The calculator and calculus packages: Arithmetic and functional calculations inside LaTeX

Robert Fuster

Abstract

The calculator package allows us to use LaTeX as a calculator, with which we can perform many of the common scientific calculations (with the limitation in accuracy imposed by the TeX arithmetic). The calculus package uses calculator to compute simultaneously a function and its derivative.

1 Introduction

The packages presented in section 2 define several commands to realize calculations within a LaTeX document. The calculator package introduces several new instructions that allow you to calculate with integer and real numbers using LaTeX. As well as add, subtract, multiply and divide, calculator computes powers, square roots, exponentials, logarithms, trigonometric and hyperbolic functions, and performs usual operations with integer numbers such that integer division (quotient and modulo), greatest common divisor, fraction simplification, ... In addition, the calculator package supports some elementary calculations with vectors in two and three dimensions and with 2×2 and 3×3 square matrices.

The calculus package adds to the calculator package several utilities to use and define elementary real functions and their derivatives, including operations with functions, polar coordinates and vector-valued functions.

Several packages can realize some arithmetic operations in TeX and LaTeX, but as far as I know, only the calculus package has the ability to calculate derivatives.

These two packages are designed to perform the calculations needed in the package xpicture (Fuster, 2012b), so the precision it gets is usually sufficient. But if we see TeX as a programming language, why should not we use it to make our calculations? In fact, we can use the calculator package as a length calculator, an alternative to the calc package; and, as another possible application, we can calculate and print the value of a mathematical expression, without using any external application. In this sense, these packages are appropriate if high precision is not required.

In section 3 we review some packages offering similar functionality to calculator. Section 4 describes the main algorithms used by calculator and, finally, in section 5 we explain our conclusions and future improvements of these packages.

2 The calculator and calculus packages

The calculator package defines a large set of commands intended to use LaTeX as a scientific calculator. Its companion package, calculus, gives us some tools to define, manipulate and operate with functions and derivatives. These packages are available, together, from CTAN (Fuster, 2012a).

2.1 calculator

This package operates with *numbers* (at least from the standpoint of the user),[1] not lengths, as is usual within other packages. The operations implemented by the calculator package include routines for assignment of variables, arithmetical calculations with real and integer numbers, two and three-dimensional vector and matrix arithmetic and the computation of square roots, trigonometrical, exponential, logarithmic and hyperbolic functions. In addition, some important numbers, such as $\sqrt{2}$, π and e, are predefined.

The names of all these commands are spelled in capital letters (with a very few exceptions) and, in general, they all need two or more mandatory arguments, one (or more) of which is a number and one (or more) the name of a command where results will be stored, as shown in the examples below.[2]

The new commands defined in this way work in any LaTeX mode.

For example, this instruction

```
\MAX{3}{5}{\solution}
```

stores 5 in the command \solution. Similarly,

```
\FRACTIONSIMPLIFY
   {10}{12}{\numerator}{\denominator}
```

defines \numerator and \denominator as 5 and 6, respectively. Moreover, some of these commands support a first optional argument.

The *data* arguments need not be explicit numbers; they may also consist of commands expanding to a number. This allows us to chain several calculations, as in the following example:

Example 1

$$\frac{2.5^2}{\sqrt{12}} + \mathrm{e}^{3.4} = 31.7685$$

```
% store 2.5^2 in \tempA
 \SQUARE{2.5}{\tempA}
```

[1] Internally, numbers are converted into lengths, but a user need not be aware of this.

[2] Logically, the control sequences that represent special numbers (such as \numberPI) do not need any argument.

```
% store sqrt(12) in \tempB
 \SQUAREROOT{12}{\tempB}

% store e^3.4 in \tempC
 \EXP{3.4}{\tempC}

% \division:=\tempA/tempB
 \DIVIDE{\tempA}{\tempB}{\division}

% \sol:=\division+\tempC
 \ADD{\division}{\tempC}{\sol}

% round to 4 decimal places
 \ROUND[4]{\sol}{\sol}

\[
 \frac{2.5^2}{\sqrt{12}}+\mathrm{e}^{3.4}
      =\sol
\]
```

It does not matter if the *results* arguments are previously defined. But these commands act as declarations, so the scope is local.

The `calculator` and `calculus` user manual, embedded as usual in the source file `calculator.dtx` and also accessible on CTAN as `calculator.pdf`, describes all the commands in that package. We include below an overview and (incomplete) summary.

2.1.1 Predefined numbers

A few numbers are predefined: π and some of its multiples and divisors, the square roots of the first natural numbers (2, 3 and 5), e, 1/e, e^2 and $1/e^2$, the useful cosines of $\pi/6$ and $\pi/4$ (or 30^o and 45^o), the golden ratio and its inverse, the logarithm of 10, and assorted others. Every predefined number is directly accessible calling the command `\number`*XXX*, where *XXX* is a reasonable name of the number (for example, `\numberPI`, `\numberSQRTTWO`, `\numberE`, `\numberCOSXXX` or `\numberGOLD`).

The choice of these numbers is obviously arbitrary, but you can define any number, either directly, using the command `\COPY`,

```
\COPY{12.56637}{\numberFOURPI}
```

or as the result of an operation

```
\SQUAREROOT{7}{\numberSQRTSEVEN}
```

You can use any admissible command name in place of `\numberSQRTSEVEN`.

2.1.2 Real arithmetic

The four basic operations are implemented as `\ADD`, `\SUBTRACT`, `\MULTIPLY` and `\DIVIDE`. As a special case, with the `\LENGTHDIVIDE` command we can divide two lengths and obtain a number.

Example 2 *One inch equals 2.54 centimeters.*

```
\LENGTHDIVIDE{1in}{1cm}\sol
One inch equals \sol{} centimeters.
```

Other implemented operations include integer powers, maximum and minimum of two numbers, absolute value, integer and fractional parts, truncation and rounding.

Example 3

$$\sqrt{2}+\sqrt{3}\approx 3.1463$$

```
\ADD{\numberSQRTTWO}
    {\numberSQRTTHREE}
    {\temp}
\ROUND[4]{\temp}{\sol}
\[
   \sqrt{2}+\sqrt{3}\approx\sol
\]
```

2.1.3 Integer numbers

We can compute the integer quotient and remainder of integer division, greatest common divisor and least common multiple of two numbers, and the irreducible fraction equivalent to a given fraction.

Example 4

$$\frac{4255}{4830}=\frac{37}{42}$$

```
\FRACTIONSIMPLIFY{4255}{4830}{\num}{\div}
\[
   \frac{4255}{4830}=\frac{\num}{\div}
\]
```

2.1.4 Elementary functions

The following real functions are defined: square root, exponential and logarithm, trigonometric (sine, cosine, tangent and cotangent) and hyperbolic (hyperbolic sine, cosine, tangent and cotangent).

Example 5

$$e^2\cos\pi/3=3.69453$$

```
\EXP{2}{\exptwo}
\COS{\numberTHIRDPI}{\costhirdpi}
\MULTIPLY{\exptwo}{\costhirdpi}{\sol}
\[
   \mathrm{e}^2\cos \pi/3=\sol
\]
```

The exponential and logarithm functions admit bases other than e. Trigonometric functions allow radians or degrees as arguments (and also an arbitrary number of circle divisions).

Example 6

$$\log_{10}2=0.30103 \quad \cos 72=0.309$$

```
\LOG[10]{2}{\logtwo}
\DEGREESCOS{72}{\cosseventytwo}
\[
   \log_{10}2=\logtwo\quad
   \cos 72=\cosseventytwo
\]
```

2.1.5 Vectors and matrices

This package operates only with two and three-dimensional vectors and 2×2 and 3×3 square matrices.

Within that limitation, it can add and subtract two vectors, compute the scalar product of two vectors, scalar-vector product, the norm of a vector, normalized vectors and absolute value (in each entry) of a vector.

Example 7

$$\|(1,2,-2)\| = 3$$

```
\VECTORNORM(1,2,-2)\sol
\[
   \left\|(1,2,-2)\right\|=\sol
\]
```

With matrices, the implemented operations are addition, subtraction, matrix product, scalar-matrix product, matrix-vector product, transposition, determinant, inverse matrix, absolute value (in each entry) of a matrix, and solution of a linear (square) system.

The `bmatrix` environment in the following example requires the `amsmath` package.

Example 8

$$\begin{bmatrix} 1 & 2 & 1 \\ -2 & 1 & 1 \\ 3 & 0 & 0 \end{bmatrix}^{-1} = \begin{bmatrix} 0 & 0 & 0.33333 \\ 1 & -1 & -1 \\ -1 & 2 & 1.66666 \end{bmatrix}$$

```
\INVERSEMATRIX(1, 2, 1;
               -2, 1, 1;
                3, 0, 0)(\aUU,\aUD,\aUT;
                         \aDU,\aDD,\aDT;
                         \aTU,\aTD,\aTT)
\[
\begin{bmatrix}
   1 & 2 & 1 \\ -2 & 1 & 1 \\ 3 & 0 & 0
\end{bmatrix}^{-1}=
\begin{bmatrix}
   \aUU & \aUD & \aUT \\
   \aDU & \aDD & \aDT \\
   \aTU & \aTD & \aTT
\end{bmatrix}
\]
```

2.2 `calculus`

The `calculus` package computes simultaneously the values of an elementary function and its derivative. It includes some predefined functions and diverse tools to define new functions, operating with either pre-existing functions or programming the required operations. Moreover, we can also define vector-valued functions and, in particular, curves referring to polar coordinates.

Example 9 *If* $f(t) = \cos t$, *then*

$$f(\pi/4) = 0.7071 \quad f'(\pi/4) = -0.70709$$

```
\COSfunction{\numberQUARTERPI}
            {\cosine}{\Dcosine}
If $f(t)=\cos t$, then
\[
   f(\pi/4)=\cosine\quad
   f'(\pi/4)=\Dcosine
\]
```

For each *function* defined here, you must use the following syntax:

function{*num*}{*cmd1*}{*cmd2*}

where *num* is a number (or a command expanding to a number), and *cmd1* and *cmd2* two control sequences where the values of the function and its derivative (in this number) will be stored.

2.2.1 Predefined functions

The following functions are defined:

- zero ($f(t) = 0$) and one ($f(t) = 1$) constant functions,
- identity ($f(t) = t$),
- reciprocal ($f(t) = 1/t$),
- square ($f(t) = t^2$),
- cube ($f(t) = t^3$),
- square root ($f(t) = \sqrt{t}$),
- exponential ($f(t) = \exp t$),
- logarithm ($f(t) = \log t$),
- trigonometric ($f(t) = \sin t$, $f(t) = \cos t$, $f(t) = \tan t$ and $f(t) = \cot t$),
- hyperbolic ($f(t) = \sinh t$, $f(t) = \cosh t$, $f(t) = \tanh t$ and $f(t) = \coth t$),
- and the Heaviside function ($f(t) = 0$, if $t < 0$; $f(t) = 1$, if $t \geq 0$).

All these functions can be used as in example 9.

2.2.2 Operating with functions

The easiest way to define new functions is to perform some operation with already-defined functions. Available operations allow us to define constant functions, to add, subtract, multiply or divide two functions, scale variable or function, raise a function to an integer power, compose two functions and make a linear combination of two functions.

Example 10 *If* $f(t) = (1 + \cos t)^2$, *then*

$$f(\pi/3) = 2.25 \text{ and } f'(\pi/3) = -2.59804$$

```
% g(t)=1+cos(t)
\SUMfunction
    {\ONEfunction}{\COSfunction}
    {\gfunction}

% F(t)=g(t)^2
\COMPOSITIONfunction
    {\SQUAREfunction}{\gfunction}
    {\Ffunction}

% sol=F(pi/3), Dsol=F'(pi/3)
\Ffunction{\numberTHIRDPI}{\sol}{\Dsol}

\noindent If $f(t)=(1+\cos t)^2$,
then $f(\pi/3)=\sol$ and $f'(\pi/3)=\Dsol$.
```

2.2.3 Polynomials and low-level function definition

Although the polynomials can be defined as linear combinations of powers, `calculus` includes some commands to directly define linear, quadratic, and cubic polynomials. For example, we can define the polynomial $p(t) = 2 - 3t^2$ by typing

```
\newqpoly{\mypoly}{2}{0}{-3}
```

Also, low-level commands exist to define a function by programming it and its derivative.

2.2.4 Vector-valued functions and polar coordinates

A vector-valued function can be identified with a pair of ordinary functions.[3] If the functions `\Xfunct` and `\Yfunct` are already defined, then

```
\VECTORfunction{\Ffunct}{\Xfunct}{\Yfunct}
```

declares the new vector-valued function `\Ffunct` with component functions `\Xfunct` and `\Yfunct`. For example, we can define the function $f(t) = (t^2, t^3)$ by typing

```
\VECTORfunction{\Ffunction}
  {\SQUAREfunction}{\CUBEfunction}
```

The `xpicture` package uses vector-valued functions to plot parametrically defined curves.

In this respect, curves defined in polar coordinates are a particularly interesting case. To define the polar curve $\rho = f(\phi)$ (where ρ and ϕ are the polar radius and arc), the `calculus` package includes the command

```
\POLARfunction{\ffunction}{\Pfunction}
```

where `\ffunction` is an already defined function and `\Pfunction` is the new polar function. In the following example, we define the *five-petal* curve, $\rho = \cos 5\phi$.

[3] Only two-dimensional vector functions are defined.

Example 11 *The polar curve $\rho = \cos 5\phi$ passes through the point* $(0.24998, 1.73204)$. *At this point, its tangent vector is* $(0.43298, 3.99986)$.

```
\SCALEVARIABLEfunction{5}{\COSfunction}
                        {\myfunction}
\POLARfunction{\myfunction}{\FIVEROSE}
\FIVEROSE{\numberTHIRDPI}{\x}{\y}{\Dx}{\Dy}

\noindent The polar curve $\rho=\cos 5\phi$
passes through the point $(\x,\y)$. At this
point, its tangent vector is $(\Dx,\Dy)$.
```

3 Other arithmetic-related packages in TEX

The limitations of classic TEX arithmetic are well known (Knuth, 1990). In short, TEX can operate with integer numbers n restricted by the relation $|n| \leq 2^{32} - 1$.[4] Noninteger arithmetic is performed on lengths via conversion to a whole number of scaled points (`sp`). The largest admissible length is $16383.99998\,\texttt{pt} \approx 2^{14}\,\texttt{pt} = 2^{30}\texttt{sp}$ (a point equals 2^{16} scaled points). Therefore standard TEX can not manage real numbers greater than 16383.99998. Moreover, considering that the smaller length is one scaled point ($1\,\texttt{sp} \approx 0.000015\,\texttt{pt}$), TEX cannot distinguish between two lengths differing in less than 0.00002 points. With the standard TEX behavior, this is the maximum level of accuracy we can expect.

These restrictions are absolutely negligible if you consider the main aim of TEX: for fine typesetting of text documents, TEX arithmetic is more than sufficient. But for some questions (e.g., composing quality graphics in high definition) more exacting arithmetic is required. Either for this reason or to implement a more user friendly syntax, a few authors have worked on issues related to the TEX arithmetic, as we can see from a look at the literature or in packages on CTAN. Some representative examples:

- Probably the most widely used package in this matter is `calc` (Thorup, Jensen, and Rowley, 1998). This package introduces a friendly syntax for the end user, reimplementing the LATEX length and counter manipulating commands and adding the ability to manipulate counters and lengths with infix notation. Among other improvements, this package allows you to multiply and divide by real numbers.
- In fact, calculating divisions is a notable problem for a TEX user, because the `\divide` command only supports integer divisors. Claudio Beccari

[4] In other words, the absolute value of an integer must not be greater than 2147483647.

introduced a nice algorithm for division (Beccari, 2006) in his `curve2e` package, which was subsequently adopted by Gäßlein, Niepraschk, and Tkadlec (2011) in `pict2e`.

- For obvious geometrical reasons, we often need to calculate trigonometrical ratios; the `trig` package, distributed within the `graphics` bundle (Carlisle, 2005), solves this problem by approximating the sine function with the Taylor-McLaurin series.[5] Beccari, in `curve2e`, uses a continued fraction approximation for the tangent function.
- The `pgf/tikz` bundle (Tantau, 2010) includes a *mathematical engine* to compute calculations needed by `tikz` pictures, including real and integer operations. This engine may be used separately by loading the `pgfmath` package.

  For the calculations it can perform, this package is the closest to `calculator`. Table 1 compares the results obtained using `calculator` and `pgf` with some of the examples included in this document. The selection is obviously arbitrary, but it shows that performance of calculations in these two packages is similar.

Regarding the problem of precision, the `fp` package (Mehlich, 1999) extends the fixed-point arithmetic to the interval

$$[-99999999999999999.99999999999999999,\\ 99999999999999999.99999999999999999]$$

which is more than enough in many cases. The `fltpoint` package (Guthöhrlein, 2004) provides similar results.[6]

To deal with numbers of arbitrary magnitudes, however, fixed-point arithmetic is always limited. The future LaTeX3 (Mittelbach et al., 2009) will support a floating point arithmetic that is already available as a part of the experimental package `expl3` (The LaTeX3 Project, 2012).

The `pgf/tikz` system includes, as an alternative to its mathematical engine, a fixed point library (using `fp`) and also a floating point library, to work with very large or very small numbers.

Another approach to arithmetic problems is to call an external application to take care of the calculations; this option has the obvious advantage of avoiding the restrictions imposed by the TeX arithmetic, both in efficiency and accuracy. The `sagetex` package (Drake, 2009)[7] or the commercial product *Scientific WorkPlace*[8] are well-known examples of these ideas. In a more general way, LuaTeX (Hoekwater, 2009) integrates an interface between TeX and the programming language Lua (Ierusalimschy, Celes, and de Figueiredo, 2012).

4 The algorithms

`calculator` performs additions, subtractions and products using ordinary TeX arithmetic on lengths. Division is essentially identical to Beccari's algorithm (adapted to the `calculator` syntax) with only one small improvement: as in Beccari's approach, each decimal place is calculated by multiplying by 10 the last remainder and dividing it by the divisor; but when the remainder is greater than 1638.3, an overflow occurs, because 16383.99998 is the greatest number. So, instead, we multiply the divisor by 0.1.

To calculate square roots and logarithms, we use Newton's method: $\sqrt{x}$ is the limit of

$$x_0 = x, \quad x_{n+1} = x_n - \frac{x_n^2 - x}{2x_n}$$

and $\log x$ is the limit of

$$x_0 = x - 1, \quad x_{n+1} = x_n + \frac{x}{\mathrm{e}^{x_n}} - 1$$

(this iterative method is used for $x \in [1, \mathrm{e}^2]$; otherwise, the relation $\log xy = \log x + \log y$ is used).

Trigonometric and exponential functions use generalized continued fractions: trigonometric functions are computed by reduction to sine or tangent,

$$\sin x = \cfrac{x}{1 + \cfrac{x^2}{2\cdot 3 - x^2 + \cfrac{2\cdot 3x^2}{4\cdot 5 - x^2 + \cfrac{4\cdot 5x^2}{6\cdot 7 - x^2 + \cdots}}}}$$

$$\tan x = \cfrac{1}{\cfrac{1}{x} - \cfrac{1}{\cfrac{3}{x} - \cfrac{1}{\cfrac{5}{x} - \cfrac{1}{\cfrac{7}{x} - \cfrac{1}{\cfrac{9}{x} - \cfrac{1}{\cfrac{11}{x} - \cdots}}}}}}$$

(for $|x| < \pi/2$; otherwise, angles are reduced to this case). The exponential applies this approximation

[5] For small values of x, the seven-degree McLaurin polynomial of $\sin x$ is a very accurate approximation.

[6] Despite its name, this package works with fixed point arithmetic.

[7] Sage (The Sage Project, 2012) is a free, open-source math software that supports research and teaching in algebra, geometry, number theory, cryptography, numerical computation, and related areas. The `sagetex` package acts as an interface between Sage and LaTeX.

[8] *Scientific WorkPlace* (MacKichan Software Inc., 2012) is a LaTeX typesetting and computer algebra integrated system.

		calculator		pgf	
expression	scilab	result	rel. error	result	rel. error
$\frac{2.5^2}{\sqrt{12}} + \mathrm{e}^{3.4}$	31.76831964	31.76854	0.000007	31.76741	0.000029
$\sqrt{2} + \sqrt{3}$	3.14626437	3.14627	0.000002	3.14627	0.000002
$\mathrm{e}^2 \cos \pi/3$	3.694528049	3.69453	0.000001	3.69435	0.000048
$\log_{10} 2$	0.301029996	0.30103	0.000000	0.3001	0.003089
$\cos 72^\circ$	0.309016994	0.309	0.000055	0.30902	0.000010

Table 1: Comparison between `calculator` and `pgf`. The second column shows results obtained with `scilab`; columns four and six contain the relative errors when using `calculator` and `pgf`.

(Lauschke, 2009):

$$\exp x \approx 1 + \cfrac{2x}{2 - x + \cfrac{x^2/6}{1 + \cfrac{x^2/60}{1 + \cfrac{x^2/140}{1 + \cfrac{x^2/256}{1 + \cfrac{x^2}{396}}}}}}$$

(for $-6 < x < 3$; otherwise, the relation $\exp(x+y) = (\exp x)(\exp y)$ is applied).

5 Conclusions and future work

The packages `calculator` and `calculus` were born as working tools for the `xpicture` package, and they provide sufficient accuracy for the requirements of this package.

The package `calculator` is also appropriate to do geometrical calculations related to page composition and, in general, this package can be used for any scientific calculation which does not require overmuch precision. In fact, the performance of this package is similar to those of other packages that use the arithmetic of TEX. Only high precision packages give good results in complex calculations.

With respect to the calculations it can make, the performance of `calculator` is similar to those of `pgfmath`; all other packages offer a very limited set of functions.

The `calculator` package uses the typical TEX syntax: calculations are performed by calling the `calculator` commands and results are stored in new commands. The `fp` package behaves similarly. Other packages, like `pgfmath` and experimental `expl3`, admit infix mathematical notation; this is a nice feature not supported by `calculator`.

The `calculus` package provides a user friendly method to define functions and simultaneously calculate the values of a function and its derivative. The accuracy of the calculations is related to the accuracy of the `calculator` package, so if this is improved, the `calculus` package will become a solid tool to evaluate functions and derivatives. No other package has the ability to calculate derivatives.

There are some improvements that we hope to add soon, such as the implementation of additional functions (the inverse trigonometric and hyperbolic functions, and maybe some boolean functions), the inclusion of more utilities related to the definition of polynomials (definition of polynomials of arbitrary degree, construction of polynomials from their roots, implementation of the interpolating polynomial), ...

On the other hand, to make these packages truly interesting, we will attack the issue of accuracy. We will study the possibility of incorporating the option to obtain more accurate results, perhaps by loading the `fp` package, or exploring the possibility of using floating point arithmetic.

References

Beccari, Claudio. "LaTeX 2_ε, `pict2e` and complex numbers". *TUGboat* **27**(2), 202–212, 2006.

Carlisle, D. P. "Packages in the 'graphics' bundle". Available from CTAN, `/macros/latex/required/graphics`, 2005.

Drake, Dan. "The SageTeX package". Available from CTAN, `/macros/latex/contrib/sagetex`, 2009.

Fuster, Robert. "The `calculator` and `calculus` packages: Use LaTeX as a scientific calculator". Available from CTAN, `/macros/latex/contrib/calculator`, 2012a.

Fuster, Robert. "The `xpicture` package: A LaTeX package intended to extend the picture environment". `http://www.upv.es/~rfuster/xpicture`, 2012b.

Gäßlein, Hubert, R. Niepraschk, and J. Tkadlec. "The `pict2e` package". Available from CTAN, `/macros/latex/contrib/pict2e`, 2011.

Guthöhrlein, Eckhart. "The `fltpoint` package". Available from CTAN, `macros/latex/contrib/fltpoint/`, 2004.

Hoekwater, Taco. "The LuaTEX program". Available from CTAN, `/systems/luatex/base`, 2009.

Ierusalimschy, Roberto, W. Celes, and L. H. de Figueiredo. "Lua: The programming language". `http://www.lua.org`, 2012.

Knuth, Donald E. *The TEXbook*. Addison Wesley, Reading, Massachussets, 1990.

Lauschke, Andreas. "Convergence Acceleration with Canonical Contractions of Continued Fractions". `http://216.80.120.13:8080/webMathematica/LC/general.jsp`, 2009.

MacKichan Software Inc. "Scientific WorkPlace. The integration of LATEX and Computer Algebra". `http://www.mackichan.com`, 2012.

Mehlich, Michael. "The `fp` Package". Available from CTAN, `/macros/latex/contrib/fp`, 1999.

Mittelbach, Frank, R. Schöpf, C. Rowley, D. Carlisle, J. Braams, R. Fairbairns, T. Lotze, W. Robertson, J. Wright, and B. Le Floch. "The LATEX3 Project". `http://www.latex-project.org/latex3.html`, 2009.

Tantau, Till. "The Ti*k*Z and PGF Packages. Manual for version 2.10". `http://sourceforge.net/projects/pgf`. Also available from CTAN, `/graphics/pgf`, 2010.

The LATEX3 Project. "The `expl3` package and LATEX3 programming". Available from CTAN, `/macros/latex/contrib/expl3`, 2012.

The Sage Project. "Sage: Open source mathematics software". `http://www.sagemath.org`, 2012.

Thorup, Kresten Krab, F. Jensen, and C. Rowley. "The `calc` package. Infix notation arithmetic in LATEX". Available from CTAN, as part of the `tools` bundle, `/macros/latex/required/tools`, 1998.

◇ Robert Fuster
Universitat Politècnica de València
Departament de Matemàtica Aplicada
Camí de Vera, 14
València E46022
Spain
`rfuster (at) mat dot upv dot es`
`http://www.upv.es/~rfuster/`