# GrafEq$\mathbb{C}$: Reliable graphing in the complex plane

Stefan Becuwe*, Annie Cuyt*, Jo Janssen*, Walter Schreppers*, Jeff Tupper†
* Universiteit Antwerpen, Departement Wiskunde en Informatica
Middelheimlaan 1, B-2020 Antwerpen, Belgium
Email: annie.cuyt@ua.ac.be
† Pedagoguery Software Inc., 4446 Lazelle Ave.
Terrace, B.C. V8G 1R8, Canada
Email: tupper@peda.com

***Abstract*— To illustrate the *Handbook of Continued Fractions for Special Functions* with level curves of significant decimal digits of continued fraction approximants, the reliable graphing package GrafEq is extended in two ways: with support for and automatic conversion of complex variables on the one hand, and with support for additional (non-elementary) functions implemented externally on the other hand.**

## I. Motivation

In the wake of the DLMF project [LOZ 06], which revises Abramowitz and Stegun's technical best-seller *Handbook of Mathematical Functions*, a smaller international team set up the project *Handbook of Continued Fractions for Special Functions* [CUY 06]. One of the aims of the latter project is to produce level curves of $s$ such that

$$|f(z) - f_n(z)| \leq \frac{1}{2} 10^{-s+1} |f(z)|, \tag{1}$$

where $f_n(z)$ is the $n$th approximant of the continued fraction for the special function $f(z)$ with $z \in \mathbb{C}$. These level curves delimit the region in the complex plane where $f_n(z)$ guarantees at least $s$ significant decimal digits (in real or complex arithmetic). In addition to these level curves, which the readers can reproduce for different values of $s$ and $n$, the latest know-how [CUY 05] on the behaviour of continued fractions and their tails, on rounding error, and on truncation error estimates is to be translated into reliable software for the evaluation of several special functions.

If we take $x = \Re z$ and $y = \Im z$, the problem is a familiar one—that of producing illustrations of relations defined implicitly by formulae in $x$ and $y$. As this problem has been discussed for centuries, there is an abundance of partial solutions to it. Until recently [TUP 01], however, there was no (published) method capable of reliably solving this problem, even when restricted to the formulae encountered in high school. The algorithm implemented in GrafEq (pronounced "graphic") correctly graphs mathematical formulae involving the basic operations, inequalities, and known elementary functions [Ped04]. When applied to a difficult formula that is beyond its capabilities, if *show work* is enabled, GrafEq clearly marks the pixels that it cannot decide on. At no point does GrafEq use any approximations that could cause it to produce an incorrect graph.

For the continued fraction project, we have extended GrafEq in two ways.

- Since the manual conversion of complex formulae into corresponding real formulae (with $x = \Re z$ and $y = \Im z$) is tedious and error-prone, we have added support for complex variables to the user interface, along with automatic conversion.
- Since GrafEq only has implementations of the elementary functions and none of the special functions, we dynamically extend the list of functions available to GrafEq and provide their implementations.

The first extension is discussed in section III and the second in section IV; section II summarizes the internal workings of GrafEq.

## II. GrafEq's internal workings

With GrafEq's mathematical foundations, any formula $r(x, y)$, when evaluated with specific real $x$ and $y$, is always either false ($\mathsf{F}$) or true ($\mathsf{T}$). Given a mathematical formula and a rectangular region $[L, R] \times [B, T]$ of the Cartesian plane, GrafEq produces an illustration—a $W \times H$ rectangular array of pixels. Each pixel represents a closed rectangular region of the plane. Since no algorithm can produce correct black and white illustrations—black meaning that there is at least one solution of $r(x, y)$ within the pixel and white meaning that there are no solutions within the pixel—we allow our algorithm to color some pixels grey—grey meaning that there may or may not be solutions of $r(x, y)$ within the pixel.

Even if the bounds $L$, $R$, $B$, and $T$ of the graphing area are given as machine numbers, the bounds of individual pixels might not be representable exactly. GrafEq uses inner and outer bounds of the rectangular region corresponding to the pixel as shown in Fig. 1: the inner bounds are used to establish the presence of solutions and the outer bounds to establish the absence of solutions. GrafEq uses an interval arithmetic of boolean values to represent, and process, the result of formula evaluations. Three boolean intervals are possible: $\langle \mathtt{F}, \mathtt{F} \rangle$, $\langle \mathtt{F}, \mathtt{T} \rangle$, and $\langle \mathtt{T}, \mathtt{T} \rangle$; $\mathtt{F} < \mathtt{T}$. The boolean intervals make possible

- domain tracking, by keeping track of whether or not a quantity, such as $\sqrt{x}$, is well-defined;
- continuity tracking, by providing information on whether a quantity is continuous or not within given bounds; and
- branch cut tracking, by tracing to which branch each piece belongs when breaking a discontinuous evaluation apart into pieces.

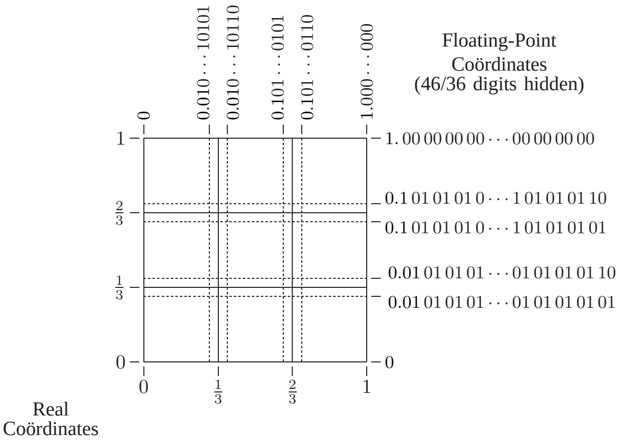


Fig. 1. GrafEq: inner and outer bounds of a rectangular region.

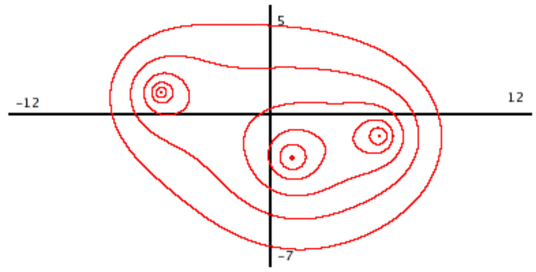


Fig. 2. Illustration of $(z-5+i)(z-1+2i)(z+5-i)=w$ for $|w| \in \{2, 16, 32, 64, 128, 256\}$.

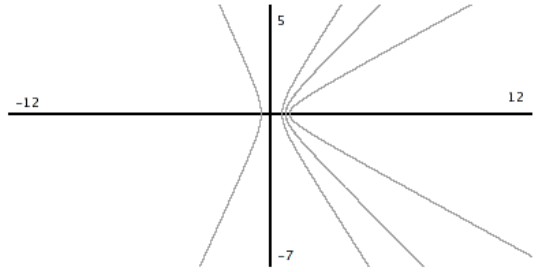


Fig. 3. Plot of $z=\cos(w)$ for $\Re w \in \{-0.5, 1, 2, \pi/4\}$.

A discontinuous evaluation requires the facility to return two disjoint intervals for a function evaluation. For instance, $\tan(x)$ for $x \in [-\pi/2-\delta, -\pi/2+\epsilon]$ with $\delta$ and $\epsilon$ small and positive evaluates to $(-\infty, \tan(-\pi/2+\epsilon)] \cup [\tan(-\pi/2-\delta), +\infty)$. GrafEq also supports this.

## III. GrafEqℂ's complex parser and optimizer

Let $z = x + \mathtt{i}y$ and $w = u + \mathtt{i}v$. We want to extend the above to allow the graphing of relations $r(z, w)$ in GrafEqℂ (pronounced "graphics"). To this end the formula involving the complex variables $z$ and $w$ is transfered to two parse trees containing $\Re r(x, y, u, v)$ and $\Im r(x, y, u, v)$, expressed in terms of the real and imaginary parts of $z$ and $w$. We here make use of GrafEq's support for two additional real parameters on top of the real variables $x$ and $y$. So, in reality, GrafEq as described in section II, permits to graph relations of the form $r_{u,v}(x, y)$ with $u, v \in \mathbb{R}$. Still, in the end, we can only visualize two-dimensional graphs.

The parse trees for $\Re r(x, y, u, v)$ and $\Im r(x, y, u, v)$, which can get very large, are optimized before passing them to the rendering algorithm. The optimizations currently carried out include

- the canonicalization of polynomials, which rewrites $x - x + x + 2x$ as $3x$, $xxx \times x^{-2}$ as $x$ (adding the condition that $x \neq 0$), $4x^2 - 2x^2$ as $2x^2$, and $3xy + 5xy$ as $8xy$;
- the elimination of neutral elements, 0 for addition and 1 for multiplication, which may come out of the symbolic simplification; and
- the recognition of simple identities, $\mathtt{i}^2 = -1$ and $\sqrt{1} = 1$.

This symbolic preprocessing was suggested for real expressions in [TUP 01].

We can illustrate—in addition to the level curves—curves such as lemniscates (as shown in Fig. 2) or images of straight lines in the complex plane by the cosine function $\cos(z)$ (as shown in Fig. 3). GrafEqℂ has finished the former, but not the latter. Generally, GrafEqℂ will not finish graphing bivariate complex formulae. It cannot as its main tool for establishing the existence of solutions of low-dimensional (one or more dimensions below the ambient space) relations is the intermediate value theorem—in Fig. 3, the curve is of real-dimension 1 and lies in a real-dimension 4 space. GrafEqℂ can finish the former as GrafEq has symbolically re-expressed the relation as a bivariate formula so that the curve is of real-dimension 1 in a real-dimension 2 space. Although Fig. 3 is not finished, one does know where the curve is allowed to be, namely in the grey area. Other programs complete the graphing process using an arbitrary, yet practical, criterion such as elapsed time.

## IV. Special function plotting

The results obtained in [CUY 05], on the implementation of special functions in certain regions of the complex plane, are used. When plotting (1), both the function $f(z)$ and the rational approximant $f_n(z)$ are evaluated reliably, the latter by making use of the interface to complex numbers discussed in section III and the former by calling a reliable interval evaluation [VER 05] of $f(z)$. The interval implementation also needs to support the domain and continuity tracking bits required by GrafEq's internal engine and described in section II.

Consider the T-fraction approximant $f_n(z)$ to $f(z) = \exp(z) - 1$,

$$f_n(z) = \frac{z}{1-z+} \mathop{\mathrm{K}}_{m=2}^{n} \frac{(m-1)z}{m-z}, \tag{2a}$$

where

$$|f(z) - f_n(z)| \leq \frac{1}{2} 10^{-s+1} |f(z)|, \tag{2b}$$
$$-30 \leq \Re z \leq 5, \ |\Im z| \leq 35, \ \text{and } s \in \{6, 7, 8\}.$$

T-fractions approximate well for small $z$ and small $1/z$.

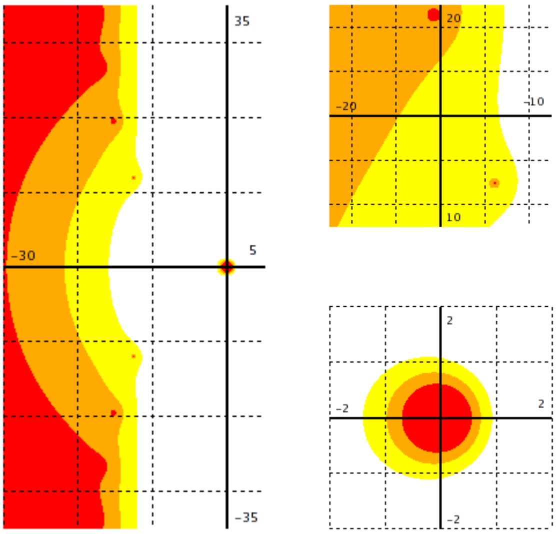


Fig. 4. Level curves of (2b) for $s \in \{6, 7, 8\}$.

```
#include "jte.h"

// define your own function sum(x,y)
int Sum(int num_args, JTE_Interval **args, JTE_Interval *result)
{
  ... // switch to round down
  result->lower = args[0]->lower + args[1]->lower;
  ... // switch to round up
  result->upper = args[0]->upper + args[1]->upper;

  result->flags = args[0]->flags | args[1]->flags;
  // and other settings that can be passed to the engine

  return 1; // number of intervals returned
}

#define NumExternals 1

JTE_ExternalDescription externals[NumExternals] = {
  {"sum",  2,  Sum}
  // name in the engine, number of arguments, external name
};

int main(int argc,char **argv)
{
  ...
  JTE_Init();
  JTE_RegisterExternals(NumExternals,externals);
  ...
}
```

Fig. 5. Sample code.

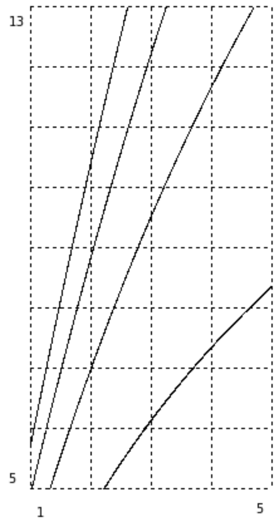


Fig. 6. Plot of (4) for $x \in [1, 5]$ and $\ell \in \{1, 2, 3, 4\}$.

The regions corresponding to $s$ being 6, 7, and 8 are, respectively, printed in light-grey (originally coloured yellow), medium-grey (originally orange) and dark-grey (originally red) in Fig. 4. Axes and grid-lines are black. For the sake of the presentation in grey-shades, we have temporarily rendered the uncertain pixels transparent instead of grey. The very small isolated regions in the left half-plane, which are clearly shown by our methods, are not shown by conventional graphing methods. The small insets are each $192 \times 192$ pixel arrays while the larger picture is $192 \times 384$ pixel arrays. With $73\,728 (= 192 \times 384)$ plotpoints, Maple does not produce the correct graph for $s = 8$.

External functions (C, C++, Fortran, Maple, ...) are called as shown in Fig. 5 for $\mathrm{sum}(x, y)$. For each external function, the engine needs a `JTE_ExternalDescription` list containing: the name used in the user interface ($\mathrm{sum}$), the number of arguments (2) and the name of the external function (`Sum`). After initialization of the engine, the external functions need to be registered. During the rendering, GrafEq's internal engine prompts `Sum`$(x, y)$ for a reliable evaluation of $\mathrm{sum}(x, y)$ with $x$ and $y$ belonging to floating-point intervals.

To conclude, we show in Fig. 6 the number of significant decimal digits that can be guaranteed by the $n$th approximant $g_n(x)$ of the continued fraction representation (3) of $\mathrm{erfc}(x)$, where $x \in [1, 5] \subseteq \mathbb{R}$:

$$g_n(x) = \frac{e^{-x^2}}{\sqrt{\pi}} \left( \frac{2x}{2x^2 + 1 + } \mathop{\mathbf{K}}_{k=1}^{n} \frac{-(2k-1)2k}{2x^2 + (4k+1)} \right). \tag{3}$$

Here $\mathrm{erfc}(x)$ is implemented as an external function. For $\ell$ being 1, 2, 3, and 4, respectively, the curves

$$y = -\log_{10} \frac{|\mathrm{erfc}(x) - g_{3\ell}(x)|}{5\,|\mathrm{erfc}(x)|} \tag{4}$$

can be found in Fig. 6 when traversing it from the bottom right to the upper left corner.

## Acknowledgment

The first author is supported by the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

## References

[CUY 05] Cuyt A., Verdonk B., Waadeland H., *Efficient and reliable multiprecision implementation of elementary and special functions*, 2005.

[CUY 06] Cuyt A., Petersen V., Verdonk B., Waadeland H., Jones W. B., Bonan-Hamada C., Handbook of Continued Fractions for Special Functions, `http://www.cant.ua.ac.be/workshops/cfhb.html`, 2006.

[LOZ 06] Lozier D., Olver F., Clark C., Boisvert R., Digital Library of Mathematical Functions, `http://dlmf.nist.gov/`, 2006.

[Ped04] Pedagoguery Software Inc., GrafEq 2.12, 2004, `http://www.peda.com/grafeq/`.

[TUP 01] Tupper J., Reliable two-dimensional graphing methods for mathematical formulae with two free variables, *ACM SIGGRAPH*, p. 77–86, 2001.

[VER 05] Verdonk B., Vervloet J., Cuyt A., *Blending interval and set arithmetic for maximal reliability*, *Computing*, vol. 74, p. 41–65, 2005.