axiom

BY

BY RICHARD D. JENKS AND ROBERT S. SUTOR

The Scientific Computation System

Numerical Algorithms Group Limited, 1992

With Contributions From

Scott C. Morrison	Manuel Bronstein	Patrizia Gianni
Jonathoon M. Steinbach	William H. Burge	Johannes Grabmeier
Barry Trager	Timothy P. Daly	William Sit
Stephen M. Watt	Michael Dewar	Clifton J. Williamson

Table of contents

Introduction to AXIOM

Welcome to the world of AXIOM. We call AXIOM a scientific computation system: a self-contained toolbox designed to meet your scientific programming needs, from symbolics, to numerics, to graphics.

This introduction is a quick overview of what AXIOM offers.

Symbolic Computation

AXIOM provides a wide range of simple commands for symbolic mathematical problem solving. Do you need to solve an equation, to expand a series, or to obtain an integral? If so, just ask AXIOM to do it.

Integrate $\frac{1}{(x^3(a+bx)^{1/3})}$ with respect to x.

```
→ integrate(1/(x**3 * (a+b*x)**(1/3)),x)
```

$$\frac{\left(\begin{array}{c} -2b^2x^2\sqrt{3}\log\left(\sqrt[3]{a}\sqrt[3]{bx+a}^2+\sqrt[3]{a}^2\sqrt[3]{bx+a}+a\right)+ \\ 4b^2x^2\sqrt{3}\log\left(\sqrt[3]{a}^2\sqrt[3]{bx+a}-a\right)+ \\ 12b^2x^2\arctan\left(\frac{2\sqrt{3}\sqrt[3]{a}^2\sqrt[3]{bx+a}+a\sqrt{3}}{3a}\right)+ \\ (12bx-9a)\sqrt{3}\sqrt[3]{a}\sqrt[3]{bx+a}^2 \end{array}\right)}{18a^2x^2\sqrt{3}\sqrt[3]{a}} \tag{1}$$

Type: Union(Expression Integer,...)

→

Axiom provides state-of-the art algebraic machinery to handle your most advanced symbolic problems. For example, AXIOM's integrator gives you the answer when an answer exists. If one does not, it provides a proof that there is no answer. Integration is just one of a multitude of symbolic operations that AXIOM provides.

Numeric Computation

AXIOM has a numerical library that includes operations for linear algebra, solutions of equations, and special functions. For many of these operations, you can select any number of floating point digits to be carried out in the computation.

Solve $x^{49}-49x^4+9$ to 49 digits of accuracy.

```
→ solve(x**49-49*x**4+9 = 0, 1.e-49)
```

$$[x=-0.65465367069042711367,\ x=1.0869213956538595085,\ x=0.65465367072552717397] \tag{1}$$

Type: List Equation Polynomial Float

→

The output of a computation can be converted to FORTRAN to be used in a later numerical computation. Besides floating point numbers, AXIOM provides literally dozens of kinds of numbers to compute with. These range from various kinds of integers, to fractions, complex numbers, quaternions, continued fractions, and to numbers represented with an arbitrary base.

What is 10 to the 100^{th} power in base 32?

```
→ radix(10**100,32)
```

$$4\#\langle I\rangle 9\#\langle L\rangle\#\langle K\rangle\#\langle I\rangle\#\langle P\rangle 9\#\langle G\rangle\#\langle R\rangle\#\langle S\rangle\#\langle T\rangle\#\langle C\rangle 5\#\langle I\rangle\#\langle F\rangle 164\#\langle P\rangle\#\langle O\rangle 5\#\langle V\rangle 72\#\langle M\rangle\#\langle E\rangle 827226\#\langle J\rangle\#\langle S\rangle\#\langle L\rangle\#\langle A\rangle\#\langle P\rangle 462585\#\langle Q\rangle 7\#\langle H|0\rangle 000000000000000\backslash 0000 \tag{3}$$

Type: RadixExpansion 32

→

Interactive Programming

AXIOMS's interactive programming language lets you define your own functions. A simple example of a user-defined function is one that computes the successive Legendre polynomials. AXIOM lets you define these polynomicals in a piece-wise way.

The first Legendre polynomial

```
→ p(0) == 1
```

Type: Void

→

The second Legendre polynomial

```
→ p(1) == x
```

Type: Void

→

The n^{th} Legendre polynomial for $(n > 1)$.

```
→ p(n) == ((2*n-1)*x*p(n-1) - (n-1) * p(n-2))/n
```

Type: Void

→

In addition to letting you define simple functions like this, the interactive language can be used to create entire application packages. All the graphs in the AXIOM Images section in the center of the book, for example, were created by programs written in the interactive language.

The above definitions for p do no computation – they simply tell AXIOM how to compute $p(k)$ for some positive integer k. To actually get a value of a Legendre polynomial, you ask for it.

What is the tenth Legendre polynomial?

```
→ p(10)
```

```
   Compiling function p with type Integer -> Polynomial Fraction
      Integer
   Compiling function p as a recurrence relation.
```

$$\frac{46189}{256}x^{10} - \frac{109395}{256}x^8 + \frac{45045}{128}x^6 - \frac{15015}{128}x^4 + \frac{3465}{256}x^2 - \frac{63}{256} \tag{4}$$

Type: Polynomial Fraction Integer

→

AXIOM applies the above pieces for p to obtain the value $p(10)$. But it does more: it creates an optimized, compiled function p. The function is formed by putting the pieces together into a single piece of code. By *compiled*, we mean that the function is translated into basic machine-code. By *optimized*, we mean that certain transformations are performed on that code to make it run faster. For p, AXIOM actually translates the original definition that is recursive (one that calls itself) to one that is iterative (one that consists of a simple loop).

What is the coefficient of x^{90} in $p(90)$?

```
→ coefficient(p(90),x,90)
```

$$\frac{5688265542052017822223458237426581853561497449095175}{773712524553626718119526 4} \tag{8}$$

Type: Polynomial Fraction Integer

→

In general, a user function is type-analyzed and compiled on first use. Later, if you use it with a different kind of object, the function is recompiled if necessary.

Data Structures

A variety of data structures are available for interactive use. These include strings, lists, vectors, sets, multisets, and hash tables. A particularly useful structure for interactive use is the infinite stream:

Create tge infinit stream of derivatives of Legendre polynomials.

```
→ [D(p(i),x) for i in 1..]
```

$$\left[1,\ 3x,\ \frac{15}{2}x^2 - \frac{3}{2},\ \frac{35}{2}x^3 - \frac{15}{2}x,\ \frac{315}{8}x^4 - \frac{105}{4}x^2 + \frac{15}{8},\right.$$

$$\frac{693}{8}x^5 - \frac{315}{4}x^3 + \frac{105}{8}x,\ \frac{3003}{16}x^6 - \frac{3465}{16}x^4 + \frac{945}{16}x^2 - \frac{35}{16},$$

$$\frac{6435}{16}x^7 - \frac{9009}{16}x^5 + \frac{3465}{16}x^3 - \frac{315}{16}x, \tag{9}$$

$$\frac{109395}{128}x^8 - \frac{45045}{32}x^6 + \frac{45045}{64}x^4 - \frac{3465}{32}x^2 + \frac{315}{128},$$

$$\left.\frac{230945}{128}x^9 - \frac{109395}{32}x^7 + \frac{135135}{64}x^5 - \frac{15015}{32}x^3 + \frac{3465}{128}x,\ ...\right]$$

Type: Stream Polynomial Fraction Integer

→

Streams display only for a few of their initial elements. Otherwise, they are "lazy": they only compute elements when you ask for them.

Data structures are an important component for building application software. Advanced users can represent data for applications in optimal fashion. In all, AXIOM offers over forty kinds of aggregate data structures, ranging from mutable structures (such as cyclic lists and flexible arrays) to storage efficient structures (such as bit vectors). As an example, streams are used as the internal data structure structure for power series.

What is the series expansion of $\log(\cot(x))$ about $x = \pi/2$?

```
→ series(log(cot(x)),x = %pi/2)
```

$$\log\left(\frac{-2x+\pi}{2}\right)+\frac{1}{3}\left(x-\frac{\pi}{2}\right)^2+\frac{7}{90}\left(x-\frac{\pi}{2}\right)^4+\frac{62}{2835}\left(x-\frac{\pi}{2}\right)^6+ \frac{127}{18900}\left(x-\frac{\pi}{2}\right)^8+\frac{146}{66825}\left(x-\frac{\pi}{2}\right)^{10}+O\left(\left(x-\frac{\pi}{2}\right)^{11}\right) \tag{10}$$

Type: GeneralUnivariatePowerSeries(Expression Integer,x,pi/2)

→

Series and streams make no attempt to compute *all* their elements! Rather, they stand ready to deliver elements on demand.

What is the coefficient of the 50^{th} term of the series?

```
→ coefficient(%,50)
```

$$\frac{44590788901016030052447242300856550965644}{7131469286438669111584090881309360354581359130859375} \tag{11}$$

Type: Expression Integer

→

Mathematical Structures

AXIOM also has many kinds of mathematical structures. These range from simple ones (like polynomials and matrices) to more esoteric ones (like ideals and Clifford algebras). Most structures allow the construction of arbitrarily complicated "types".

Even a simple input expression can result in a type with several levels.

```
→ matrix [[x+%i,0], [1,-2]]
```

$$\begin{bmatrix} x+i & 0 \\ 1 & -2 \end{bmatrix} \tag{12}$$

Type: Matrix Polynomial Complex Integer

→

The AXIOM interpreter builds types in response to user input. Often, the type of the result is changed in order to be applicable to an operation.

The inverse operation requires that elements of the above are fractions.

```
→ inverse(%)
```

$$\begin{bmatrix} \frac{1}{x+i} & 0 \\ \frac{1}{2x+2i} & -\frac{1}{2} \end{bmatrix} \tag{13}$$

```
                                    Type: Union(Matrix Fraction Polynomial Complex Integer,...)
```

→

Pattern Matching

A convenient facility for symbolic computation is "pattern matching". Suppose you have a trigonometric expression and you want to transform it to some equivalent form. Use a **rule** command to describe the transformation rules you need. Then give the rules a name and apply that name as a function to your trigonometric expressioin.

Introduce rewrite rules.

```
→ sinCosExpandRules := rule (
    sin(x+y) == sin(x)*cos(y)+sin(y)*cos(x);
    cos(x+y) == cos(x)*cos(y) - sin(x)*sin(y);
    sin(2*x) == 2*sin(x)*cos(x);
    cos(2*x) == cos(x)**2 - sin(x)**2
  );
                                    Type: Ruleset(Integer,Integer,Expression Integer)
```

→

Apply the rules to a simple trigonometric expression.

```
→ sinCosExpandRules(sin(a+2*b+c))
```

$$\left(-\cos(a)\sin(b)^2 - 2\cos(b)\sin(a)\sin(b) + \cos(a)\cos(b)^2\right)\sin(c) - \cos(c)\sin(a)\sin(b)^2 + 2\cos(a)\cos(b)\cos(c)\sin(b) + \cos(b)^2\cos(c)\sin(a) \tag{15}$$

```
                                    Type: Expression Integer
```

→

Using input files, you can create your own library of transformation rules relevant to your applications, then selectively apply the rules you need.

Polymorphic Algorthms

All componetns of the AXIOM algebra library are written in the AXIOM library language. This language is similar to the interactive language except for protocols that authors are obliged to follow. The library language permits you to write "polymorphic algorithms", algorithms defined to work in their most natural settings and over a variety of types.

Define a system of polynomial equations S.

```
→ S := [3*x**3 + y +1 = 0, y**2 = 4]
```

Solve the system S using rational number arithmetic and 30 digits of accuracy.

```
→ solve (S,1/10**30)
```

$$\left[\left[y = -2,\ x = \frac{1757879671211184245283070414507}{2535301200456458802993406410752}\right],\ [y = 2,\ x = -1]\right] \tag{17}$$

Type: List List Equation Polynomial Fraction Integer

→

Solve S with the solution expressed in radicals.

→ radicalSolve(S)

$$\left[[y=2,\ x=-1],\ \left[y=2,\ x=\frac{-\sqrt{-3}+1}{2}\right],\right.$$

$$\left[y=2,\ x=\frac{\sqrt{-3}+1}{2}\right],\ \left[y=-2,\ x=\frac{1}{\sqrt[3]{3}}\right], \tag{18}$$

$$\left.\left[y=-2,\ x=\frac{\sqrt{-1}\sqrt{3}-1}{2\sqrt[3]{3}}\right],\ \left[y=-2,\ x=\frac{-\sqrt{-1}\sqrt{3}-1}{2\sqrt[3]{3}}\right]\right]$$

Type: List List Equation Expression Integer

→

While these solutions look very different, the results were produced by the same internal algorithm! The internal algorithm actually works with equations over any "field". Examples of fields are the rational numbers, floating point numbers, rational fractions, power series, and general expressions involving radicals.

Extensibility

Users and system developers alike can augment the AXIOM library, all using one common language. Library code, like interpreter code, is compiled into machine binary code for run-time efficiency.

Using this language, you can create new compuational types and new algorithmic packages. All library code is polymorphic, described in terms of a database of algebraic properties. By following the language protocols, there is an automatic, guaranteed interaction between your code and that of colleagues and system implementers.