Can the Eureqa Symbolic Regression Program, Computer Algebra, and Numerical Analysis Help Each Other?

David R. Stoutemyer

Eureqa is a symbolic regression program described by Schmidt and Lipson [11], freely downloadable from [13], where there are press citations, a bibliography of its use in articles, a blog, and a discussion group. In contrast to typical regression software, the user does not have to explicitly or implicitly provide a specific expression containing unknown constants for the software to determine. With little or no guidance, symbolic regression determines not only unknown coefficients but also the class and form of the model expression. See [9] for quick insight into the underlying "survival of the fittest" paradigm for symbolic regression. An Internet search on "symbolic regression" reveals that there are other such programs. However, my skeptic's curiosity was aroused by such extreme press praise as

> Move over, Einstein: Machines will take it from here. [10]
>
> There are very clever "thinking machines" in existence today, such as Watson, the IBM computer that conquered *Jeopardy!* last year. But next to Eureqa, Watson is merely a glorified search engine. [5]
>
> By one yardstick, Eureqa has even discovered the answer to the ultimate question of life, the universe and everything. [5]

The program is designed to work with *noisy experimental data,* searching for, then returning a set of result expressions that attempt to optimally trade off conciseness with accuracy. However, I was curious to learn if the program could also be used as a supplementary tool for *experimental mathematics*, *computer algebra*, and *numerical analysis*. In the first few weeks of using this tool, I have already found that:

1) Eureqa can sometimes do a job of exact simplification better than existing computer algebra systems if there is a concise equivalent expression.
2) Eureqa can often discover simple expressions that *approximate* more complicated expressions or fit a set of numerical results well.
3) Even when the fit isn't as accurate as desired, the *form* of a returned expression often suggests a class of forms to try for classic regressions, interpolations, or series expansions giving higher accuracy.
4) Eureqa could do an even better job if it supplemented its search with exploitation of more computer algebra and numerical methods, including classic regression.
5) There are important caveats about how to use Eureqa effectively.
6) The most extreme press quotes are exaggerations, but Eureqa really is quite impressive.

This article has examples that illustrate these findings, using Eureqa 0.93.1 beta.

Regarding computing times reported herein, the computer is a 1.60GHz Intel Core 2 Duo U9600 CPU with 3 gigabytes of RAM.

David Stoutemyer is a retired professor of information and computer science at the University of Hawaii. His email address is `dstout@hawaii.edu`.

DOI: http://dx.doi.org/10.1090/noti1000

Exact Simplification and Transformation

A Trigonometric Simplification Example

Here is a Maple 15 assignment of an input trigonometric expression to a dependent variable y:

$$\begin{aligned} y := {} & \cos(x)^3 \sin(x) + \frac{\cos(x)^3 \sin(x)}{2} \\ & + 2\cos(x)^3 \cos(2x) \sin(x) \\ & + \frac{\cos(x)^3 \cos(4x) \sin(x)}{2} \\ & - \frac{3}{2}\cos(x)\sin(x)^3 - 2\cos(x)\cos(2x)\sin(x)^3 \\ & - \frac{\cos(x)\cos(4x)\sin(x)^3}{2}. \end{aligned} \tag{1}$$

The default simplification merely combines the first two terms, which are similar.

However, the simplify(y) function[1] required only 0.03 seconds to return the much simpler

$$4\sin(x)\cos(x)^5\left(2\cos(x)^2 - 1\right). \tag{2}$$

An equivalent expression produced by the Mathematica FullSimplify[...] function in only 0.08 seconds is also much simpler:

$$2\left(\sin(3x) - \sin(x)\right)\cos^5(x).$$

But the equivalent even simpler form discovered by Eureqa is

$$y = \cos(x)^4 \sin(4x). \tag{3}$$

That even the simplified model of evolution in Eureqa can work so well might change some minds.

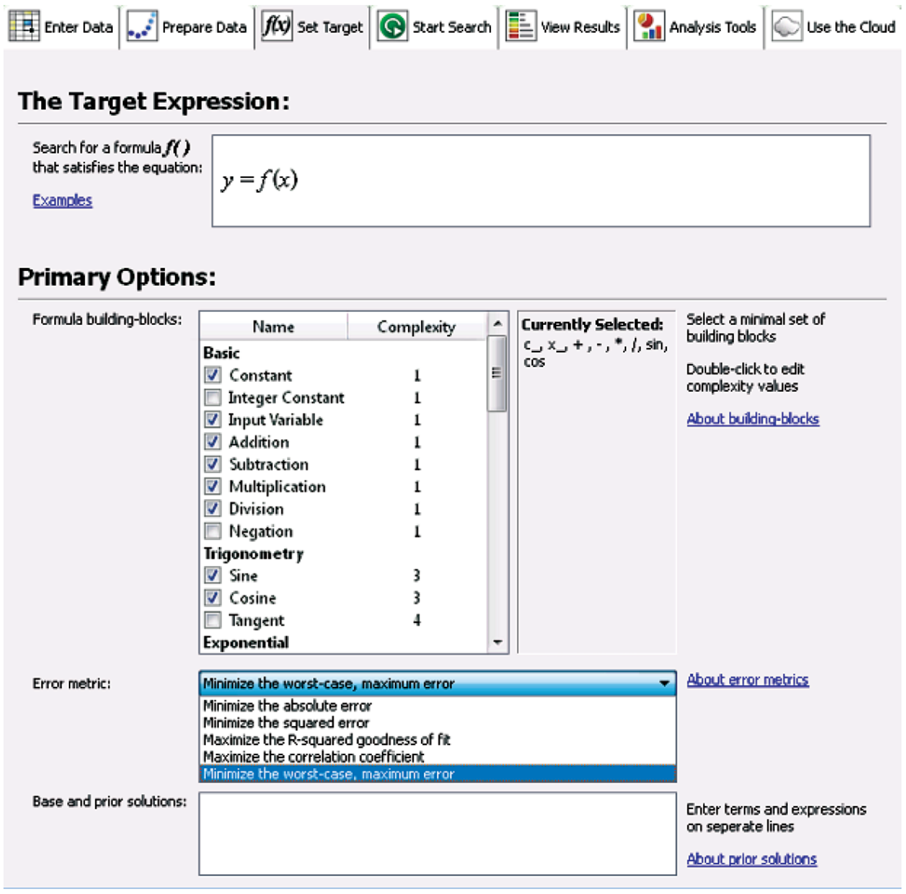
Caveat: The algorithm uses a random number generator with no current user control over its seeding, which appears to be done by the clock. Therefore sequences are not currently repeatable, and the computing times to obtain expression (3) varied dramatically, from 3 seconds to several minutes. It conceivably could require more time than you would ever be willing to invest. However, although even 3 seconds is much longer than the computer algebra times, it is certainly worth a few minutes wait to obtain such a nice result, and such experiments can be done while away from the computer, such as while eating, sleeping, or playing cell phone games. There is also an option to use parallel cloud computing, which reduces the mean and variance of the elapsed time necessary to obtain a satisfactory result.

Here is how I used Eureqa to obtain this delightfully simple exact result (3):

1) First I plotted expression (1) in Mathematica, revealing that it is antisymmetric and that its fundamental period appeared to be π, with higher frequency components appearing to have a minimum period of $\pi/4$. This was confirmed by TrigReduce[y], which returned the equivalent expression
$$\frac{1}{16}\left(\sin(2x) + 6\sin(4x) + 4\sin(6x) + \sin(8x)\right).$$
2) I decided to use evenly spaced values of x because expression (1) is periodic, bounded, and C^∞ for all real x. I guessed that it might help Eureqa discover and exploit the antisymmetry if I used sample points symmetric about $x = 0$. I also guessed that it might help Eureqa discover the periodicities if I used exactly two fundamental periods. I guessed that using 16 samples within the minimum period $\pi/4$ would be sufficient to resolve it quite well; then I doubled that because Eureqa uses some points for fitting and others for error assessment. This implies 128 intervals of width $\pi/64$ from $-\pi$ through π. I then created a table of 17-digit floating-point pairs of x and y, then exported it to a file by entering
$$\text{Export}\left[\text{“trigExample.csv”, Table}\left[\text{N}\left[\{x,\ldots\},17\right],\{x,-\pi,\pi,\frac{\pi}{128}\}\right]\right]$$
where the ellipsis was expression (1).[2]
3) I then launched Eureqa, opened its spreadsheet-like Enter Data tab, then replaced the default data there with mine by importing file trigExample.csv. In the row labeled var, I then entered x in column A and y in column B.
4) The Prepare Data tab then showed superimposed plots of the x and y data values and offered preprocessing options that aren't relevant for this very accurate data.
5) Figure 1 shows the Set Target tab:
(a) The pane labeled The Target Expression suggests the fitting model $y = f(x)$, which is what I want, so I didn't change it.
(b) The pane labeled Primary Options has check boxes for the desired Formula building-blocks used in composing candidate $f(x)$ expressions. The default checked ones are sufficient for the sort of concise equivalent

[1]simplify(...) *has an optional second keyword argument "size" that is intended to minimize size, but for this example the result is less concise than* (2).

[2]*The reason for requesting 17 significant digits was that I wanted Mathematica to use its adaptive significance arithmetic to give me results estimated to be accurate to 17 digits despite any catastrophic cancellations, and I wanted Eureqa to receive a 17th significant digit to help it round the sequences of input digit characters to the closest representable 16-digit IEEE double values, which are used by Eureqa.*



The Target Expression:

Search for a formula $f()$ that satisfies the equation:

Examples

$y = f(x)$

Primary Options:

Formula building-blocks:

Name	Complexity
Basic	
✓ Constant	1
☐ Integer Constant	1
✓ Input Variable	1
✓ Addition	1
✓ Subtraction	1
✓ Multiplication	1
✓ Division	1
☐ Negation	1
Trigonometry	
✓ Sine	3
✓ Cosine	3
☐ Tangent	4
Exponential	

Currently Selected: c_, x_, +, -, *, /, sin, cos

Select a minimal set of building blocks

Double-click to edit complexity values

About building-blocks

Error metric: Minimize the worst-case, maximum error

Minimize the absolute error
Minimize the squared error
Maximize the R-squared goodness of fit
Maximize the correlation coefficient
Minimize the worst-case, maximum error

About error metrics

Base and prior solutions:

Enter terms and expressions on seperate lines

About prior solutions

Figure 1. Eureqa Set Target tab for exact trigonometric simplification example.

to expression (1) that I am seeking. Therefore I didn't check any of the other offered functions and operators, not all of which are shown on this screen shot.[3] Formula building blocks also shows the corresponding Complexity measures, which can be altered by the user. The complexity measure of an expression is the sum of the complexities of its parts.

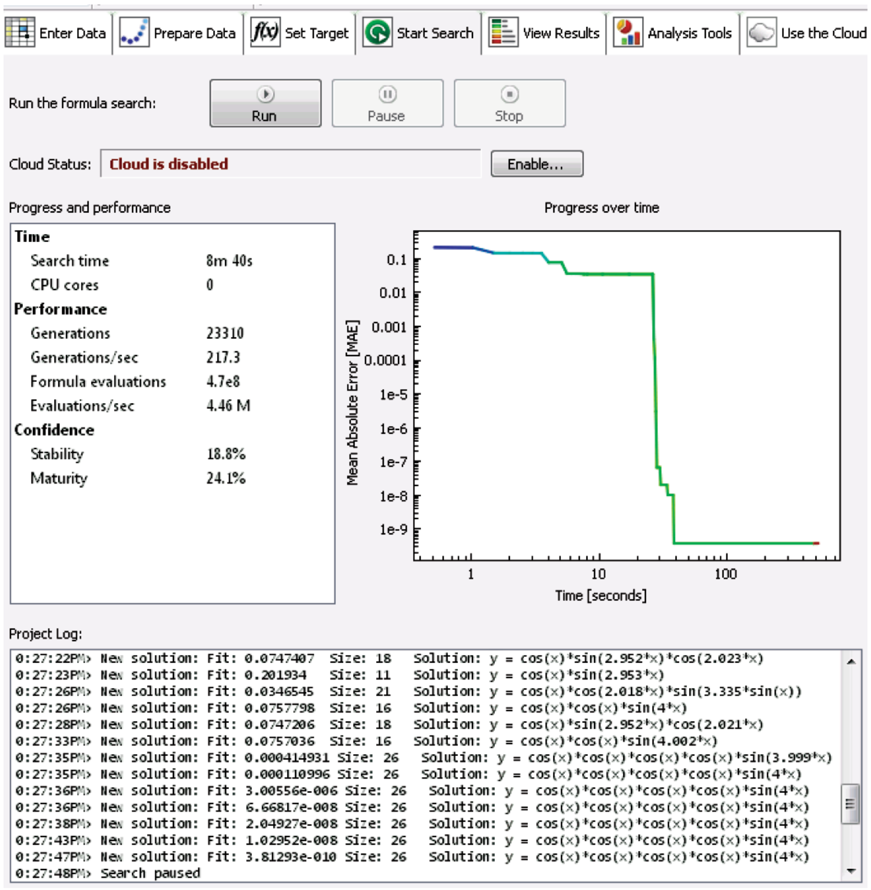
(c) The drop-down menu labeled Error metric offers different built-in error measures for Eureqa to try optimizing. A bound is usually more reassuring than the alternatives, so I chose Minimize the worst-case maximum error. The documentation suggests that Maximize the R-squared goodness of fit or Maximize the correlation coefficient is more scale and mean invariant, which are desirable properties too. However, the data is already well scaled with means of 0.

(d) For the covered Data Splitting drop-down menu the default alternative is to designate a certain percentage of the data points for fitting the data (*training*) and a certain percentage for assessing the error measure (*validation*). The individual assignments to these categories are done randomly, with some overlap if there aren't many data points. For the almost exact data in this article, I would prefer that alternative points be assigned to alternative categories, with the end points being used for training. However, none of the alternatives offered this, so I chose the default.[4]

6) I then pressed the Run button on the Start Search tab and watched the temporal progress of the search. The plot in

[3] *I could have saved search time by unchecking Division, because the floating-point coefficients make a denominator unnecessary for this class of expressions. I could also have saved search time and perhaps obtained a more accurate result by checking the Integer Constant box, because integer coefficients are quite likely for exact equivalent expressions, making it worth having Eureqa try rounding to see if that improves the accuracy. However, I decided to accept the default checked boxes to see if Eureqa could find a good exact equivalent without any more help from me. Other building blocks are described at* [6].

[4] *I have since learned that, with such precise data, another promising alternative would be to use all of the points for both training and validation, which is a current option.*



Time	
Search time	8m 40s
CPU cores	0
Performance	
Generations	23310
Generations/sec	217.3
Formula evaluations	4.7e8
Evaluations/sec	4.46 M
Confidence	
Stability	18.8%
Maturity	24.1%

```
0:27:22PM> New solution: Fit: 0.0747407  Size: 18   Solution: y = cos(x)*sin(2.952*x)*cos(2.023*x)
0:27:23PM> New solution: Fit: 0.201934   Size: 11   Solution: y = cos(x)*sin(2.953*x)
0:27:26PM> New solution: Fit: 0.0346545  Size: 21   Solution: y = cos(x)*cos(2.018*x)*sin(3.335*sin(x))
0:27:26PM> New solution: Fit: 0.0757798  Size: 16   Solution: y = cos(x)*cos(x)*sin(4*x)
0:27:28PM> New solution: Fit: 0.0747206  Size: 18   Solution: y = cos(x)*sin(2.952*x)*cos(2.021*x)
0:27:33PM> New solution: Fit: 0.0757036  Size: 16   Solution: y = cos(x)*cos(x)*sin(4.002*x)
0:27:35PM> New solution: Fit: 0.000414931 Size: 26   Solution: y = cos(x)*cos(x)*cos(x)*cos(x)*sin(3.999*x)
0:27:35PM> New solution: Fit: 0.000110996 Size: 26   Solution: y = cos(x)*cos(x)*cos(x)*cos(x)*sin(4*x)
0:27:36PM> New solution: Fit: 3.00556e-006 Size: 26   Solution: y = cos(x)*cos(x)*cos(x)*cos(x)*sin(4*x)
0:27:36PM> New solution: Fit: 6.66817e-008 Size: 26   Solution: y = cos(x)*cos(x)*cos(x)*cos(x)*sin(4*x)
0:27:38PM> New solution: Fit: 2.04927e-008 Size: 26   Solution: y = cos(x)*cos(x)*cos(x)*cos(x)*sin(4*x)
0:27:43PM> New solution: Fit: 1.02952e-008 Size: 26   Solution: y = cos(x)*cos(x)*cos(x)*cos(x)*sin(4*x)
0:27:47PM> New solution: Fit: 3.81293e-010 Size: 26   Solution: y = cos(x)*cos(x)*cos(x)*cos(x)*sin(4*x)
0:27:48PM> Search paused
```

Figure 2. Eureqa Start Search tab for exact trigonometric simplification example.

Figure 2 dynamically zoomed out to show a log-log plot of the best error measure obtained so far as a function of computing time. The Project Log dynamically showed successive candidates that are better than any so far on the basis of either complexity or accuracy. The entertainment of watching the evolution of these panes is very appealing. It is similar to rooting for the home team at a sporting event. Notice that:

(a) Integer powers of subexpressions are represented in the Project Log as repeated multiplications, and their complexity is measured that way. For example, several results are displayed as

$$y = \cos(x) * \cos(x) * \cos(x) * \cos(x) * \sin(4 * x) \tag{4}$$

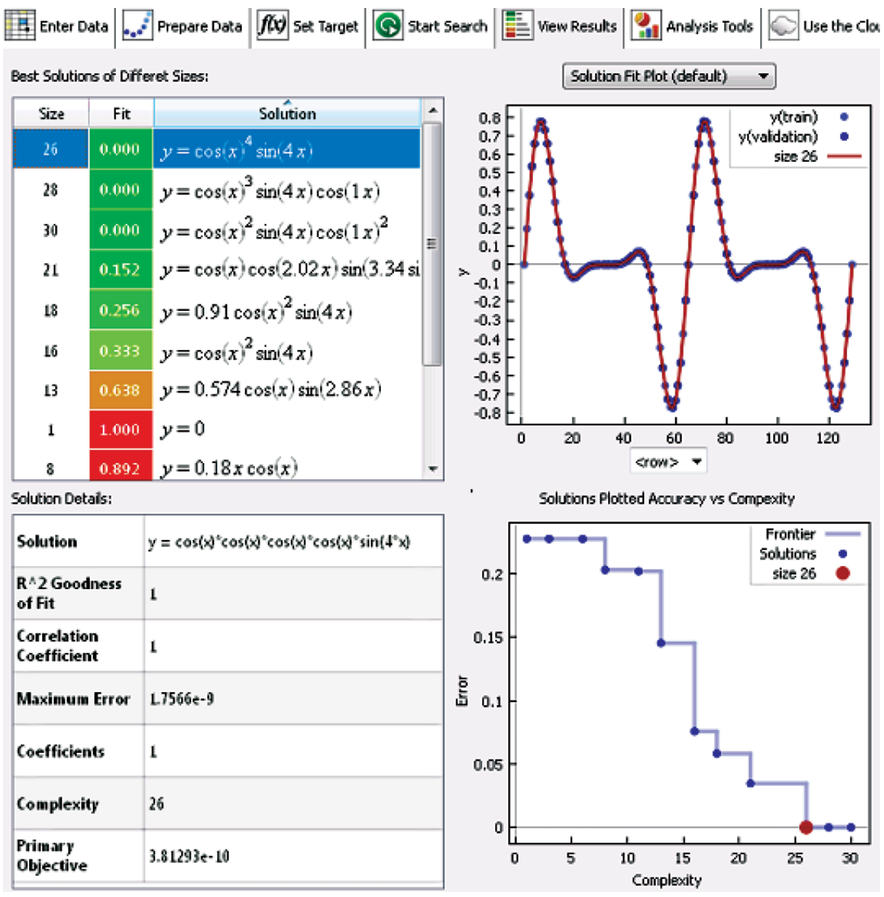
with a complexity measure of 26.[5]

(b) Result (4) appears first with a reported Fit of 0.000111, followed by the same expression with monotonically better fits up through 3.81×10^{-10} at 39 seconds.[6] I terminated the

[5] *The complexity measure seems less than ideal: the ultimately reported formula in the* View Results *tab of Figure 3 is* $\cos(x)^4 \sin(4x)$*. This is more concise and requires only one cosine and two multiplies to compute the* $\cos(x) * \cos(x) * \cos(x) * \cos(x)$ *factor if compiled by any decent compiler.*

[6] *The same displayed formula was associated with such dramatically different error measures because Eureqa always rounds its displayed coefficients to about four significant digits, and in this case it enabled display of an exact result to which it was merely converging. If I had checked the* Integer Constant *building block, then Eureqa probably would have rounded the coefficients to integers. However,* $y(0.0) = 0.0$ *and the data varies between about* ± 3.8*, so a maximum residual of* 3.81×10^{-10} *is enough to convince most people that the displayed result is exact. Nonetheless, the result expression might not be equivalent to the sampled expression everywhere between samples even if all the residuals are* 0.0*. Also, one or more of those floating-point zeros might be caused by underflow of numbers whose exact values are nonzero. For this example we can prove equivalence for all complex* x *because*

$$\mathrm{TrigReduce}\left[\left(\cos(x)^3 \sin(x) + \frac{\cos(x)^3 \sin(x)}{2} + \cdots\right) - \mathrm{Cos}[x]^4\,\mathrm{Sin}[4x]\right] \to 0\,.$$



Best Solutions of Differet Sizes:

Size	Fit	Solution
26	0.000	$y = \cos(x)^4 \sin(4x)$
28	0.000	$y = \cos(x)^3 \sin(4x) \cos(1x)$
30	0.000	$y = \cos(x)^2 \sin(4x) \cos(1x)^2$
21	0.152	$y = \cos(x) \cos(2.02x) \sin(3.34$ si
18	0.256	$y = 0.91 \cos(x)^2 \sin(4x)$
16	0.333	$y = \cos(x)^2 \sin(4x)$
13	0.638	$y = 0.574 \cos(x) \sin(2.86x)$
1	1.000	$y = 0$
8	0.892	$y = 0.18x \cos(x)$

Solution Details:

Solution	y = cos(x)*cos(x)*cos(x)*cos(x)*sin(4*x)
R^2 Goodness of Fit	1
Correlation Coefficient	1
Maximum Error	1.7566e-9
Coefficients	1
Complexity	26
Primary Objective	3.81293e-10

Figure 3. Eureqa View Results tab for exact trigonometric simplification example.

search at 8 minutes, during which Eureqa tried different formulas that didn't fit as well, regardless of how complicated they were.

7) Figure 3 shows the results on the View Results tab:

(a) In the pane labeled Best Solutions of Different Sizes, the first column lists the complexity measure, the second column lists an error measure, and the third column lists the corresponding candidate equation. If you click on a row, the pane in the lower left corner gives more detail, with various rounded error measures, including the Primary Objective that I chose, which was maximum absolute error.

(b) Eureqa does some automatic expression simplification: Whenever a *mutation* produces a new candidate expression or a *crossover* produces two new candidates containing mixtures of the two parents' subexpressions, a few transformations are applied to make the expression more nearly canonical and to avoid having a misleadingly high complexity on account of uncollected similar terms, uncombined numeric subexpressions, etc. This minimal computer algebra is also used to make the displayed results in the Best solutions of different sizes pane more attractive. These transformations include:

- Integer powers and products of sums are expanded to make them more nearly canonical.[7]
- Factors and terms are sorted; then similar factors and terms are collected.
- Numeric subexpressions are reduced to a single number.
- A few rules such as $\text{abs}(\text{abs}(u)) \rightarrow \text{abs}(u)$ and $1 * u \rightarrow u$ are applied.

However, you will often see a sub-expression such as $1\cos(1x)$. This is because the 1's are a result of rounding noninteger coefficients to the typically displayed 3 or 4 significant digits. Nonetheless, I noticed unexploited opportunities, such as Eureqa unnecessarily trying the sub-expression $\sin(x + 6.283)$. Exploiting symmetry and angle reduction could

[7] *Unfortunately, expanded forms are often less concise than equivalent factored forms that would have smaller Complexity.*

preclude the need to try candidates having constant terms outside a much smaller interval.

(c) The pane in the lower right corner plots an error measure as a function of complexity for the twelve reported optimal candidates, with the currently highlighted candidate as a larger red dot.

(d) The pane in the upper right corner shows a plot of the highlighted expression superimposed on the validation points in dark blue and the training points in lighter blue. The drop-down menu above it offers the option of instead plotting the *residuals* at all of the data points, which gives a much better idea of the fitting errors as a function of x. If these residuals had revealed a nonnegligible recognizable pattern such as being approximately proportional to $\sin(16x)$, then I would have tried another search with the model $y = f(\cos(x), \sin(4x), \sin(16x))$. I would iterate this process until there was no further improvement or the residuals revealed no nonnegligible recognizable pattern.

Instead of searching for the most concise equivalent, one can also search for equivalent expressions of a *particular form*. For example, we could request that expression (1) be transformed into a function without cosines by unchecking that building block. As another example, the request $z = f_1(x) * f_2(y)$ can be used to search for an equivalent form having separated variables. (Nonnegative integer suffixes on variable and function names are pretty-printed as subscripts.)

Some Other Exact Simplification Examples

Using "Assuming[..., FullSimplify[...]]" in Mathematica and "simplify(...), assuming ..." in Maple, I could not make them accomplish the following quickly successful Eureqa simplifications, for which I selected the default building blocks, Integer Constants, and other functions or operators that occur in the specific given expressions:

$$(5) \qquad \left\lfloor \sqrt{\lfloor x \rfloor} \right\rfloor + \lfloor \sqrt{x} \rfloor \mid x \geq 0 \to 2 \lfloor \sqrt{x} \rfloor .$$

$$(6) \qquad q^2 + \frac{2^{1/3}\left(\sqrt{81q^2-12}-9q\right)^{2/3}+24^{1/3}}{6^{2/3}\left(\sqrt{81q^2-12}-9q\right)^{1/3}} - \frac{2\cos\left(\frac{1}{3}\operatorname{acos}\left(-\frac{3\sqrt{3}q}{2}\right)\right)}{\sqrt{3}} \quad \mid \; |q| < \frac{2}{3\sqrt{3}} \to q^2.$$

$$(7) \qquad \max(x-y,\,0) - \max(y-x,\,0) \to x - y.$$

- Example (5) comes from [7].
- I tabulated only the *real* part for example (6), because rounding errors for the principal branch of the fractional powers of the negative quantity $\sqrt{81q^2-12} - 9q$ generated some relatively very small magnitude imaginary parts.
- To generate $17^2 \to 289$ rows of data for example (7), I used

$$\text{Apply[Join,Table[N[}\{x, y, \text{Max}[x - \frac{2}{3\sqrt{3}}y, 0] \\ -\text{Max}[y-x, 0]\}, 17], \{x, -1, 1, \frac{1}{8}\}, \{y, -1, 1, \frac{1}{8}\}]].$$

The examples so far illustrate that Eureqa can sometimes determine a simpler exact result than a computer algebra system. However, I constructed these very simple results, then transformed subexpressions in ways that I suspected would be difficult for common transformations to reverse, particularly if applied to the entire expression rather than well-chosen pieces. Despite this it was not easy to find examples for which neither FullSimplify[...] nor simplify(...) could determine the very simple equivalent. Thus, Eureqa should be regarded as an *occasionally* beneficial *supplement* to these functions rather than as a replacement. Also, effective use of Eureqa requires good judgment in:

- the interval spanned by the sample points, their number and spacing;
- the form selected for the target expression; and
- the building blocks that are selected together with their complexities.

Effective use also probably depends on experience with Eureqa. Therefore, the benefits that I have discovered should be regarded as a lower bound because of my novice status.

How to Reduce the Curse of Dimensionality

As illustrated by example (7), data for representing all combinations of n different values for each of m different independent variables requires n^m rows, and we must have $n \geq 2$ for Eureqa to discern any nonconstant dependence on each variable. Thus, for a given n, this exponential growth in data rows with the number of independent variables can greatly increase the time required for Eureqa to find a good fit. Moreover, yielding to the consequent temptation to reduce computation time by reducing n as m increases tends to reduce the precision of the results. Fortunately, there are several complementary techniques that sometimes reduce the effective dimensionality:

1) *Dimensional analysis* can sometimes reduce the number of independent variables, and [15] describes a Maxima program that automates this.

2) If you can partition an expanded or factored form of your expression into two or more components having distinct variable sets, then you can independently fit each of those components. For example, you can separately fit $\max(x-y,0)-\max(y-x,0)$ and $\max(y-z,0)-\max(z-y,0)$ in the sum of these two expressions, converting a trivariate example into two bivariate examples.
3) If in every term of a given sum the exponents of some subset of the variables sum to the same homogeneity exponent k, then substitute 1 for one of those variables v, fit this isomorphic problem, then multiply every resulting term by the individual power of v necessary to restore the homogeneity.
4) Unlike many experimental situations, for experimental mathematics we usually have complete freedom to choose the data values of the independent variables. In both univariate and multivariate problems there can be *sweet spots* that deliver more accuracy for a given number of samples than does a brute force Cartesian product of uniformly spaced points. These special values are often related to the zeros or extrema of orthogonal polynomials. For example, see the multidimensional integral formulas in Chapter 25 of [1].

Approximate Symbolic Results

Ideally a floating point result is either exact or the closest representable number to the exact result, with ties broken in favor of having the last significant bit be 0. Well-designed floating-point *arithmetic* comes very close to this ideal: If the exact result is representable within the thresholds for overflow and for gradual underflow, then the result is the exact result for inputs that differ from the actual inputs by factors between $1-\varepsilon_m$ and $1+\varepsilon_m$, where *machine epsilon* ε_m is about 1.11×10^{-16} for IEEE double, which corresponds to about 16 significant digits. A *relative* error bound of $\pm\varepsilon_m$ is the *gold standard*. There are similar expectations for operations such as exponentiation and for functions such as sinusoids or logarithms that are commonly built into compilers but allowing a few times ε_m, which I call the *silver standard*. It is a matter of professional pride among numerical analysts designing *general purpose* mathematical software to strive for the silver standard for all functions provided with the software, and many users assume this.[8] I was curious to know if Eureqa can help numerical analysis by discovering concise approximate expressions that meet the silver standard.

[8] *For example, early Microsoft Basic computed elementary functions, fractional powers, and integer powers to half the precision of the other arithmetic operators, causing unsuspecting users to have results that were often far less accurate than assumed.*

An Antiderivative Example

Many well-known and special-purpose functions are defined by a definite integral containing a variable that is not the integration variable, a definite integral for which there is no known closed-form expression in terms of simpler known functions. For example, suppose that I want to implement an IEEE double version of the Dogbert W function defined by

(8)
$$W(x) := \int_0^x \frac{dt}{\sqrt{1-t^2+\frac{2}{\pi}\cos\left(\frac{\pi t}{2}\right)}} \quad \mid -1 \le x \le 1\,.$$

No computer algebra systems that I tried can determine a closed form for $W(x)$.

Plan B: Do a gold standard numeric integration for a sequence of x values in the interval $-1 \le x \le 1$, then use Eureqa to discover a silver standard expression that fits those values well. Here is an abridged account of my attempt to do this.

1) I entered the Mathematica input

$$\begin{aligned}&\text{xWPairs} = \text{Table}[\{\text{N}[x,17],\\ &\text{NIntegrate}[\frac{1}{\sqrt{(1-t)(1+t)+\frac{2}{\pi}\cos\left(\frac{\pi t}{2}\right)}},\\ &\{t,0,x\},\text{PrecisionGoal}\to 17,\\ &\text{WorkingPrecision}\to 25],\{x,-1,1,1/64\}].\end{aligned} \quad (9)$$

Then I inspected the 129 number pairs to make sure there were no imaginary parts, infinities, or undefined values.[9]

[9]
- *The data in the* Enter Data *tab must currently be integers or finite real floats. A fraction, a floating point infinity, a nan, or a nonreal number is not accepted. If an imaginary part of your data is negligible noise, then replace it with* 0.0*. Otherwise you must separately fit the real and imaginary parts or the absolute value and the arg, which is the* Two-Argument Arctangent *in Eureqa. If your data has an undefined value due to an indeterminate form, then replace it with the value returned by your computer algebra* limit(...) *function. If your table has an infinity, then subtract out or divide out the singularity that is causing it. Don't expect a good fit if you simply omit a value that has infinite magnitude or replace it with an exceptionally large magnitude having the correct sign, either of which can dramatically mislead the search.*
- *I entered $1-t^2$ as $(1-t)(1+t)$ in input (9) and elsewhere to reduce the magnification of rounding errors by catastrophic cancellation near $x=\pm1$.*
- *The reason for WorkingPrecision → 25 is that I wanted to further reduce this catastrophic cancellation.*

2) All the values were real and finite, so I then entered

   Export ["xWPairs.csv", xyPairs] ;

   then I imported the resulting file into Eureqa and tried to fit $W = f(x)$ with the default formula building blocks. After three minutes of search the most accurate formula was

$$W = \frac{0.7773112536\,x}{\cos(0.4696613973\,x^7) - 0.2294845216\,x^2},$$

   which is noninsightful and accurate to a disappointing maximum absolute error of about 0.4%.

3) A Mathematica plot of the *integrand* reveals a probable explanation: The integrand has endpoint singularities, and although they are integrable, the resulting integral has infinite magnitude low-order derivatives at the endpoints, which probably makes the quadrature less accurate and the Eureqa search slower than otherwise.

4) The plot of numeric *antiderivative* values on the Prepare Data tab revealed that it looks approximately proportional to $\operatorname{asin}(x)$.

5) So next I did another Eureqa search after disabling the sine and cosine building blocks and making the target expression be $W = f(x, \operatorname{asin}(x))$. I did *not* enable the arcsine building block because I didn't want Eureqa to waste time trying arguments that cause $\operatorname{asin}(\ldots)$ to be nonreal. In 11 seconds Eureqa found the following seven-term expression having a maximum absolute error of only about 3.1×10^{-9}:

$$\begin{aligned} W = &-1.870027576 \times 10^{-13} \\ &+ 0.7816747744x \\ &+ 0.0147770774\,x^3 \\ &- 0.03033616234\,x^2 \operatorname{asin}(x) \\ &+ 0.07586494202\,x \operatorname{asin}(x)^2 \\ &+ 0.0818165982 \operatorname{asin}(x)^3 \\ &+ 0.0009144579166\,x^3 \operatorname{asin}(x)^2. \end{aligned} \tag{10}$$

   I aborted the search at 3 minutes with no further improvement. Regarding this result:

   (a) To view the coefficients rounded to 10 significant digits rather than about 4, I had to copy the expression from the Best solutions of different sizes pane in the View Results tab into Mathematica or a text editor. Unfortunately there is currently no way to view or access all 16 digits, so I will have to polish a result with classic regression software to obtain a silver standard result.[10]

   (b) I edited the copied formula because it contained annoying superfluous parentheses and represented integer powers by repeated multiplication.

   (c) The even symmetry of the integrand and the centered integration from 0 imply that the antiderivative should have odd symmetry. Therefore the spurious constant term in expression (10) should be discarded. Terms that are contrary to odd symmetry might arise from the random partitioning of the data into training and validation sets.

   (d) Converting the result to recursive form by collecting similar powers of x or $\operatorname{asin}(x)$ can significantly reduce the Complexity. For example, deleting the spurious constant term, rounding the coefficients in result (10) for brevity, then collecting with respect to $\operatorname{asin}(x)^2$ give

$$\begin{aligned} W = &\ 0.78\,x + 0.015\,x^3 - 0.03\,x^2 \operatorname{asin}(x) \\ &+ (0.076\,x + 0.00091\,x^3) \operatorname{asin}(x)^2 \\ &+ 0.082 \operatorname{asin}(x)^3, \end{aligned}$$

   which saves one instance of $*\operatorname{asin}(x)^2$. At the expense of comprehensibility, the Complexity and the floating-point substitution time can be further reduced by *Hornerizing* this recursive representation to

$$\begin{aligned} W = x(0.78 + 0.015x^2) + \operatorname{asin}(x)(-0.03\,x^2 \\ + (x\,(0.076 + 0.00091\,x^2) \\ + 0.082 \operatorname{asin}(x)) \operatorname{asin}(x))). \end{aligned}$$

   This is another place where more computer algebra could help Eureqa.

   (e) Notice that result (10) has no term that is simply a numeric multiple of $\operatorname{asin}(x)$. Therefore it doesn't model the infinite endpoint slopes associated with the integrand singularities there. But that is my fault, because the target expression contains no special encouragement to include a term of that form, and the plot from the Eureqa Prepare Data tab reveals that the sample points were not closely enough spaced near the endpoints to suggest the infinite slope magnitudes there.

[10] *The rounding of coefficients in the Project Log and Best solutions of different sizes panes does increase comprehensibility, particularly since those panes are regrettably unscrollable. Even more rounding could justifiably be done for coefficients of terms whose relative maximum contribution is small. For example, if the maximum contribution of a term over all data points is 1%, then its coefficient justifiably could be rounded to two less significant digits than the coefficient that contributes the most over all data points. See* [3] *for more about this idea.*

6) Result (10) suggests that a set of particularly good fits would be a truncated series of the form

$$(c_1 \operatorname{asin}(x) + c_2 x) + \left(c_3 \operatorname{asin}(x)^3 + c_4 x \operatorname{asin}(x)^2 + c_5 x^2 \operatorname{asin}(x) + c_6 x^3\right) + \cdots$$

with numeric coefficients c_k. This can be regarded as a truncated bivariate series in terms having a power of x times a power of arcsin(x) that have nondecreasing odd total degrees. I could constrain Eureqa to search only within this class of expressions by entering a target expression having a particular total degree, such as

$$\begin{aligned} W = {} & f_1()\operatorname{asin}(x) + f_2()\,x \\ & + f_3()\operatorname{asin}(x)^3 + f_4()\,x\operatorname{asin}(x)^2 \\ & + f_5()\,x^2\operatorname{asin}(x) + f_6()\,x^3. \end{aligned}$$

However, standard regression software is much faster for merely optimizing some parameters in a specific form/and more accurate if done with arbitrary-precision arithmetic. The largest total degree in Eureqa result (10) is 5. Consequently, I next used the Mathematica LinearModelFit [...] function with the 12 basis expressions of the form $x^j \operatorname{asin}(x)^k$ having odd total degree $1 \le j + k \le 5$. After 0.27 seconds I received the following eight-term result, whose significant digits I have truncated for brevity:

$$\begin{aligned} & 22.9\,x + 0.012\,x^3 + 0.000097\,x^5 \\ & - 22.1\operatorname{asin}(x) - 0.068\,x^2\operatorname{asin}(x) \\ & + 0.78\,x\operatorname{asin}(x)^2 + 3.09\operatorname{asin}(x)^3 \\ & - 0.078\operatorname{asin}(x)^5. \end{aligned} \tag{11}$$

The maximum absolute error at the tabulated points was about 4×10^{-11}, which is two significant digits more than (10) for the same total degree and only one more term. The 96 percent cancellation between the dominant terms $22.9\,x$ and $-22.1\operatorname{asin}(x)$ for x near 0 is less than ideal, but the small error measure is very gratifying.

Encouraged, I next tried LinearModelFit [...] with total degree 7, which entails twenty basis terms. However, the most accurate returned result *also* had eight terms and cancellation between the two dominant terms. Moreover, the error measure was only slightly smaller, and the coefficients changed dramatically from those of corresponding terms in (11). Therefore it seems likely that something important is missing from the basis, preventing results from corresponding to economized truncations of a convergent infinite series.

Thus it is not promising to try larger total degrees with this basis in pursuit of a concise numerically stable silver standard. Nonetheless, this has been a nice combined use of Eureqa and Mathematica: After the decision to try a target expression of the form $f(x, \operatorname{asin}(x))$, Eureqa discovered a concise form having 9 significant digits and basis functions of the form $x^j \operatorname{asin}(x)$. I then used Mathematica to obtain a result having only one more term that is accurate to 11 significant digits, which is more than adequate for many purposes. For this example Eureqa has served as a *muse* rather than as an *oracle*.

However, I couldn't help wondering: *Is* there a relatively simple class of forms that Eureqa might have revealed for further polishing to a concise *silver standard* fit by Mathematica?

With the help of the Mathematica Series [...] function and an embarrassingly large amount of my time adapting its results to my desires via a sixteen line *Mathematica* function downloadable from a website [16], I was able to determine an exactly integrable truncated infinite series expansion of the integrand that converged sufficiently fast without undue catastrophic cancellation over the entire interval $-1 \le x \le 1$. Using Assuming$[-1 \le x \le 1$, Integrate[...]], the Hornerized representation of the corresponding *antiderivative* is

(12)

$$\begin{aligned} \tilde{W}(x) := {} & c_0 \operatorname{asin}(x) \\ & + x\sqrt{(1-x)(1+x)}\left(c_1 + x^2\left(c_2 + x^2(c_3 + \cdots)\right)\right). \end{aligned}$$

Being a single series bifocally expanded about $x = \pm 1$, it is C^∞ over the entire interior interval $-1 < x < 1$. Notice that, contrary to the previous efforts, asin(x) occurs only to the first power and that all the other terms are multiplied by $\sqrt{(1-x)(1+x)}$. A good set of basis expressions is thus $\{\operatorname{asin}(x),\ x^{1+2k}\sqrt{(1-x)(1+x)}\}$ for $k = 0, 1, \ldots$. The higher powers of asin(x) in the previous basis were being used as flawed surrogates for $x^{1+2k}\sqrt{(1-x)(1+x)}$: octagonal pegs in circumscribed round holes—they sort of fit, but in an impaired way.

The Mathematica function that I wrote computes the exact coefficients c_k, which involve $\sqrt{6}$ and powers of π. After writing that function I used LinearModelFit [...] to effectively economize a higher-degree approximation to a lower-degree one having nearly the same precision. For the data I used high-precision numerical integration after using truncated series (12) to subtract out and exactly integrate two terms of the endpoint singularities. This made equally spaced x values quite acceptable. Experiments revealed that using terms through coefficient c_7 gave an approximation to $y(x)$ that had a worst absolute error of about

6×10^{-16} at the right endpoint, where the value was about 1.255. A plot of the residuals revealed that this eight-coefficient approximation has a relative error of about 5×10^{-16}, making it silver standard. Moreover, this plot was essentially noise, revealing no remaining exploitable residual for IEEE double.

Eureqa did not by itself discover this better basis, but that is my fault: I never allowed square roots as a building block. I should have done a little analysis earlier to wisely suggest a target expression of the form $f(x, \operatorname{asin}(x), \sqrt{(1-x)(1+x)})$. With a square root in the integrand denominator, I should have anticipated a square root in a good approximate antiderivative numerator. Better yet, I could have tried the square root that also contains the cosine term. Using Eureqa effectively for computer algebra and numerical analysis should be viewed as a *collaboration* rather than as a *competition*. And Eureqa is not a black box oracle. You cannot leave your brain behind.

Other Possible Approximate Symbolic Results

There are many other possibilities for using Eureqa to find a concise expression or to suggest a good class of expressions that closely approximates a result that would otherwise have to be presented only graphically or as a table. For example,

- *Inverse functions and solving parametric algebraic equations*: Suppose that for the example (8) we also or instead wanted an approximate symbolic expression for the *inverse* Dogbert W function: x as a function of W. Eureqa can search for such expressions if we simply enter $x = f(W)$ or, more efficiently for this example, $x = \sin(f_2()\,W) + f_3(W)$.
- *Solution of differential, integral, delay, and other kinds of functional equations*: For example, suppose that we have a system of first-order ordinary differential equations and a tabulated numerical solution vector $[y_1(t), y_2(t), \ldots]$ for $t = t_0, t_1, \ldots, t_n$. Then we can use Eureqa to separately search for good expressions for $y_1(t), y_2(t), \ldots$. If we want $t_n = \infty$, then we should first make a change of independent variable to map that endpoint to a finite value in a way that doesn't introduce a singularity elsewhere.
- *Implicitization*: A major application of computer algebra is implicitization of parametric equations defining curves, surfaces, or higher-dimensional manifolds. Implicit representations are better than parametric for determining whether a point is inside, outside, or on a closed manifold. However, exact implicitization algorithms are known only for certain classes of parametric equations. For other classes we can try using Eureqa to find approximate implicit equations. (See [12] for tips on how to use Eureqa for this purpose.)

Additional Tips and Tricks

Here are a few additional tips and tricks for using Eureqa effectively in conjunction with computer algebra or numerical analysis:

1) To increase the chance of obtaining exact rational coefficients, reduce your input expression over a common denominator, then separately tabulate and fit the numerator and denominator with the Integer Constant building block checked.
2) To possibly include *foreseen* exact symbolic known irrational constants such as π and $\sqrt{2}$ in your result, for each such constant make a labeled column of corresponding floating-point values, preferably accurate to 16 or 17 digits. For example if a column labeled *pi* contains 3.1415926535897932 and a column labeled *sqrt*2 contains 1.414213562373095, then check the Integer Constant building block and use a target expression of a form such as "$\ldots = f(pi, sqrt2, \ldots)$". These symbolic constants don't contribute much to the curse of dimensionality because they are invariant, so there is no need to include more rows on their account.
3) To see if an unrecognizable floating-point constant in a result is a close approximation to some *unknown relatively simple rational or irrational constant* such as $3/7$ or $\sqrt{7}\pi$, enter all of the computed digits into the remarkable free online Inverse Symbolic Calculator [2] or use the Maple identify(...) function, which is an earlier version thereof.[11]
4) As with most regression software, Eureqa tends to express results in terms of the notoriously ill-conditioned monomial basis $1, x, x^2, x^3, \ldots$. To express your result more accurately in terms of an orthogonal polynomial basis, such as the Chebychev T polynomials, generate columns labeled $T_0, T_1, \ldots, T_n$, the desired constant n at the tabulated values of the independent variables. After 1 in the column labeled T_0 and the sample values of x in the column labeled T_1, successive columns can be computed from these by entering $= 2 * x * T_1 - T_0$ in the column labeled T_2, then $= 2 * x * T_2 - T_1$ in the column labeled T_3, etc. You can then use a target

[11] *Inverse Symbolic Calculator can be regarded as symbolic regression of a single floating-point number to an exact numeric expression that it approximates well. It would be nice if Eureqa included building blocks for at least simple rational constants, quadratic numbers, and multiples of π or e, which is easy to implement.*

expression of the form "$\ldots = f(T_0, T_1, \ldots)$". The T_k variables don't contribute much to the curse of dimensionality because they are totally correlated rather than truly independent. Moreover, these variables have the same Complexity as any other variable, thus preventing them from being discriminated against as they would be if you otherwise equivalently made the target expression be "$\ldots = f(1, x, x^2 - 1, x^3 - 3x, \ldots)$".

Seamless Integration of Software Packages

It is usually somewhat of a nuisance to work between two or more separate software packages compared to working entirely within one that has all of the necessary features built in. The extra effort and the lesser chance of even knowing about two or more complementary software packages is a great deterrent to such use. Thus it is good to know that interfaces are under development for using Eureqa from within Mathematica, MatLab, Python, .Net, and Knime, with current versions downloadable from [13]. Such added interfaces are rarely as seamless as features that are built in, but they are often great improvements over communicating by export-import of files or by copy and paste. Among other things, the interfaces can accommodate differences in syntax for expressions.

There is also a different symbolic regression add-on Gptips for MatLab [8] and one named DataModeler for Mathematica [4].

The Need for Automation

At any one time, most of us are amateurs with most of the software that we use. We need all of the help we can obtain. In response to that need, many of the best software packages automate certain portions. However, the previous examples were *not* automatic. Training, experience, and judgment were involved in choosing the data points, target expression, etc. Some of this could and should be automated. A qualitative analysis program such as the Maxima program described in [17] could help in this regard: Given an expression, this program attempts to return bounds and indications of monotonicity, convexity, symmetries, periodicities, zeros, singularities, limits, and stationary points. To the extent that this is successful, this information could be used to choose automatically the function blocks and for each independent variable the endpoints, the number of samples, and perhaps even their distribution. For example:

- If there are poles, then it is best to automatically convert any tangents to sines divided by cosines, then form a reduced common denominator, then use Eureqa to fit separately the numerator and denominator. Neither the numerator nor the denominator will contain a *pole*, but they still could contain a logarithmic singularity that would have to be handled by subtracting or dividing it out.
- It is probably best to choose endpoints that extend modestly beyond all stationary points and the real parts of all zeros, including at least one fundamental period, if any, with enough points to resolve the shortest period.
- If there is a symmetry and the expression is C^∞ at the symmetric point, then it is probably best to center the data values on that point. Otherwise it might be more efficient to make that symmetry point be one of the endpoints.

Even a purely *numeric* program that searches for zeros, singularities, extrema, periodicities, and symmetries could help in this regard.

Conclusions

1) With wise use, Eureqa can *sometimes* determine a simpler exact equivalent expression better than current computer algebra systems or *sometimes* transform an expression into a desired form that isn't provided by a computer algebra system.
2) With wise use, Eureqa can sometimes suggest promising forms of expressions that *approximate* a result for which an exact closed form is unobtainable or excessively complicated. However, often you will want to obtain a more accurate result in a thus-revealed class by using a linear or nonlinear regression program built into a computer algebra system or a statistics package.
3) Some simple interface additions *within* Eureqa could increase its utility for the above two purposes. Interfaces *to* Eureqa within more software packages would encourage more use of Eureqa. Embedding Eureqa within those packages would probably be even more seamless.
4) Eureqa uses some custom computer algebra and some standard numerical methods internally. Eureqa would almost certainly benefit from embedding and exploiting a well-developed full-fledged computer algebra system together with classic regression and powerful numerical methods.
5) Eureqa's most frequent and notable successes will probably continue to be with noisy experimental data, but Eureqa shows promise for purposes (1) and (2) above.
6) Some of the most extreme press praise is poetic license sound bites, but symbolic regression deserves a place in the tool kits of many mathematicians, engineers, and scientists.

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Professor of Mathematics

The Department of Mathematics at ETH Zurich (www.math.ethz.ch) invites applications for a position in an algorithmic area of mathematics related to optimization. The duties of the future professor include teaching in mathematics and related areas.

We are seeking candidates with an internationally recognized research record and with proven ability to direct research of highest quality. Expertise and a strong background in optimization and/or computation will be especially appreciated. Willingness to teach at all university levels and to collaborate with colleagues from departments outside mathematics is expected.

Together with the colleagues from the department, the new professor will be responsible for undergraduate courses in mathematics at ETH Zurich for students of mathematics, engineering and natural sciences, and for graduate courses in the programs MSc in Applied Mathematics, MSc in Computational Science and Engineering, MSc in Statistics, and MSc in Quantitative Finance (joint degree with the University of Zurich). The successful candidate will be expected to teach undergraduate level courses (German or English) and graduate level courses (English).

Please apply online at www.facultyaffairs.ethz.ch

Applications should include a curriculum vitae and a list of publications. The letter of application should be addressed **to the President of ETH Zurich, Prof. Dr. Ralph Eichler. The closing date for applications is 30 September 2013.** ETH Zurich is an equal opportunity and family friendly employer and is further responsive to the needs of dual career couples. In order to increase the number of women in leading academic positions, we specifically encourage women to apply.

Acknowledgments

I thank Michael Schmidt and some referees for their helpful suggestions.

References

[1] M. Abramowitz and I. A. Segun, *Handbook of Mathematical Functions*, Dover Publications, 1965.
[2] D. Bailey and J. Borwein, Inverse symbolic calculator, http://isc.carma.newcastle.edu.au/.
[3] R. Corless, E. Postma and D. R. Stoutemyer, Rounding coefficients and artificially underflowing terms in nonnumeric expressions, *ACM Communications in Computer Algebra* **45** (1/2), 2011, 17–48.
[4] DataModeler, http://www.evolved-analytics.com/?q=node/77.
[5] R. Ehrenberg, *Science News* **181**(1), 2012, p. 20, http://www.sciencenews.org/view/feature/id/337207/title/Software_Scientist.
[6] Eureqa building blocks, http://creativemachines.cornell.edu/eureqa_ops.
[7] R. L. Graham, D. E. Knuth and O. Patashnik, *Concrete Mathematics,* 2nd edition, Addison Wesley, 1994, Section 3.2.
[8] Gptips, http://sites.google.com/site/gptips4matlab/.
[9] J. R. Koza, Example of a run of genetic programming, http://www.genetic-programming.com/gpquadraticexample.html.
[10] J. Mullins, Move over Einstein: Machines will take it from here, *New Scientist*, March 22, 2011.
[11] M. Schmidt and H. Lipson, Distilling free-form natural laws from experimental data, *Science* **324** (5923) 2009, 81–85.
[12] M. Schmidt and H. Lipson, Symbolic regression of implicit equations, *Genetic Programming Theory and Practice*, Volume 7, Chapter 5, 2009, 73–85, http://creativemachines.cornell.edu/sites/default/files/Schmidt-Final-2009-06-05.pdf.
[13] M. Schmidt, Eureqa webpage, http://creativemachines.cornell.edu/eureqa.
[14] SourceForge, http://maxima.sourceforge.net/compalg.html.
[15] D. R. Stoutemyer, Dimensional analysis using computer symbolic mathematics, *Journal of Computational Physics* **24** (2), 1977, 141–149.
[16] ______, Can the Eureqa symbolic regression program, computer algebra and numerical analysis help each other? e-print, http://arxiv.org/abs/1203.1023.
[17] ______, Qualitative analysis of mathematical expressions using computer algebra, *Proceedings of the 1976 ACM Symposium on Symbolic and Algebraic Computation*, 97–104.