] Paul, Richard
“Robot Manipulators”
MIT Press 1981

[Pearcey 56] Pearcey, T.
“Table of the Fresnel Integral”
Cambridge University Press 1956

[Pereyra 79] Pereyra V.
“PASVA3: An Adaptive Finite-Difference Fortran Program for First Order Nonlinear,
Ordinary Boundary Problems”
Codes for Boundary Value Problems in Ordinary Differential Equations. Lecture Notes
in Computer Science. (ed B Childs, M Scott, J W Daniel, E Denman and P Nelson)
76 Springer-Verlag. (1979)

[Peters 67a] Peters G.
“NPL Algorithms Library”
Document No. F2/03/A. (1967)

[Peters 67b] Peters G.
“NPL Algorithms Library”
Document No. F1/04/A (1967)

[Peters 70] Peters G.; Wilkinson J H.
“The Least-squares Problem and Pseudo-inverses”

[Peters 71] Peters G.; Wilkinson J H.
“Practical Problems Arising in the Solution of Polynomial Equations”

[Pierce 82] R.S. Pierce
“Associative Algebras”
90693-2
[Piessens 73] Piessens R.
“An Algorithm for Automatic Integration”
Angewandte Informatik. 15 399–401. (1973)

[Piessens 74] Piessens R.; Mertens I.; Branders M.
“Integration of Functions having End-point Singularities”

[Piessens 75] Piessens R.; Branders M.
“Algorithm 002. Computation of Oscillating Integrals”

[Piessens 76] Piessens R.; Van Roy-Branders M.; Mertens I.
“The Automatic Evaluation of Cauchy Principal Value Integrals”
Angewandte Informatik. 18 31–35. (1976)

[Piessens 83] Piessens R.; De Doncker-Kapenga E.; Uberhuber C.; Kahaner D.
“QUADPACK, A Subroutine Package for Automatic Integration”
Springer-Verlag.(1983)

[Polya 37] Polya, G.
“Kombinatorische Anzahlbestimmungen fur Gruppen, Graphen und chemische
Verbindungen”

[Powell 70] Powell M J D.
“A Hybrid Method for Nonlinear Algebraic Equations”
Numerical Methods for Nonlinear Algebraic Equations. (ed P Rabinowitz) Gordon and
Breach. (1970)

[Powell 74] Powell M J D.
“Introduction to Constrained Optimization”
Numerical Methods for Constrained Optimization. (ed P E Gill and W Murray) Aca-

Mathematical Programming: The State of the Art. (ed A Bachem, M Groetschel and

[Pratt 73] Pratt, Vaughan R.
“Top down operator precedence”
POPL ’73 Proceedings of the 1st annual ACM SIGACT-SIGPLAN symposium on
Principles of programming languages
hall.org.ua/halls/wizzard/pdf/Vaughan.Pratt.TDOP.pdf

[Press 95] Press, William H.; Teukolsky, Saul A.; Vetterling, William T.; Flannery, Brian P.
“Numerical Recipes in C”
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Pryce 77] Pryce J D.; Hargrave B A.
“The Scale Pruefer Method for one-parameter and multi-parameter eigenvalue problems in ODEs”

[Pryce 81] Pryce J D.
“Two codes for Sturm-Liouville problems”

[Pryce 86] Pryce J D.
“Error Estimation for Phase-function Shooting Methods for Sturm-Liouville Problems”

[Puffinware 09] Puffinware LLC.
“Singular Value Decomposition (SVD) Tutorial”
www.puffinwarellc.com/p3a.htm

[Quintana-Orti 06] Quintana-Orti, Gregorio; van de Geijn, Robert
“Improving the performance of reduction to Hessenberg form”

[Rabinowitz 70] Rabinowitz P.
“Numerical Methods for Nonlinear Algebraic Equations”
Gordon and Breach. (1970)

[Ralston 65] Ralston A.
“A First Course in Numerical Analysis”
McGraw-Hill. 87–90. (1965)

[Ramakrishnan 03] Ramakrishnan, Maya
“A Gentle Introduction to Lyapunov Functions”
ORSUM August 2003
www.or.ms.unimelb.edu.au/handouts/lyaptalk.1.pdf

[Ramsey 03] Ramsey, Norman
“Noweb–A Simple, Extensible Tool for Literate Programming”
www.eecs.harvard.edu/~nr/noweb

[Redfield 27] Redfield, J.H.
“The Theory of Group-Reduced Distributions”
[Reinsch 67] Reinsch C H.
“Smoothing by Spline Functions”

[Renka 84] Renka R L.
“Algorithm 624: Triangulation and Interpolation of Arbitrarily Distributed Points in the Plane”

[Renka 84] Renka R L.; Cline A K.
“A Triangle-based C Interpolation Method”

[Reutenauer 93] Reutenauer, Christophe
“Free Lie Algebras”
Oxford University Press, June 1993 ISBN 0198536798

“Rule-based Mathematics”
www.apmaths.uwo.ca/~arich

[Richardson 94] Richardson, Dan; Fitch, John
“The identity problem for elementary functions and constants”

[Richtmyer 67] Richtmyer R D.; Morton K W.
“Difference Methods for Initial-value Problems”

[Rioboo 92] Rioboo, R.
“Real algebraic closure of an ordered field, implementation in Axiom”

[Rioboo 96] Rioboo, R.
“Generic computation of the real closure of an ordered field”
In Mathematics and Computers in Simulation Volume 42, Issue 4-6, November 1996.

[Ritt 50] Ritt, Joseph Fels
“Differential Algebra”

[Rote 01] Rote, Günter
“Division-free algorithms for the determinant and the Pfaffian”
page.mi.fu-berlin.de/rote/Papers/pdf/Division-free+algorithms.pdf
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Rubey 07] Rubey, Martin
"Formula Guessing with Axiom"
April 2007

[Rutishauser 69] Rutishauser H.
"Computational aspects of F L Bauer’s simultaneous iteration method"

[Rutishauser 70] Rutishauser H.
"Simultaneous iteration method for symmetric matrices"

S

[Schafer 66] Schafer, R.D.
"An Introduction to Nonassociative Algebras"
Academic Press, New York, 1966

[Schoenberg 53] Schoenberg I.J.; Whitney A.
"On Polya Frequency Functions III"

[Schoenhage 82] Schoenhage, A.
"The fundamental theorem of algebra in terms of computational complexity"
preliminary report, Univ. Tuebingen, 1982

[Schonfelder 76] Schonfelder J.L.
"The Production of Special Function Routines for a Multi-Machine Library"
Software Practice and Experience. 6(1) (1976)

[Seggern 93] von Seggern, David Henry
"CRC Standard Curves and Surfaces"

[Seiler 95a] Seiler, W.M.; Calmet, J.
"JET – An Axiom Environment for Geometric Computations with Differential Equations"

[Shepard 68] Shepard D.
"A Two-dimensional Interpolation Function for Irregularly Spaced Data"

[Sims 71] Sims, C.
"Determining the Conjugacy Classes of a Permutation Group"

[Sit 92] Sit, William
"An Algorithm for Parametric Linear Systems"
J. Sym. Comp., April 1992
[Smith 67] Smith B T.
“ZERPOL: A Zero Finding Algorithm for Polynomials Using Laguerre’s Method”
Technical Report, Department of Computer Science, University of Toronto, Canada.
(1967)

[Smith 85] Smith G D.

[Sobol 74] Sobol I M.
“The Monte Carlo Method”
The University of Chicago Press. 1974

[Steele 90] Steele, Guy L.
“Common Lisp The Language”

[Stichtenoth 93] Stichtenoth, H.
“Algebraic function fields and codes”
Springer-Verlag, 1993, University Text.

[Stinson 90] Stinson, D.R.
“Some observations on parallel Algorithms for fast exponentiation in GF(2^n)”
Siam J. Comp., Vol.19, No.4, pp.711-717, August 1990

[Stroud 66] Stroud A H.; Secrest D.
“Gaussian Quadrature Formulas”
Prentice-Hall. (1966)

[Stroud 71] Stroud A H.
“Approximate Calculation of Multiple Integrals”
Prentice-Hall 1971

[Swarztrauber 79] Swarztrauber P N.; Sweet R A.
“Efficient Fortran Subprograms for the Solution of Separable Elliptic Partial Differential Equations”

[Swarztrauber 84] Swarztrauber P N.
“Fast Poisson Solvers”
(1984)

[Tait 1890] Tait, P.G.
“An Elementary Treatise on Quaternions”
C.J. Clay and Sons, Cambridge University Press Warehouse, Ave Maria Lane 1890
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

[Taivalsaari 96] Taivalsaari, Antero
“On the Notion of Inheritance”

[Temme 87] Temme N M.
“On the Computation of the Incomplete Gamma Functions for Large Values of the Parameters”

[Temperton 83a] Temperton C.
“Self-sorting Mixed-radix Fast Fourier Transforms”

[Temperton 83b] Temperton C.
“Fast Mixed-Radix Real Fourier Transforms”

[U]

[Unknown 61] Unknown
“Chebyshev-series”
16 HMSO. 1961

[V]

[Van Dooren 76] Van Dooren P.; De Ridder L.
“An Adaptive Algorithm for Numerical Integration over an N-dimensional Cube”

[van Hoeij 94] van Hoeij, M.
“An algorithm for computing an integral basis in an algebraic function field”

[Van Loan 92] Van Loan, C.
“Computational Frameworks for the Fast Fourier Transform”
SIAM Philadelphia. (1992)

[W]

“Finite Element Analysis and Application”
Wiley. (1985)
[Wang 92] Wang, D.M.
“An implementation of the characteristic set method in Maple”
Proc. DISCO’92 Bath, England

[Ward 75] Ward, R C.
“The Combination Shift QZ Algorithm”

[Watt 03] Watt, Stephen
“Aldor”
www.aldor.org

[Weil 71] Weil, André
“Courbes algébriques et variétés Abéliennes”
Hermann, Paris, 1971

[Weisstein] Weisstein, Eric W.
“Hypergeometric Function”
MathWorld - A Wolfram Web Resource
mathworld.wolfram.com/HypergeometricFunction.html

[Weitz 03] Weitz, E.
“CL-WHO -Yet another Lisp markup language”
www.weitz.de/cl-who/

[Weitz 06] Weitz, E.
“HUNCHENTOOT - The Common Lisp web server formerly known as TBNL”
www.weitz.de/hunchentoot/

[Wesseling 82a] Wesseling, P.
“MGD1 - A Robust and Efficient Multigrid Method”

[Wesseling 82b] Wesseling, P.
“Theoretical Aspects of a Multigrid Method”

[Wicks 89] Wicks, Mark; Carlisle, David, Rahtz, Sebastian
“dvipdfm.def”
web.mit.edu/texsrc/source/latex/graphics/dvipdfm.def

[Wiki 3] .
“Givens Rotations”
en.wikipedia.org/wiki/Givens_rotation

[Williamson 85] Williamson, S.G.
“Combinatorics for Computer Science”
1.2. AXIOM CITATIONS OF EXTERNAL SOURCES

“Handbook for Automatic Computation II, Linear Algebra”
Springer-Verlag. 1971

“Rounding Errors in Algebraic Processes”
Chapter 2. HMSO. (1963)

“The Algebraic Eigenvalue Problem”
Oxford University Press. (1965)

[Wilkinson 78] Wilkinson J H.
“Singular Value Decomposition – Basic Aspects”
(1978)

“Kronecker’s Canonical Form and the QZ Algorithm”
Linear Algebra and Appl. 28 285–303. 1979

[Wisbauer 91] Wisbauer, R.
“Bimodule Structure of Algebra”
Lecture Notes Univ. Duesseldorf 1991

[Woerz-Busekros 80] Woerz-Busekros, A.
“Algebra in Genetics”
Lectures Notes in Biomathematics 36, Springer-Verlag, Heidelberg, 1980

[Wolberg 67] Wolberg J R.
“Prediction Analysis”
Van Nostrand. (1967)

[Wolfram 09] Wolfram Research
mathworld.wolfram.com/Quaternion.html

[Wu 87] Wu, W.T.
“A Zero Structure Theorem for polynomial equations solving”
MM Research Preprints, 1987

[Wynn 56] Wynn P.
“On a Device for Computing the $e_m(S_n)$ Transformation”

Y

[Yun 76] Yun, D.Y.Y.
“On square-free decomposition algorithms”
Proceedings of SYMSAC’76 pages 26-35, 1976
1.3 Special Topics

Solving Systems of Equations

[Bronstein 86] Bronstein, Manuel
“Gsolve: a faster algorithm for solving systems of algebraic equations”

We apply the elimination property of Gröbner bases with respect to pure lexicographic ordering to solve systems of algebraic equations. We suggest reasons for this approach to be faster than the resultant technique, and give examples and timings that show that it is indeed faster and more correct, than MACSYMA’s solve.

Numerical Algorithms

[Bronstein 99] Bronstein, Manuel
“Fast Deterministic Computation of Determinants of Dense Matrices”
www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html

In this paper we consider deterministic computation of the exact determinant of a dense matrix $M$ of integers. We present a new algorithm with worst case complexity

$$O(n^4 \log n + \log ||M|| + x^3 \log^2 ||M||)$$

, where $n$ is the dimension of the matrix and $||M||$ is a bound on the entries in $M$, but with average expected complexity

$$O(n^4 + m^3 (\log n + \log ||M||)^2)$$

, assuming some plausible properties about the distribution of $M$. We will also describe a practical version of the algorithm and include timing data to compare this algorithm with existing ones. Our result does not depend on “fast” integer or matrix techniques.

[Kelsey 00] Kelsey, Tom
“Exact Numerical Computation via Symbolic Computation”
tom.host.cs.st-andrews.ac.uk/pub/ccapaper.pdf

We provide a method for converting any symbolic algebraic expression that can be converted into a floating point number into an exact numeric representation. We use this method to demonstrate a suite of procedures for the representation of, and arithmetic over, exact real numbers in the Maple computer algebra system. Exact reals are represented by potentially infinite lists of binary digits, and interpreted as sums of negative powers of the golden ratio.

[Yang 14] Yang, Xiang; Mittal, Rajat
“Acceleration of the Jacobi iterative method by factors exceeding 100 using scheduled
Special Functions

[Corless 05] Corless, Robert M.; Jeffrey, David J.; Watt, Stephen M.; Bradford, Russell; Davenport, James H.
“Reasoning about the elementary functions of complex analysis”

There are many problems with the simplification of elementary functions, particularly over the complex plane. Systems tend to make “howlers” or not to simplify enough. In this paper we outline the “unwinding number” approach to such problems, and show how it can be used to prevent errors and to systematise such simplification, even though we have not yet reduced the simplification process to a complete algorithm. The unsolved problems are probably more amenable to the techniques of artificial intelligence and theorem proving than the original problem of complex-variable analysis.

Exponential Integral $E_1(x)$

[Segletes 98] Segletes, S.B.
“A compact analytical fit to the exponential integral $E_1(x)$

A four-parameter fit is developed for the class of integrals known as the exponential integral (real branch). Unlike other fits that are piecewise in nature, the current fit to the exponential integral is valid over the complete domain of the function (compact) and is everywhere accurate to within $\pm 0.0052\%$ when evaluating the first exponential integral, $E_1$. To achieve this result, a methodology that makes use of analytically known limiting behaviors at either extreme of the domain is employed. Because the fit accurately captures limiting behaviors of the $E_1$ function, more accuracy is retained when the fit is used as part of the scheme to evaluate higher-order exponential integrals, $E_n$, as compared with the use of brute-force fits to $E_1$, which fail to accurately model limiting behaviors. Furthermore, because the fit is compact, no special accommodations are required (as in the case of spliced piecewise fits) to smooth the value, slope, and higher derivatives in the transition region between two piecewise domains. The general methodology employed to develop this fit is outlined, since it may be used for other problems as well.

[Segletes 09] Segletes, S.B.
“Improved fits for $E_1(x)$ vis-à-vis those presented in ARL-TR-1758
This is a writeup detailing the more accurate fits to $E_1(x)$, relative to those presented in ARL-TR-1758. My actual fits are to

$$F_1 = [x \exp(x)E_1(x)]$$

which spans a functional range from 0 to 1. The best accuracy I have been yet able to achieve, defined by limiting the value of

$$[(F_1)_{fit} - F_1] / F_1$$

over the domain, is approximately 3.1E-07 with a 12-parameter fit, which unfortunately isn’t quite to 32-bit floating-point accuracy. Nonetheless, the fit is not a piecewise fit, but rather a single continuous function over the domain of non-negative $x$, which avoids some of the problems associated with piecewise domain splicing.

**Polynomial GCD**

[Knuth 71] Knuth, Donald  
“The Art of Computer Programming”  
2nd edition Vol. 2 (Seminumerical Algorithms) 1st edition, 2nd printing,  
Addison-Wesley 1971, section 4.6 pp399-505

[Ma 90] Ma, Keju; Gathen, Joachim von zur  
“Analysis of Euclidean Algorithms for Polynomials over Finite Fields”  

This paper analyzes the Euclidean algorithm and some variants of it for computing the greatest common divisor of two univariate polynomials over a finite field. The minimum, maximum, and average number of arithmetic operations both on polynomials and in the ground field are derived.

[Naylor 00a] Naylor, Bill  
“Polynomial GCD Using Straight Line Program Representation”  
PhD. Thesis, University of Bath, 2000  
www.sci.csd.uwo.ca/~bill/thesis.ps

This thesis is concerned with calculating polynomial greatest common divisors using straight line program representation.

In the Introduction chapter, we introduce the problem and describe some of the traditional representations for polynomials, we then talk about some of the general subjects central to the thesis, terminating with a synopsis of the category theory which is central to the Axiom computer algebra system used during this research.
1.3. SPECIAL TOPICS

The second chapter is devoted to describing category theory. We follow with a chapter detailing the important sections of computer code written in order to investigate the straight line program subject. The following chapter on evaluation strategies and algorithms which are dependant on these follows, the major algorithm which is dependant on evaluation and which is central to our thesis being that of equality checking. This is indeed central to many mathematical problems. Interpolation, that is the determination of coefficients of a polynomial is the subject of the next chapter. This is very important for many straight line program algorithms, as their non-canonical structure implies that it is relatively difficult to determine coefficients, these being the basic objects that many algorithms work on. We talk about three separate interpolation techniques and compare their advantages and disadvantages. The final two chapters describe some of the results we have obtained from this research and finally conclusions we have drawn as to the viability of the straight line program approach and possible extensions.

Finally we terminate with a number of appendices discussing side subjects encountered during the thesis.

[Shoup 93] Shoup, Victor
“Factoring Polynomials over Finite Fields: Asymptotic Complexity vs Reality*”

This paper compares the algorithms by Berlekamp, Cantor and Zassenhaus, and Gathen and Shoup to conclude that (a) if large polynomials are factored the FFT should be used for polynomial multiplication and division, (b) Gathen and Shoup should be used if the number of irreducible factors of \( f \) is small. (c) if nothing is know about the degrees of the factors then Berlekamp’s algorithm should be used

[Gathen 01] Gathen, Joachim von zur; Panario, Daniel
“Factoring Polynomials Over Finite Fields: A Survey”
people.csail.mit.edu/dmoshkov/courses/codes/poly-factorization.pdf

This survey reviews several algorithms for the factorization of univariate polynomials over finite fields. We emphasize the main ideas of the methods and provide and up-to-date bibliography of the problem. This paper gives algorithms for squarefree factorization, distinct-degree factorization, and equal-degree factorization. The first and second algorithms are deterministic, the third is probabilistic.

[van Hoeij] Hoeij, Mark van; Monagen, Michael
“Algorithms for Polynomial GCD Computation over Algebraic Function Fields”
www.cecm.sfu.ca/personal/mmonagan/papers/AFGCD.pdf

Let \( L \) be an algebraic function field in \( k \geq 0 \) parameters \( t_1, \ldots, t \) \( k \). Let \( f_1, f_2 \) be non-zero polynomials in \( L[x] \). We give two algorithms for computing their \( \text{gcd} \). The first, a modular \( \text{gcd} \) algorithm, is an extension of the modular \( \text{gcd} \) algorithm for \( \mathbb{Z}[x_1, \ldots, x_n] \) and Encarnacion for \( \mathbb{Q}(\alpha[x]) \) to function fields. The second, a fraction-free algorithm, is a modification of the Moreno Maza
and Rioobo algorithm for computing gcds over triangular sets. The modification reduces coefficient growth in \( L \) to be linear. We give an empirical comparison of the two algorithms using implementations in Maple.

[Wang 78] Wang, Paul S.
“An Improved Multivariate Polynomial Factoring Algorithm”

A new algorithm for factoring multivariate polynomials over the integers based on an algorithm by Wang and Rothschild is described. The new algorithm has improved strategies for dealing with the known problems of the original algorithm, namely, the leading coefficient problem, the bad-zero problem and the occurrence of extraneous factors. It has an algorithm for correctly predetermining leading coefficients of the factors. A new and efficient p-adic algorithm named EEZ is described. Basically it is a linearly convergent variable-by-variable parallel construction. The improved algorithm is generally faster and requires less store than the original algorithm. Machine examples with comparative timing are included.

“Polynomial greatest common divisor”
en.wikipedia.org/wiki/Polynomial_greatest_common_divisor

Category Theory

[Baez 09] Baez, John C.; Stay, Mike
“Physics, Topology, Logic and Computation: A Rosetta Stone”
arxiv.org/pdf/0903.0340v3.pdf

In physics, Feynman diagrams are used to reason about quantum processes. In the 1980s, it became clear that underlying these diagrams is a powerful analogy between quantum physics and topology. Namely, a linear operator behaves very much like a “cobordism”: a manifold representing spacetime, going between two manifolds representing space. But this was just the beginning: similar diagrams can be used to reason about logic, where they represent proofs, and computation, where they represent programs. With the rise of interest in quantum cryptography and quantum computation, it became clear that there is an extensive network of analogies between physics, topology, logic and computation. In this expository paper, we make some of these analogies precise using the concept of “closed symmetric monodial category”. We assume no prior knowledge of category theory, proof theory or computer science.

[Meijer 91] Meijer, Erik; Fokkinga, Maarten; Paterson, Ross
“Functional Programming with Bananas, Lenses, Envelopes and Barbed Wire”
eprints.eemcs.utwente.nl/7281/01/db-utwente-40501F46.pdf
1.3. SPECIAL TOPICS

We develop a calculus for lazy functional programming based on recursion operators associated with data type definitions. For these operators we derive various algebraic laws that are useful in deriving and manipulating programs. We shall show that all example functions in Bird and Wadler’s “Introduction to Functional Programming” can be expressed using these operators.

[Youssef 04] Youssef, Saul
“Prospects for Category Theory in Aldor”
October 2004

Ways of incorporating category theory constructions and results into the Aldor language are discussed. The main features of Aldor which make this possible are identified, examples of categorical constructions are provided and a suggestion is made for a foundation for rigorous results.

Proving Axiom Correct

[Bertot 04] Bertot, Yves; Castéran, Pierre
“Interactive Theorem Proving and Program Development”
Springer ISBN 3-540-20854-2

Coq is an interactive proof assistant for the development of mathematical theories and formally certified software. It is based on a theory called the calculus of inductive constructions, a variant of type theory.

This book provides a pragmatic introduction to the development of proofs and certified programs using Coq. With its large collection of examples and exercises it is an invaluable tool for researchers, students, and engineers interested in formal methods and the development of zero-fault software.

[Boulme 00] Boulmé, S.; Hardin, T.; Rioboo, R.
“Polymorphic Data Types, Objects, Modules and Functors,: is it too much?”

Abstraction is a powerful tool for developers and it is offered by numerous features such as polymorphism, classes, modules, and functors, . . . A working programmer may be confused by this abundance. We develop a computer algebra library which is being certified. Reporting this experience made with a language (Ocaml) offering all these features, we argue that the are all needed together. We compare several ways of using classes to represent algebraic concepts, trying to follow as close as possible mathematical specification. Then we show how to combine classes and modules to produce code having very strong typing properties. Currently, this library is made of one hundred units of functional code and behaves faster than analogous ones such as Axiom.

“On the way to certify Computer Algebra Systems”
Calculemus-2001
The FOC project aims at supporting, within a coherent software system, the entire process of mathematical computation, starting with proved theories, ending with certified implementations of algorithms. In this paper, we explain our design requirements for the implementation, using polynomials as a running example. Indeed, proving correctness of implementations depends heavily on the way this design allows mathematical properties to be truly handled at the programming level.

The FOC project, started at the fall of 1997, is aimed to build a programming environment for the development of certified symbolic computation. The working languages are Coq and Ocaml. In this paper, we present first the motivations of the project. We then explain why and how our concern for proving properties of programs has led us to certain implementation choices in Ocaml. This way, the sources express exactly the mathematical dependencies between different structures. This may ease the achievement of proofs.

[Daly 10] Daly, Timothy
“Intel Instruction Semantics Generator”
daly.axiom-developer.org/TimothyDaly_files/publications/sei/intel/intel.pdf

Given an Intel x86 binary, extract the semantics of the instruction stream as Conditional Concurrent Assignments (CCAs). These CCAs represent the semantics of each individual instruction. They can be composed to represent higher level semantics.

[Danielsson 06] Danielsson, Nils Anders; Hughes, John; Jansson, Patrik; Gibbons, Jeremy
“Fast and Loose Reasoning is Morally Correct”
ACM POPL’06 January 2005, Charleston, South Carolina, USA

Functional programmers often reason about programs as if they were written in a total language, expecting the results to carry over to non-total (partial) languages. We justify such reasoning.

Two languages are defined, one total and one partial, with identical syntax. The semantics of the partial language includes partial and infinite values, and all types are lifted, including the function spaces. A partial equivalence relation (PER) is then defined, the domain of which is the total subset of the partial language. For types not containing function spaces the PER relates equal values, and functions are related if they map related values to related values.

It is proved that if two closed terms have the same semantics in the total language, then they have related semantics in the partial language. It is also shown that the PER gives rise to a bicartesian closed category which can be used to reason about values in the domain of the relation.

[Davenport 12] Davenport, James H.; Bradford, Russell; England, Matthew; Wilson, David
“Program Verification in the presence of complex numbers, functions with branch cuts etc.”
arxiv.org/pdf/1212.5417.pdf
In considering the reliability of numerical programs, it is normal to “limit our study to the semantics dealing with numerical precision”. On the other hand, there is a great deal of work on the reliability of programs that essentially ignores the numerics. The thesis of this paper is that there is a class of problems that fall between these two, which could be described as “does the low-level arithmetic implement the high-level mathematics”. Many of these problems arise because mathematics, particularly the mathematics of the complex numbers, is more difficult than expected: for example the complex function log is not continuous, writing down a program to compute an inverse function is more complicated than just solving an equation, and many algebraic simplification rules are not universally valid.

The good news is that these problems are theoretically capable of being solved, and are practically close to being solved, but not yet solved, in several real-world examples. However, there is still a long way to go before implementations match the theoretical possibilities.

[Dolzmann 97] Dolzmann, Andreas; Sturm, Thomas
“Guarded Expressions in Practice”
redlog.dolzmann.de/papers/pdf/MIP-9702.pdf

Computer algebra systems typically drop some degenerate cases when evaluating expressions, e.g. $x/x$ becomes 1 dropping the case $x = 0$. We claim that it is feasible in practice to compute also the degenerate cases yielding guarded expressions. We work over real closed fields but our ideas about handling guarded expressions can be easily transferred to other situations. Using formulas as guards provides a powerful tool for heuristically reducing the combinatorial explosion of cases: equivalent, redundant, tautological, and contradictive cases can be detected by simplification and quantifier elimination. Our approach allows to simplify the expressions on the basis of simplification knowledge on the logical side. The method described in this paper is implemented in the REDUCE package GUARDIAN, which is freely available on the WWW.

[Dos Reis 11] Dos Reis, Gabriel; Matthews, David; Li, Yue
“Retargeting OpenAxiom to Poly/ML: Towards an Integrated Proof Assistants and Computer Algebra System Framework”
Calculemus (2011) Springer paradise.caltech.edu/~yli/paper/oa-polyml.pdf

This paper presents an ongoing effort to integrate the Axiom family of computer algebra systems with Poly/ML-based proof assistants in the same framework. A long term goal is to make a large set of efficient implementations of algebraic algorithms available to popular proof assistants, and also to bring the power of mechanized formal verification to a family of strongly typed computer algebra systems at a modest cost. Our approach is based on retargeting the code generator of the OpenAxiom compiler to the Poly/ML abstract machine.

[Dunstan 00a] Dunstan, Martin N.
“Adding Larch/Aldor Specifications to Aldor”
We describe a proposal to add Larch-style annotations to the Aldor programming language, based on our PhD research. The annotations are intended to be machine-checkable and may be used for a variety of purposes ranging from compiler optimizations to verification condition (VC) generation. In this report we highlight the options available and describe the changes which would need to be made to the compiler to make use of this technology.

[Dunstan 98] Dunstan, Martin; Kelsey, Tom; Linton, Steve; Martin, Ursula
“Lightweight Formal Methods For Computer Algebra Systems”
www.cs.st-andrews.ac.uk/~tom/pub/issac98.pdf

Demonstrates the use of formal methods tools to provide a semantics for the type hierarchy of the Axiom computer algebra system, and a methodology for Aldor program analysis and verification. There are examples of abstract specifications of Axiom primitives.

[Dunstan 99a] Dunstan, MN
“Larch/Aldor - A Larch BISL for AXIOM and Aldor”

In this thesis we investigate the use of lightweight formal methods and verification conditions (VCs) to help improve the reliability of components constructed within a computer algebra system. We follow the Larch approach to formal methods and have designed a new behavioural interface specification language (BISL) for use with Aldor: the compiled extension language of Axiom and a fully-featured programming language in its own right. We describe our idea of lightweight formal methods, present a design for a lightweight verification condition generator and review our implementation of a prototype verification condition generator for Larch/Aldor.

[Dunstan 00] Dunstan, Martin; Kelsey, Tom; Martin, Ursula; Linton, Steve
“Formal Methods for Extensions to CAS”

We demonstrate the use of formal methods tools to provide a semantics for the type hierarchy of the AXIOM computer algebra system, and a methodology for Aldor program analysis and verification. We give a case study of abstract specifications of AXIOM primitives, and provide an interface between these abstractions and Aldor code.

[Hardin 13] Hardin, David S.; McClurg, Jedidiah R.; Davis, Jennifer A.
“Creating Formally Verified Components for Layered Assurance with an LLVM to ACL2 Translator”

This paper describes an effort to create a library of formally verified software component models from code that have been compiled using the Low-Level Virtual Machine (LLVM) intermediate form. The idea is to build a translator from
LLVM to the applicative subset of Common Lisp accepted by the ACL2 theorem prover. They perform verification of the component model using ACL2’s automated reasoning capabilities.

[Hardin 14] Hardin, David S.; Davis, Jennifer A.; Greve, David A.; McClurg, Jedidiah R.
“Development of a Translator from LLVM to ACL2”

In our current work a library of formally verified software components is to be created, and assembled, using the Low-Level Virtual Machine (LLVM) intermediate form, into subsystems whose top-level assurance relies on the assurance of the individual components. We have thus undertaken a project to build a translator from LLVM to the applicative subset of Common Lisp accepted by the ACL2 theorem prover. Our translator produces executable ACL2 formal models, allowing us to both prove theorems about the translated models as well as validate those models by testing. The resulting models can be translated and certified without user intervention, even for code with loops, thanks to the use of the def::ung macro which allows us to defer the question of termination. Initial measurements of concrete execution for translated LLVM functions indicate that performance is nearly 2.4 million LLVM instructions per second on a typical laptop computer. In this paper we overview the translation process and illustrate the translator’s capabilities by way of a concrete example, including both a functional correctness theorem as well as a validation test for that example.

[Lamport 02] Lamport, Leslie
“Specifying Systems”

[Mason 86] Mason, Ian A.
“The Semantics of Destructive Lisp”
Center for the Study of Language and Information ISBN 0-937073-06-7

Our basic premise is that the ability to construct and modify programs will not improve without a new and comprehensive look at the entire programming process. Past theoretical research, say, in the logic of programs, has tended to focus on methods for reasoning about individual programs; little has been done, it seems to us, to develop a sound understanding of the process of programming – the process by which programs evolve in concept and in practice. At present, we lack the means to describe the techniques of program construction and improvement in ways that properly link verification, documentation and adaptability.

[Newcombe 13] Newcombe, Chris; Rath, Tim; Zhang, Fan; Munteanu, Bogdan; Brooker, Marc; Deardeuff, Michael
“Use of Formal Methods at Amazon Web Services”

In order to find subtle bugs in a system design, it is necessary to have a precise description of that design. There are at least two major benefits to writing a precise
design; the author is forced to think more clearly, which helps eliminate “plausible hand-waving”, and tools can be applied to check for errors in the design, even while it is being written. In contrast, conventional design documents consist of prose, static diagrams, and perhaps pseudo-code in an ad hoc untestable language. Such descriptions are far from precise; they are often ambiguous, or omit critical aspects such as partial failure or the granularity of concurrency (i.e. which constructs are assumed to be atomic). At the other end of the spectrum, the final executable code is unambiguous, but contains an overwhelming amount of detail.

We needed to be able to capture the essence of a design in a few hundred lines of precise description. As our designs are unavoidably complex, we need a highly-expressive language, far above the level of code, but with precise semantics. That expressivity must cover real-world concurrency and fault-tolerance. And, as we wish to build services quickly, we wanted a language that is simple to learn and apply, avoiding esoteric concepts. We also very much wanted an existing ecosystem of tools. We found what we were looking for in TLA+, a formal specification language.

[Poll 99a] Poll, Erik
“The Type System of Axiom”

This is a slide deck from a talk on the correspondence between Axiom/Aldor types and Logic.

[Poll 99] Poll, Erik; Thompson, Simon
“The Type System of Aldor”
www.cs.kent.ac.uk/pubs/1999/874/content.ps

This paper gives a formal description of – at least a part of – the type system of Aldor, the extension language of the Axiom. In the process of doing this a critique of the design of the system emerges.

[Poll (a)] Poll, Erik; Thompson, Simon
“Adding the axioms to Axiom. Toward a system of automated reasoning in Aldor”
citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.7.1457&rep=rep1&type=ps

This paper examines the proposal of using the type system of Axiom to represent a logic, and thus to use the constructions of Axiom to handle the logic and represent proofs and propositions, in the same way as is done in theorem provers based on type theory such as Nuprl or Coq.

The paper shows an interesting way to decorate Axiom with pre- and post-conditions.

The Curry-Howard correspondence used is

<table>
<thead>
<tr>
<th>PROGRAMMING</th>
<th>LOGIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Formula</td>
</tr>
<tr>
<td>Program</td>
<td>Proof</td>
</tr>
<tr>
<td>Product/record type</td>
<td>(...,...)</td>
</tr>
<tr>
<td>Sum/union type</td>
<td>/</td>
</tr>
</tbody>
</table>
1.3. SPECIAL TOPICS

<table>
<thead>
<tr>
<th>Type</th>
<th>Implication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function type</td>
<td>-&gt;</td>
</tr>
<tr>
<td>Dependent function type</td>
<td>(x:A) -&gt; B(x)</td>
</tr>
<tr>
<td>Dependent product type</td>
<td>(x:A, B(x))</td>
</tr>
<tr>
<td>Empty type</td>
<td>Exit</td>
</tr>
<tr>
<td>One element type</td>
<td>Triv</td>
</tr>
</tbody>
</table>

Function type -> Implication
Dependent function type (x:A) -> B(x) Universal quantifier
Dependent product type (x:A, B(x)) Existential quantifier
Empty type Exit Contradictory proposition
One element type Triv True proposition

[Poll 00] Poll, Erik; Thompson, Simon
“Integrating Computer Algebra and Reasoning through the Type System of Aldor”

A number of combinations of reasoning and computer algebra systems have been proposed; in this paper we describe another, namely a way to incorporate a logic in the computer algebra system Axiom. We examine the type system of Aldor – the Axiom Library Compiler – and show that with some modifications we can use the dependent types of the system to model a logic, under the Curry-Howard isomorphism. We give a number of example applications of the logic we construct and explain a prototype implementation of a modified type-checking system written in Haskell.

**Interval Arithmetic**

[Boehm 86] Boehm, Hans-J.; Cartwright, Robert; Riggle, Mark; O’Donnell, Michael J.
“Exact Real Arithmetic: A Case Study in Higher Order Programming”
[Boehm 86]

[Briggs 04] Briggs, Keith
“Exact real arithmetic”
[Briggs 04]

[Fateman 94] Fateman, Richard J.; Yan, Tak W.
“Computation with the Extended Rational Numbers and an Application to Interval Arithmetic”
[Fateman 94]

Programming languages such as Common Lisp, and virtually every computer algebra system (CAS), support exact arbitrary-precision integer arithmetic as well as exact rational number computation. Several CAS include interval arithmetic directly, but not in the extended form indicated here. We explain why changes to the usual rational number system to include infinity and “not-a-number” may be useful, especially to support robust interval computation. We describe techniques for implementing these changes.

[Lambov 06] Lambov, Branimir
“Interval Arithmetic Using SSE-2”
Numerics

[LeFevre 06] LeFèvre, Vincent; Stehlé, Damien; Zimmermann, Paul
“Worst Cases for the Exponential Function in the IEEE-754r decimal64 Format”

We searched for the worst cases for correct rounding of the exponential function in
the IEEE 754r decimal64 format, and computed all the bad cases whose distance
from a breakpoint (for all rounding modes) is less than \(10^{-15}\) ulp, and we give
the worst ones. In particular, the worst case for \(|x| \geq 3\times10^{-11}\) is

\[
\exp(9.407822313572878\times10^{-2}) = 1.098645682066338500000000000000278\ldots
\]

This work can be extended to other elementary functions in the decimal64 format
and allows the design of reasonably fast routines that will evaluate these functions
with correct rounding, at least in some situations.

[Hamming 62] Hamming R W.
“Numerical Methods for Scientists and Engineers”

Advanced Documentation

[Bostock 14] Bostock, Mike
“Visualizing Algorithms”
bost.ocks.org/mike/algorithms

This website hosts various ways of visualizing algorithms. The hope is that these
kind of techniques can be applied to Axiom.

[Leeuwen] van Leeuwen, André M.A.
“Representation of mathematical object in interactive books”

We present a model for the representation of mathematical objects in struc-
tured electronic documents, in a way that allows for interaction with applications
such as computer algebra systems and proof checkers. Using a representation
that reflects only the intrinsic information of an object, and storing application-
dependent information in so-called application descriptions, it is shown how the
translation from the internal to an external representation and vice versa

[Soiffer 91] Soiffer, Neil Morrell
“The Design of a User Interface for Computer Algebra Systems”
www.eecs.berkeley.edu/Pubs/TechRpts/1991/CSD-91-626.pdf
This thesis discusses the design and implementation of natural user interfaces for Computer Algebra Systems. Such an interface must not only display expressions generated by the Computer Algebra System in standard mathematical notation, but must also allow easy manipulation and entry of expressions in that notation. The user interface should also assist in understanding of large expressions that are generated by Computer Algebra Systems and should be able to accommodate new notational forms.

[Victor 11] Victor, Bret
“Up and Down the Ladder of Abstraction”
worrydream.com/LadderOfAbstraction

This interactive essay presents the ladder of abstraction, a technique for thinking explicitly about these levels, so a designer can move among them consciously and confidently.

[Victor 12] Victor, Bret
“Inventing on Principle”
www.youtube.com/watch?v=PUv66718DII

This video raises the level of discussion about human-computer interaction from a technical question to a question of effectively capturing ideas. In particular, this applies well to Axiom’s focus on literate programming.

Differential Equations

[Abramov 95] Abramov, Sergei A.; Bronstein, Manuel; Petkovsek, Marko
“On Polynomial Solutions of Linear Operator Equations”
www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html

[Abramov 01] Abramov, Sergei; Bronstein, Manuel
“On Solutions of Linear Functional Systems”
www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html

We describe a new direct algorithm for transforming a linear system of recurrences into an equivalent one with nonsingular leading or trailing matrix. Our algorithm, which is an improvement to the EG elimination method, uses only elementary linear algebra operations (ranks, kernels, and determinants) to produce an equation satisfied by the degree of the solutions with finite support. As a consequence, we can bound and compute the polynomial and rational solutions of very general linear functional systems such as systems of differential or \((q-)\)difference equations.

[Bronstein 96a] Bronstein, Manuel; Petkovsek, Marko
“An introduction to pseudo-linear algebra”
www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html
Pseudo-linear algebra is the study of common properties of linear differential and difference operators. We introduce in this paper its basic objects (pseudo-derivations, skew polynomials, and pseudo-linear operators) and describe several recent algorithms on them, which, when applied in the differential and difference cases, yield algorithms for uncoupling and solving systems of linear differential and difference equations in closed form.

[Bronstein xb] Bronstein, Manuel

“Computer Algebra Algorithms for Linear Ordinary Differential and Difference equations”


Galois theory has now produced algorithms for solving linear ordinary differential and difference equations in closed form. In addition, recent algorithmic advances have made those algorithms effective and implementable in computer algebra systems. After introducing the relevant parts of the theory, we describe the latest algorithms for solving such equations.

[Bronstein 94] Bronstein, Manuel

“An improved algorithm for factoring linear ordinary differential operators”

www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html

We describe an efficient algorithm for computing the associated equations appearing in the Beke-Schlesinger factorisation method for linear ordinary differential operators. This algorithm, which is based on elementary operations with sets of integers, can be easily implemented for operators of any order, produces several possible associated equations, of which only the simplest can be selected for solving, and often avoids the degenerate case, where the order of the associated equation is less than in the generic case. We conclude with some fast heuristics that can produce some factorizations while using only linear computations.

[Bronstein 90] Bronstein, Manuel

“On Solutions of Linear Ordinary Differential Equations in their Coefficient Field”

www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html

We describe a rational algorithm for finding the denominator of any solution of a linear ordinary differential equation in its coefficient field. As a consequence, there is now a rational algorithm for finding all such solutions when the coefficients can be built up from the rational functions by finitely many algebraic and primitive adjunctions. This also eliminates one of the computational bottlenecks in algorithms that either factor or search for Liouvillian solutions of such equations with Liouvillian coefficients.

[Bronstein 96] Bronstein, Manuel

“Σ^T – A strongly-typed embeddable computer algebra library”

www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html

We describe the new computer algebra library Σ^T and its underlying design. The development of Σ^T is motivated by the need to provide highly efficient implementations of key algorithms for linear ordinary differential and (q)-difference
equations to scientific programmers and to computer algebra users, regardless of the programming language or interactive system they use. As such, $\Sigma^IF$ is not a computer algebra system per se, but a library (or substrate) which is designed to be “plugged” with minimal efforts into different types of client applications.

[Bronstein 99a] Bronstein, Manuel
“Solving linear ordinary differential equations over $C(x, e \int f(x)dx)$
www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html

We describe a new algorithm for computing the solutions in

$$F = C(x, e \int f(x)dx)$$

of linear ordinary differential equations with coefficients in $F$. Compared to the general algorithm, our algorithm avoids the computation of exponential solutions of equations with coefficients in $C(x)$, as well as the solving of linear differential systems over $C(x)$. Our method is effective and has been implemented.

[Bronstein 00] Bronstein, Manuel
“On Solutions of Linear Ordinary Differential Equations in their Coefficient Field”
www-sop.inria.fr/cafe/Manuel.Bronstein/publications/mb_papers.html

We extend the notion of monomial extensions of differential fields, i.e. simple transcendental extensions in which the polynomials are closed under differentiation, to difference fields. The structure of such extensions provides an algebraic framework for solving generalized linear difference equations with coefficients in such fields. We then describe algorithms for finding the denominator of any solution of those equations in an important subclass of monomial extensions that includes transcendental indefinite sums and products. This reduces the general problem of finding the solutions of such equations in their coefficient fields to bounding their degrees. In the base case, this yields in particular a new algorithm for computing the rational solutions of $q$-difference equations with polynomial coefficients.

[Bronstein 02] Bronstein, Manuel; Lafaille, Sébastien
“Solutions of linear ordinary differential equations in terms of special functions”

We describe a new algorithm for computing special function solutions of the form $y(x) = m(x)F(\eta(x))$ of second order linear ordinary differential equations, where $m(x)$ is an arbitrary Liouvillian function, $\eta(x)$ is an arbitrary rational function, and $F$ satisfies a given second order linear ordinary differential equations. Our algorithm, which is base on finding an appropriate point transformation between the equation defining $F$ and the one to solve, is able to find all rational transformations for a large class of functions $F$, in particular (but not only) the $qF_1$ and $1F_1$ special functions of mathematical physics, such as Airy, Bessel, Kummer and Whittaker functions. It is also able to identify the values of the parameters entering those special functions, and can be generalized to equations of higher order.
We propose a definition of regularity of a linear differential system with coefficients in a monomial extension of a differential field, as well as a global and truly rational (i.e., factorisation-free) iteration that transforms a system with regular finite singularities into an equivalent one with simple finite poles. We then apply our iteration to systems satisfied by bases of algebraic function fields, obtaining algorithms for computing the number of irreducible components and the genus of algebraic curves.

We relate sequences generated by recurrences with polynomial coefficients to interleaving and multiplexing of sequences generated by recurrences with constant coefficients. In the special case of finite fields, we show that such sequences are periodic and provide linear complexity estimates for all three constructions.

Picard-Vessiot extensions for ordinary differential and difference equations are well known and are at the core of the associated Galois theories. In this paper, we construct fundamental matrices and Picard-Vessiot extensions for systems of linear partial functional equations having finite linear dimension. We then use those extensions to show that all the solutions of a factor of such a system can be completed to solutions of the original system.

Computer algebra systems often have to deal with piecewise continuous functions. These are, for example, the absolute value function, signum, piecewise defined functions but also functions that are the supremum or infimum of two functions. We present a new algebraic approach to these types of problems. This paper presents a normal form for a function ring containing piecewise polynomial functions of an expression. The main result is that this normal form can be used to decide extensional equality of two piecewise functions. Also we define supremum and infimum for piecewise functions; in fact, we show that the function
ring forms a lattice. Additionally, a method to solve equalities and inequalities in this function ring is presented. Finally, we give a “user interface” to the algebraic representation of the piecewise functions.

[Weber 06] Weber, Andreas
“Quantifier Elimination on Real Closed Fields and Differential Equations”

This paper surveys some recent applications of quantifier elimination on real closed fields in the context of differential equations. Although polynomial vector fields give rise to solutions involving the exponential and other transcendental functions in general, many questions can be settled within the real closed field without referring to the real exponential field. The technique of quantifier elimination on real closed fields is not only of theoretical interest, but due to recent advances on the algorithmic side including algorithms for the simplification of quantifier-free formulae the method has gained practical applications, e.g. in the context of computing threshold conditions in epidemic modeling.

Expression Simplification

[Carette 04] Carette, Jacques
“Understanding Expression Simplification”
www.cas.mcmaster.ca/~carette/publications/simplification.pdf

We give the first formal definition of the concept of simplification for general expressions in the context of Computer Algebra Systems. The main mathematical tool is an adaptation of the theory of Minimum Description Length, which is closely related to various theories of complexity, such as Kolmogorov Complexity and Algorithmic Information Theory. In particular, we show how this theory can justify the use of various “magic constants” for deciding between some equivalent representations of an expression, as found in implementations of simplification routines.

Integration

[Baddoura 94] Baddoura, Mohamed Jamil
“Integration in Finite Terms with Elementary Functions and Dilogarithms”
dspace.mit.edu/bitstream/handle/1721.1/26864/30757785.pdf

In this thesis, we report on a new theorem that generalizes Liouville’s theorem on integration in finite terms. The new theorem allows dilogarithms to occur in the integral in addition to elementary functions. The proof is based on two identities for the dilogarithm, that characterize all the possible algebraic relations among dilogarithms of functions that are built up from the rational functions by taking transcendental exponentials, dilogarithms, and logarithms.
[Bronstein 97] Bronstein, M.  
“Symbolic Integration I–Transcendental Functions.”  

[Bronstein 05a] Bronstein, Manuel  
“The Poor Man’s Integrator, a parallel integration heuristic”  
www-sop.inria.fr/cafe/Manuel.Bronstein/pmint/pmint.txt  
www-sop.inria.fr/cafe/Manuel.Bronstein/pmint/examples

[Cherry 84] Cherry, G.W.  
“Integration in Finite Terms with Special Functions: The Error Function”  
A decision procedure for integrating a class of transcendental elementary func- 
tions in terms of elementary functions and error functions is described. The proce- 
dure consists of three mutually exclusive cases. In the first two cases a generalised  
procedure for completing squares is used to limit the error functions which can  
appear in the integral of a finite number. This reduces the problem to the so- 
lution of a differential equation and we use a result of Risch (1969) to solve it.  
The third case can be reduced to the determination of what we have termed  
∑-decompositions. The result presented here is the key procedure to a more  
general algorithm which is described fully in Cherry (1983).

[Cherry 86] Cherry, G.W.  
“Integration in Finite Terms with Special Functions: The Logarithmic Integral”  

[Cherry 89] Cherry, G.W.  
“An Analysis of the Rational Exponential Integral”  

[Davenport 79b] Davenport, James Harold  
“On the Integration of Algebraic Functions”  

[Davenport 82] Davenport, J.H.  
“On the Parallel Risch Algorithm (III): Use of Tangents”  
SIGSAM V16 no. 3 pp3-6 August 1982

[Fateman 02] Fateman, Richard  
“Symbolic Integration”  
inst.eecs.berkeley.edu/~cs282/sp02/lects/14.pdf

“The Risch Integration Algorithm”  
Algorithms for Computer Algebra, Ch 12 pp511-573 (1992)

[Hardy 1916] Hardy, G.H.  
“The Integration of Functions of a Single Variable”  
Cambridge University Press, Cambridge, 1916
1.3. SPECIAL TOPICS

[Hermite 1872] Hermite, E.
“Sur l’intégration des fractions rationelles.”
Nouvelles Annales de Mathématiques (2ème série), 11:145-148, 1872

[Jeffrey 97] Jeffrey, D.J.; Rich, A.D.
“Recursive integration of piecewise-continuous functions”
www.cybertester.com/data/recint.pdf

An algorithm is given for the integration of a class of piecewise-continuous functions. The integration is with respect to a real variable, because the functions considered do not in general allow integration in the complex plane to be defined. The class of integrands includes commonly occurring waveforms, such as square waves, triangular waves, and the floor function; it also includes the signum function. The algorithm can be implemented recursively, and it has the property of ensuring that integrals are continuous on domains of maximum extent.

“Integration of the signum, piecewise and related functions”
cs.uwaterloo.ca/~glabahn/Papers/issac99-2.pdf

When a computer algebra system has an assumption facility, it is possible to distinguish between integration problems with respect to a real variable, and those with respect to a complex variable. Here, a class of integration problems is defined in which the integrand consists of compositions of continuous functions and signum functions, and integration is with respect to a real variable. Algorithms are given for evaluating such integrals.

[Knowles 93] Knowles, P.
“Integration of a class of transcendental liouvillian functions with error-functions i”

[Knowles 95] Knowles, P.
“Integration of a class of transcendental liouvillian functions with error-functions ii”

[Lang 93] Lang, S.
“Algebra”
Addison-Wesly, New York, 3rd edition 1993

[Liouville 1833a] Liouville, Joseph
“Premier mémoire sur la détermination des intégrales dont la valeur est algébrique”

[Liouville 1833b] Liouville, Joseph
“Second mémoire sur la détermination des intégrales dont la valeur est algébrique”
Journal de l’Ecole Polytechnique, 14:149-193, 1833

[Liouville 1833c] Liouville, Joseph
“Note sur la determination des intégrales dont la valeur est algébrique”
[Liouville 1833d] Liouville, Joseph
“Sur la determination des intégrales dont la valeur est algébrique”

[Liouville 1835] Liouville, Joseph
“Mémoire sur l’intégration d’une classe de fonctions transcendentes”
Journal für die Reine und Angewandte Mathematik, Vol 13(2) pp 93-118, (1835)

[Moses 71a] Moses, Joel
“Symbolic Integration: The Stormy Decade”
www-inst.eecs.berkeley.edu/~cs282/sp02/readings/moses-int.pdf

Three approaches to symbolic integration in the 1960’s are described. The first, from artificial intelligence, led to Slagle’s SAINT and to a large degree to Moses’ SIN. The second, from algebraic manipulation, led to Monove’s implementation and to Horowitz’ and Tobey’s reexamination of the Hermite algorithm for integrating rational functions. The third, from mathematics, led to Richardson’s proof of the unsolvability of the problem for a class of functions and for Risch’s decision procedure for the elementary functions. Generalizations of Risch’s algorithm to a class of special functions and programs for solving differential equations and for finding the definite integral are also described.

[Ostrowski 46] Ostrowski, A.
“Sur l’intégrabilité élémentaire de quelques classes d’expressions”

[Raab 13] Raab, Clemens G.
“Generalization of Risch’s Algorithm to Special Functions”
arxiv.org/pdf/1305.1481.pdf

Symbolic integration deals with the evaluation of integrals in closed form. We present an overview of Risch’s algorithm including recent developments. The algorithms discussed are suited for both indefinite and definite integration. They can also be used to compute linear relations among integrals and to find identities for special functions given by parameter integrals. The aim of this presentation is twofold: to introduce the reader to some basic idea of differential algebra in the context of integration and to raise awareness in the physics community of computer algebra algorithms for indefinite and definite integration.

[Raab xx] Raab, Clemens G.
“Integration in finite terms for Liouvillian functions”
www.mmrc.iss.ac.cn/~dart4/posters/Raab.pdf

Computing integrals is a common task in many areas of science, antiderivatives are one way to accomplish this. The problem of integration in finite terms can be stated as follows. Given a differential field \((F, D)\) and \(f \in F\), compute \(g\) in some elementary extension of \((F, D)\) such that \(Dg = f\) if such a \(g\) exists.

This problem has been solved for various classes of fields \(F\). For rational functions \((C(x), \frac{d}{dx})\) such a \(g\) always exists and algorithms to compute it are known already
for a long time. In 1969 Risch published an algorithm that solves this problem when \((F, D)\) is a transcendental elementary extension of \((C(x), \frac{d}{dx})\). Later this has been extended towards integrands being Liouvillian functions by Singer et. al. via the use of regular log-explicit extensions of \((C(x), \frac{d}{dx})\). Our algorithm extends this to handling transcendental Liouvillian extensions \((F, D)\) of \((C, 0)\) directly without the need to embed them into log-explicit extensions. For example, this means that

\[
\int (z - x)x^{z-1}e^{-x}dx = x^ze^{-x}
\]

can be computed without including \(\log(x)\) in the differential field.

[Risch 68] Risch, Robert
“On the integration of elementary functions which are built up using algebraic operations”
Research Report SP-2801/002/00, System Development Corporation, Santa Monica, CA, USA, 1968

[Risch 69a] Risch, Robert
“Further results on elementary functions”
Research Report RC-2042, IBM Research, Yorktown Heights, NY, USA, 1969

[Risch 69b] Risch, Robert
“The problem of integration in finite terms”

[Risch 69c] Risch, Robert
“The Solution of the Problem of Integration in Finite Terms”
S0002-9904-1970-12454-5.pdf

The problem of integration in finite terms asks for an algorithm for deciding whether an elementary function has an elementary indefinite integral and for finding the integral if it does. “Elementary” is used here to denote those functions build up from the rational functions using only exponentiation, logarithms, trigonometric, inverse trigonometric and algebraic operations. This vaguely worded question has several precise, but inequivalent formulations. The writer has devised an algorithm which solves the classical problem of Liouville. A complete account is planned for a future publication. The present note is intended to indicate some of the ideas and techniques involved.

[Risch 79] Risch, Robert
“Algebraic properties of the elementary functions of analysis”

[Ritt 48] Ritt, J.F.
“Integration in Finite Terms”
Columbia University Press, New York 1948
[Rosenlicht 72] Rosenlicht, Maxwell
“Integration in finite terms”

[Rothstein 76] Rothstein, Maxwell
“Aspects of symbolic integration and simplification of exponential and primitive functions”

[Rothstein 77] Rothstein, Michael
“A new algorithm for the integration of exponential and logarithmic functions”

[Seidenberg 58] Seidenberg, Abraham
“Abstract differential algebra and the analytic case”

[Seidenberg 69] Seidenberg, Abraham
“Abstract differential algebra and the analytic case. II”

[Singer 85] Singer, M.F.; Saunders, B.D.; Caviness, B.F.
“An extension of Liouville’s theorem on integration in finite terms”

[Trager 76] Trager, Barry
“Algebraic factoring and rational function integration”
In Proceedings of SYMSAC’76 pages 219-226, 1976

[Trager 84] Trager, Barry
“On the integration of algebraic functions”
PhD thesis, MIT, Computer Science, 1984

[Würfl 07] Würfl, Andreas
“Basic Concepts of Differential Algebra”
www14.in.tum.de/konferenzen/Jass07/courses/1/Wuerfl/wuerfl_paper.pdf

Modern computer algebra systems symbolically integrate a vast variety of functions. To reveal the underlying structure it is necessary to understand infinite integration not only as an analytical problem but as an algebraic one. Introducing the differential field of elementary functions we sketch the mathematical tools like Liouville’s Principle used in modern algorithms. We present Hermite’s method for integration of rational functions as well as the Rothstein/Trager method for rational and for elementary functions. Further applications of the mentioned algorithms in the field of ODE’s conclude this paper.
1.3. SPECIAL TOPICS

Partial Fraction Decomposition

[Angell] Angell, Tom
“Guidelines for Partial Fraction Decomposition”
www.math.udel.edu/~angell/partfrac_I.pdf

[Laval 08] Laval, Philippe B.
“Partial Fractions Decomposition”
www.math.wisc.edu/~park/Fall2011/integration/Partial%20Fraction.pdf

[Mudd 14] Harvey Mudd College
“Partial Fractions”

[Rajasekaran 14] Rajasekaran, Raja
“Partial Fraction Expansion”

[Wootton 14] Wootton, Aaron
“Integration of Rational Functions by Partial Fractions”
faculty.up.edu/wootton/calc2/section7.4.pdf

Ore Rings

This is used as a reference for the LeftOreRing category, in particular, the least left common multiple (lcmCoef) function.

[Delenclos 06] Delenclos, Jonathon; Leroy, André
“Noncommutative Symmetric functions and $W$-polynomials”
arxiv.org/pdf/math/0606614.pdf

Let $K$, $S$, $D$ be a division ring an endomorphism and a $S$-derivation of $K$, respectively. In this setting we introduce generalized noncommutative symmetric functions and obtain Viète formula and decompositions of different operators. $W$-polynomials show up naturally, their connections with $P$-independency. Vandermonde and Wronskian matrices are briefly studied. The different linear factorizations of $W$-polynomials are analysed. Connections between the existence of LLCM (least left common multiples) of monic linear polynomials with coefficients in a ring and the left duo property are established at the end of the paper.

[Abramov 05] Abramov, S.A.; Le, H.Q.; Li, Z.
“Univariate Ore Polynomial Rings in Computer Algebra”
www.mmrc.iss.ac.cn/~zmli/papers/oretools.pdf

We present some algorithms related to rings of Ore polynomials (or, briefly, Ore rings) and describe a computer algebra library for basic operations in an arbitrary Ore ring. The library can be used as a basis for various algorithms in Ore rings, in particular, in differential, shift, and $q$-shift rings.