A continued fractions package for special functions

FRANKY BACKELJAUW and ANNIE CUYT
University of Antwerp, Belgium

The continued fractions for special functions package (in the sequel abbreviated as CFSF package) complements a systematic study of continued fraction representations for special functions. It provides all the functionality to create continued fractions, in particular k-periodic or limit k-periodic fractions, to compute approximants, make use of continued fraction tails, perform equivalence transformations and contractions, and much more. The package, developed in Maple, includes a library of more than 200 representations of special functions, of which only 10% can be found in the NBS Handbook of Mathematical Functions with Formulas, Graphs and mathematical Tables by Abramowitz and Stegun.

Categories and Subject Descriptors: G.1.2 [**Numerical Analysis**]: Approximation—*Special function approximations*

Additional Key Words and Phrases: CAS software, continued fractions, Maple, special functions

1. INTRODUCTION

Special functions are pervasive in all fields of science and industry. The most well-known application areas are in physics, engineering, chemistry, computer science and statistics. Because of their importance, several books and websites (see for instance `functions.wolfram.com`) and a large collection of papers have been devoted to these functions. Of the standard work on the subject, the *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical table* edited by Milton Abramowitz and Irene Stegun, the American National Institute of Standards and Technology (formerly National Bureau of Standards) claims to have sold over 700 000 copies (over 150 000 directly and more than fourfold that number through commercial publishers)!

But the NBS Handbook [Abramowitz and Stegun 1964], as well as the Bateman volumes written in the early fifties [Erdélyi et al. 1953a; 1953b; 1955], are currently out of date due to the rapid progress in research and the revolutionary changes in technology. Already in the nineties (see for instance [Lozier 2000]) several projects were launched to update the principal handbooks on special functions and to make them available on the web and extend them with computational facilities.

This package complements the *Handbook of Continued Fractions for Special Functions* [Cuyt et al. 2008], which results from a systematic study of continued fraction representations for special functions. Only 10% of the continued fractions contained in this book, can also be found in the NBS Handbook or at the Wolfram website! And in this count we systematically disregard continued fractions which are equivalent to series, because of their limited interest (see section 5.3). The Maple package discussed here offers all the basic support required to handle continued fractions on the one hand, and implements all series and continued fraction representations listed in [Cuyt et al. 2008] on the other hand. The package requires Maple 9 or above and can be downloaded from `www.cfsf.ua.ac.be`.

2. MAPLE'S NUMTHEORY[CFRAC]

A continued fraction is an expression of the form

$$f(z) := b_0(z) + \cfrac{a_1(z)}{b_1(z) + \cfrac{a_2(z)}{b_2(z) + \cfrac{a_3(z)}{b_3(z) + \ddots}}} \tag{1}$$

where the partial numerators $a_m(z)$ and partial denominators $b_m(z)$ are complex numbers or functions of z with $a_m(z) \neq 0$ for all m. A common name for the ordered pair $[a_m(z), b_m(z)]$ is element. In the sequel, the variable z is dropped when no confusion can arise. The following alternative notations are also used:

$$f(z) = b_0 + \frac{a_1}{b_1} \; + \; \frac{a_2}{b_2} \; + \; \ldots = b_0 + \mathop{\mathrm{K}}_{m=1}^{\infty} \left(\frac{a_m}{b_m} \right).$$

The development of this continued fraction package is necessary because the built-in support for continued fractions in symbolic computing environments is rather inadequate. For instance, Maple only offers limited basic support to handle continued fractions through its `numtheory[cfrac]` command. With this command it is possible to create a continued fraction expansion for a number, a rational function, a series or another algebraic object (such as a special function), up to a user-defined constant number of elements (the default is 10). For example, a continued fraction expansion for the complementary error function is obtained via:

```
> erfcCF := cfrac( erfc(z) );
```

$$erfcCF := 1 + \cfrac{z}{-\cfrac{\sqrt{\pi}}{2} + \cfrac{z^2}{\cfrac{6}{\sqrt{\pi}} + \cfrac{z^2}{5\sqrt{\pi} + \ddots}}} \tag{2}$$

Other forms can be created by specifying either `regular`, `simregular`, `simple`, `semisimple` or `monic`, e.g.:

```
> cfrac( erfc(z), simregular );
```

$$1 - \cfrac{2az/\sqrt{\pi}}{1 + \cfrac{z^2/3}{1 - \cfrac{z^2/30}{1 + \cfrac{39z^2/70}{1 - \ddots}}}}$$

However, this command sometimes fails to return a continued fraction expansion while an expansion of the requested form exists. Some examples:

```
> cfrac( erf(z), 'semisimple' );
Error, (in numtheory:-cfrac) unable to evaluate sign
> cfrac( GAMMA(a,z), z );
Error, (in series/exp) unable to compute series
> cfrac( Ei(1,z) );
Error, in (numtheory:-cfrac) invalid series
```

Classical approximants (also called convergents, see section 4.4) can be computed using Maple's `nthconver` command. For example, the fifth approximant of (2) is computed by:

```
> nthconver( erfcCF, 5 );
```

$$-\frac{1478}{15}\,\frac{z^5-\frac{2475}{1478}\sqrt{\pi}z^4+\frac{3570}{739}z^3-\frac{9975}{739}\sqrt{\pi}z^2+\frac{49140}{739}z-\frac{24570}{739}\sqrt{\pi}}{\sqrt{\pi}\,(165\,z^4+1330\,z^2+3276)}$$

But when computing the tenth approximant, one gets:

```
> nthconvert( erfcCF, 10 );
Error, (in numtheory:-nthconver) wrong arguments
```

Most important is that `numtheory[cfrac]` lacks the functionality to deal with limit-periodic continued fractions. Many continued fraction expansions for special functions are defined by a repetition of a limited number of general elements in terms of a running index, such as in (3) and (4). Yet `numtheory[cfrac]` does not allow to create continued fraction representations of that form.

Finally, `numtheory[cfrac]` does not offer the possibility of equivalence transformations, transformations between series and fractions or any functionality to work with continued fraction tails (see section 4.5).

3. FAMILIES OF CONTINUED FRACTIONS

For a lot of elementary and special functions in mathematics, physics and engineering, and for many well-known constants, there exist elegant continued fraction expansions. Many of these continued fractions offer the advantage that they converge much more rapidly than power series and in a much larger domain of the complex plane. In addition, the partial numerators and/or denominators often satisfy some useful properties such as monotonicity or limit-periodicity. Such properties allow to further improve the truncation error bounds [Cuyt et al. 2006].

Continued fractions can be categorized into several families, depending on the properties of their partial numerators and denominators. Among others, one can distinguish between the following families:

— *C-fractions* are continued fractions of the form

$$1+\mathop{\mathrm{K}}_{m=1}^{\infty}\left(\frac{\alpha_m z^{\beta_m}}{1}\right),\quad \alpha_m\in\mathbb{C}\setminus\{0\},\quad \beta_m\in\mathbb{N}.$$

If furthermore $\beta_m = 1$ for $m \geq 1$, then it is called a *regular C-fraction*.

— *S-fractions* or Stieltjes fractions are of the form

$$\mathop{\mathrm{K}}_{m=1}^{\infty}\left(\frac{a_m z}{1}\right), \quad a_m > 0.$$

Any continued fraction that is equivalent to this form is also called an S-fraction. For a definition of equivalence, we refer to section 5.

— *general T-fractions* or Thron fractions are of the form

$$\mathop{\mathrm{K}}_{m=1}^{\infty}\left(\frac{F_m z}{1+G_m z}\right), \quad F_m \in \mathbb{C}\setminus\{0\}, \quad G_m \in \mathbb{C}.$$

If furthermore all $F_m = 1$ then the fraction is called a *T-fraction* without further specification. When $F_1 z$ is replaced by F_1, the continued fraction is called an *M-fraction* after Murphy and Mc Cabe.

— *J-fractions*, introduced by Jacobi, are of the form

$$\frac{\alpha_1}{\beta_1+z} \;+\; \mathop{\mathrm{K}}_{m=2}^{\infty}\left(\frac{-\alpha_m}{\beta_m+z}\right), \quad \alpha_m \in \mathbb{C}\setminus\{0\}, \quad \beta_m \in \mathbb{C}.$$

— Thiele interpolating continued fractions are of the form

$$b_0 + \mathop{\mathrm{K}}_{m=1}^{\infty}\left(\frac{z-z_{m-1}}{b_m}\right), \quad b_m \in \mathbb{C}, \quad z_m \in \mathbb{C}.$$

A continued fraction $C(z)$ is called a *modified X-fraction* (with $X \in \{\mathrm{C,S,T,M,J}\}$) if there exist transformations $C(z) \to B(C(z))$ and $z \to a(z)$ such that the resulting continued fraction $B(C(a(z)))$ is an X-fraction.

A nonexhaustive list of special functions for which continued fraction expansions of the above types exist, is given in Table 1.

Running examples. The continued fraction

$$\exp(z) = 1 + \frac{2z}{2-z} \;+\; \frac{z^2/6}{1} \;+\; \mathop{\mathrm{K}}_{m=3}^{\infty}\left(\frac{a_m z^2}{1}\right), \quad z \in \mathbb{C} \tag{3}$$

with

$$a_m = \frac{1}{4(2m-3)(2m-1)}$$

is an S-fraction for the exponential function.

A modified C-fraction involving the complementary incomplete gamma function $\Gamma(a,z)$ is given by

$$\frac{\Gamma(a,z)}{z^a e^{-z}} = \frac{1}{z} \;+\; \mathop{\mathrm{K}}_{m=1}^{\infty}\left(\frac{m-a}{1} \;+\; \frac{m}{z}\right), \quad a \in \mathbb{C}, \quad |\operatorname{Arg} z| < \pi. \tag{4}$$

When adding the constraint $-\infty < a < 1$ the continued fraction in (4) becomes a modified S-fraction.

Table I. List of special functions (notation and name) and families of continued fractions as defined in section 3 (checkmark indicates availability). Information about the variable and parameter constraints can be found in the `constraints` argument.

Special function		Continued fraction				
notation	name	C	S	T	M	J
$\exp(z), \log(z), \sin(z), \ldots$	elementary	✓	✓	✓		
$\Gamma(a,z), \gamma(a,z)$	(compl.) incomplete gamma	✓	✓		✓	✓
$\psi_1(z)$	trigamma	✓	✓			✓
$\psi_2(z)$	tetragamma	✓	✓			
$\mathrm{erf}(z), \mathrm{dawson}(z)$	error, Dawson's integral	✓		✓		
$\mathrm{erfc}(z)$	complementary error		✓			✓
$w(z)$	complex error		✓			✓
$I^k \operatorname{erfc}(z)$	repeated integrals of erfc		✓			
$C(z), S(z)$	Fresnel integral	✓		✓		
$E_m(z)$	exponential integral	✓	✓		✓	✓
$\mathrm{Ei}(z), \mathrm{li}(z)$	exponential and logarithmic integral	✓	✓			
${}_2F_1(a,b;c;z)$	hypergeometric	✓		✓	✓	
${}_3F_2(a,b,c;d,e;z)$	ratios of hypergeometric					
${}_1F_1(a;b;z), M(a,b,z), U(a,b,z)$	confluent hypergeometric function	✓		✓	✓	
${}_2F_0(a,b;z)$	confluent hypergeometric series	✓				
${}_0F_1(;b;z)$	confluent hypergeometric limit	✓		✓		
$D_\nu(z)$	parabolic cylinder	✓				
$J_\nu(z), Y_\nu(z)$	Bessel		✓	✓		
$H_\nu^{(1)}(z), H_\nu^{(2)}(z)$	Hankel	✓				✓
$j_n(z), y_n(z)$	spherical Bessel/Hankel		✓	✓		
$I_\nu(z), K_\nu(z)$	modified Bessel	✓	✓	✓		✓
$(\log I_\nu(z))', (\log K_\nu(z))'$	log. derivative of mod. Bessel	✓	✓	✓		✓
$i_n(z), k_n(z)$	modified spherical Bessel		✓	✓		
$F(x,\mu,\sigma^2), Q(x,\mu,\sigma^2)$	normal distribution (cdf)	✓	✓			✓
$R(x)$	Mills ratio	✓	✓			
$I_k(x)$	repeated integrals of probability integral		✓			
$P(x^2;\nu), Q(x^2;\nu)$	chi-square distribution (cdf)	✓	✓		✓	
$P(x;\alpha;\theta), Q(x;\alpha;\theta)$	gamma distribution (cdf)	✓	✓		✓	
$I_x(a,b)$	incomplete beta distribution (cdf)	✓			✓	

4. THE CFSF PACKAGE

The CFSF package, developed in Maple, provides functionality for creating continued fractions, retrieving information about them and computing approximants with the possibility to make use of continued fraction tail estimates. A continued fraction expression can also be transformed into another form and it is possible to construct the continued fraction representation of a given series and vice versa.

We now discuss this functionality in more detail.

4.1 Creating continued fractions

The CFSF package supports any continued fraction that can be written in the form

$$b_0 + f\left(\frac{a_1}{b_1} + \cdots + \frac{a_n}{b_n} + \operatorname*{\mathbf{K}}_{\substack{k=0\\ \ell=n+1+kt}}^{\infty} \left(\frac{c_1(\ell)}{d_1(\ell)} + \cdots + \frac{c_t(\ell+t-1)}{d_t(\ell+t-1)}\right)\right). \quad (5)$$

Only the last part is compulsory. That is, a continued fraction may start with a *front term* b_0, it may contain a *factor* f and some non-general *begin elements* $[a_i, b_i]$ for $i = 1, \ldots, n$, but it must always be followed by a repetition of its *general*

elements $c_j(m)/d_j(m)$ for $j = 1, \ldots, t$.

Using the CFSF package, such a continued fraction can easily be created with the `create` command. This command takes several arguments, most of which are optional. Its first argument is compulsory and must be set to `contfrac`. Also required is the `general` argument which is assigned the list of the successive general elements $[c_j(m), d_j(m)]$ of (5). The optional front term is assigned to the `front` argument and defaults to 0 when missing. The optional factor is assigned to the `factor` argument, which defaults to 1 when missing. The optional `begin` argument is assigned the list of begin elements $[a_i, b_i]$.

For example, the continued fraction given in (3) can be constructed using the statement:

```
> expCF := create( contfrac,
    front = 1,
    begin = [ [2*z, 2-z], [z^2/6, 1] ],
    general = [ [z^2/(4*(2*m-3)*(2*m-1)), 1] ]
  );
```

$$expCF := formula\left(\left[\, type = contfrac,\ front = 1,\ begin = \left[[2z, 2-z], \left[\frac{z^2}{6}, 1\right]\right],\right.\right.$$
$$general = \left[\left[\frac{z^2}{4(2m-3)(2m-1)}, 1\right]\right],$$
$$\left.\left. variable = z,\ index = m \,\right]\right)$$

Note that not all arguments are explicitly specified in the `create` command. If missing, required arguments are given their default value. For instance, it is assumed by default that all parts of the formula (be it the `front`, `factor`, `begin` or `general` arguments) are expressions in the variable z. However, the user can assign another symbol to the `variable` argument. To replace the variable by an expression, see section 5.4. Likewise, it is assumed that all the general elements are written in function of the index m. This also can be changed by reassigning the `index` argument. Note that the running index m takes into account the number of begin elements.

The `create` command also supports parametrized functions as well as constraints. If the fraction is parametrized, the parameters must be identified by assigning their set to the `parameters` argument. Any constraints which must be satisfied for numerical computations, such as the convergence domain of the continued fraction or restrictions on its parameters, can be specified using the `constraints` argument. It must be a set with logical expressions involving the variable or the parameters, or items of the form `e::prop` where `e` is an expression containing the variable or one of the parameters and `prop` being a Maple property, which must be written in a form described in the `?property` help page of Maple.

Other optional arguments are `lhs`, `function`, `category`, `label` and `comment`. The `lhs` argument is used to store the left hand side $f(z)$ as in (1). The `function` argument can be used to identify the set of function names to which the continued fraction relates. Information about the family to which the continued fraction belongs can be stored as a string in the `category` argument. By assigning a string

to the `label` argument, a formula is identified for future reference. Finally, the `comment` argument allows adding a comment to the formula. None of these arguments influence any numerical computations.

Using all these extra arguments, one constructs the continued fraction for the complementary incomplete gamma function $\Gamma(a, z)$ from (4) with the following statement:

```
> CIGammaCF := create( contfrac,
    label = "CIGammaCF",
    factor = z^a * exp(-z),
    begin = [ [1,z] ],
    general = [ [(m/2)-a,1], [(m-1)/2,z] ],
    parameters = {a},
    function = GAMMA,
    lhs = GAMMA(a, z),
    category = "S-fraction",
    constraints = { abs(functions:-argument(z)) < Pi }
  );
```

$$CIGammaCF := formula(\text{“CIGammaCF”});$$

Here, the denominator from the left hand side of (4) has been incorporated into the `factor` argument.

Note that, by specifying the `label` argument, the formula is added to the list of predefined formulas (provided that no other formula with such a label already exists). From then on, it is printed in short form, showing only its label. Together with the `function` and `category` arguments, this makes reusing and retrieving the formula at a later stage possible using the `formula` and `query` commands, respectively. This functionality is discussed in section 6.

4.2 The underlying structure

The CFSF library defines the new type `formula` that can be used to check whether a given formula is correctly constructed. For example, to check whether `expCF` is a valid formula, one uses the statement:

```
> type( expCF, formula );
```

$$true$$

Type checking is handled by an internal routine, called `validate`, which performs several checks to make sure that all parts of the formula are well formed. For instance, this routine checks whether the `begin` and `general` arguments contain items that conform to the Maple structured type `list([exprtype,exprtype])`, where `exprtype` is defined in the CFSF library as

```
exprtype := { '+', '*', '^', function, symbol, complex(numeric) };
```

The underlying structure of a formula is a table in which the argument names have been used as indices. As such, its content fully resembles the call to the `create` command, except for the first argument `contfrac` which is assigned to the `type` index. Using a table makes retrieving information on a formula very easy, as is shown in the next section.

4.3 Retrieving information on a continued fraction

One can easily check which parts of the general form (5) are specified by using the Maple functions `assigned` and `indices`. For example:

```
> assigned( CIGammaCF[front] );
```

$$false$$

```
> assigned( CIGammaCF[begin] );
```

$$true$$

```
> map( op, { indices(CIGammaCF) } );
```

$$\{\, type,\ factor,\ begin,\ general,\ parameters,\ constraints,\ lhs,\ function,\ category,\ label,\ variable,\ index \,\}$$

The last statement returns the set of all arguments that are specified for `CIGammaCF`. Accessing the individual parts is done using the subscript selector:

```
> CIGammaCF[general];
```

$$\left[\left[\frac{m}{2} - a, 1 \right], \left[\frac{m-1}{2}, z \right] \right]$$

The commands `nthpnumer`, `nthpdenom` and `nthpelement` can be used to get the N-th partial numerator, denominator or element respectively. These commands take two arguments, namely a formula as its first argument and a positive integer value for N as its second argument. For example, the next statement gives the fourth partial numerator of (4):

```
> nthpnumer( CIGammaCF, 4 );
```

$$2 - a$$

If the second argument to these commands is a symbol, the statement is returned unevaluated. The evaluation is deferred until a numeric value for the second argument is known.

Using Maple's `seq` command, one can retrieve the sequence with the first 6 elements of (4) as follows:

```
> seq( nthpelement( CIGammaCF, i ), i=1..6 );
```

$$[z^a e^{-z}, z],\ [1-a, 1],\ [1, z],\ [2-a, 1],\ [2, z],\ [3-a, 1]$$

Note that the factor has been incorporated into the first element, that is, the first element in this sequence is actually of the form $[fa_1, b_1]$.

One gets information about the domain of convergence of (4) by looking at its constraints:

```
> CIGammaCF[constraints];
```

$$\{\, |functions{:-}argument(z)| < \pi \,\}$$

This constraint indicates that the continued fraction (4) converges for all $z \in \mathbb{C}\backslash\{0\}$ with $|\operatorname{Arg}(z)| < \pi$ (see also section 6.3). It is important to realize that the use of $functions{:-}argument$ implies that all functions of a complex variable are treated as single-valued.

4.4 Computing classical approximants

Evaluating a continued fraction means truncating expression (1) after a finite number of elements, say N, and evaluating the resulting *classical approximant* (also called *convergent*), which is of the form

$$f_N(z) := b_0 + \mathop{\mathrm{K}}_{m=1}^{N} \left(\frac{a_m}{b_m} \right). \tag{6}$$

Such approximants can be computed using the `nthapprox` command. This command takes at least two arguments, namely a formula as its first argument and a positive number that specifies which approximant to compute. When evaluating a continued fraction that has been parametrized, one can also pass a set of parameter substitutions as a third argument to the `nthapprox` command (for all parameters specified by the `parameters` argument in the `create` command).

For example, the following statement computes the fifth approximant of (4) for $\Gamma(1/2, z)$ symbolically:

```
> nthapprox( CIGammaCF, 5, { a = 1/2 } );
```

$$\frac{2\left(2\,z^2+9\,z+4\right)e^{-z}}{\left(4\,z^2+20\,z+15\right)\sqrt{z}}$$

The approximant is computed using the backward recurrence scheme

$$\begin{aligned} F_{N+1} &= 0, \\ F_i &= a_i/(b_i + F_{i+1}), \qquad i = N, \ldots, 1, \\ F_0 &= b_0 + F_1. \end{aligned} \tag{7}$$

When a symbolic result is requested, the intermediate results F_i are occasionally normalized to avoid excessive memory usage when Maple tries to simplify the final result F_0.

The N-th numerator A_N and the N-th denominator B_N of (1) are defined by the recurrence relations

$$\begin{bmatrix} A_n \\ B_n \end{bmatrix} := b_n \begin{bmatrix} A_{n-1} \\ B_{n-1} \end{bmatrix} + a_n \begin{bmatrix} A_{n-2} \\ B_{n-2} \end{bmatrix}, \qquad n = 1, 2, \ldots, N,$$

with initial conditions

$$A_{-1} := 1, \quad B_{-1} := 0, \quad A_0 := b_0, \quad B_0 := 1.$$

Note that $f_N(z) = A_N/B_N$. They can be computed using the commands `nthnumer` and `nthdenom`, which take the same arguments as the `nthapprox` command. Like `nthapprox`, these commands occasionally normalize their intermediate results.

For (4), this gives:

```
> nthnumer( CIGammaCF, 5, { a = 1/2 } );
```

$$\frac{1}{2}\frac{\left(2\,z^2+9\,z+4\right)\sqrt{z}}{e^z}$$

```
> nthdenom( CIGammaCF, 5, { a = 1/2 } );
```

$$z^3 + 5\,z^2 + \frac{15}{4}\,z$$

To compute a numerical evaluation, an argument of the form *variable* = *value* must be provided as the last argument to the `nthapprox`, `nthnumer` or `nthdenom` commands. In this case, the same recurrence relations are used, but the normalization process is replaced by a floating-point evaluation at every step (`evalf`), where the Maple environment variable `Digits` determines the decimal precision in which the calculation is performed. Doing an evaluation at every step is much faster than first having to compute the symbolic result separately and then substituting the numeric values.

Computing the 10-th approximant of (3) for $z = 1.0$ in 40 digit decimal arithmetic gives:

```
> Digits := 40:
> nthapprox( expCF, 10, z = 1.0 );
```

$$2.7182818284590452353602871799000862593 51$$

which has 25 significant digits. In contrast, the partial sum of degree 10 of the exponential series only has 8 significant digits.

During a numerical evaluation, it is also checked whether the constraints (as specified by the `constraints` argument to the `create` command) are satisfied. If a constraint involving a parameter is violated, an error is generated. On the other hand, if a constraint involving the variable is violated, a warning message is printed, and the calculation proceeds as usual. This allows one to investigate the behavior of a continued fraction outside its domain of convergence. Note, however, that the result is not guaranteed in any way.

For example, when evaluating the fifth approximant of (4) to obtain an approximation to $\Gamma(1/2, -1.5)$, one gets:

```
> nthapprox( CIGammaCF, 5, { a = 1/2 }, z = -1.5 );

Warning, constraint 'abs(functions:-argument(z)) < Pi'
is not satisfied
Warning, subsequent results are not guaranteed
```

$$-6.0988063378536483210631690576692184127 62\,\mathrm{I}$$

4.5 Tails and modifications

Next to computing the approximant $f_N(z)$ as defined in (6), one is often interested in the truncated part as well. This truncated part of a continued fraction expansion, which is of the form

$$t_N(z) := \operatorname*{\mathrm{K}}_{m=N+1}^{\infty} \left(\frac{a_m}{b_m} \right) \tag{8}$$

is called the N-th tail.

Such tails can easily be constructed with the `nthtail` command. For example, the fifth tail of (4) can be computed with the following statement:

```
> nthtail( CIGammaCF, 5 );
```

$$formula\left(\left[\, type = contfrac,\ variable = z,\ index = m,\right.\right.$$
$$general = \left[\left[\frac{m+5}{2} - a, 1\right], \left[\frac{m+1}{2}, z\right]\right],\ parameters = \{a\},$$
$$\left.\left. comment = \text{“5-th tail of “CIGammaCF””}\ \right]\right)$$

Note that the `comment` argument of the newly created formula contains information on how it was constructed.

Computing classical approximants means replacing the continued fraction tails with zero. However, unlike for series, these tails do not need to converge to zero as $N \to \infty$ [Lorentzen and Waadeland 1992, ch. II]. Take for instance the continued fraction expansion

$$\frac{\sqrt{1+4x}-1}{2} = \mathop{\mathrm{K}}_{n=1}^{\infty}\left(\frac{x}{1}\right), \qquad x \geq -\frac{1}{4}.$$

Each tail t_N converges to $(\sqrt{1+4x}-1)/2$ as well. More remarkable is that the even-numbered tails of the convergent continued fraction

$$\sqrt{2}-1 = \mathop{\mathrm{K}}_{n=1}^{\infty}\left(\frac{(3+(-1)^n)/2}{1}\right) = \frac{1}{1} \;+\; \frac{2}{1} \;+\; \frac{1}{1} \;+\; \frac{2}{1} \;+\; \cdots$$

converge to $\sqrt{2}-1$ while the odd-numbered tails converge to $\sqrt{2}$ (hence the sequence of tails does not converge), and that the sequence of tails $\{t_N\}_{N\geq 1} = \{N+1\}_{N\geq 1}$ of

$$1 = \mathop{\mathrm{K}}_{n=1}^{\infty}\left(\frac{n(n+2)}{1}\right)$$

converges to $+\infty$.

Hence a more accurate approximant than the classical $f_N(z)$ is obtained if the N-th tail is replaced with some suitable value w_N, called a *modification*. That is, instead of computing the classical approximant (6), one can also compute a *modified approximant* [Lorentzen and Waadeland 1992, p. 20], defined by

$$f_N(z; w_N) := b_0 + \mathop{\mathrm{K}}_{m=1}^{N-1}\left(\frac{a_m}{b_m}\right) \;+\; \frac{a_N}{b_N + w_N}. \tag{9}$$

A proper choice of a modification for a limit-periodic continued fraction of the form $\mathrm{K}(a_m/1)$ depends on the limiting behavior of its partial numerators a_m. One considers the following cases:

— When

$$\lim_{m\to\infty} a_m = a \neq \infty, \quad a \notin (-\infty, -1/4),$$

one can replace the N-th tail by the value of the periodic continued fraction $\mathrm{K}(a/1)$, namely

$$w_N = w := \frac{a}{1} \;+\; \frac{a}{1} \;+\; \frac{a}{1} \;+\; \ldots = \frac{\sqrt{4a+1}-1}{2}, \tag{10}$$

where we take $\sqrt{4a+1}$ with $\Re(\sqrt{4a+1}) > 0$.

— If furthermore

$$r = \lim_{k\to\infty} \frac{a_{k+1}-a}{a_k-a}$$

exists, then one can use [Jacobsen and Waadeland 1988]

$$w_N := w + \frac{a_{N+1}-a}{1+(r+1)w}. \tag{11}$$

— When

$$\lim_{m\to\infty} a_m = \infty, \quad a_{N+1} \notin (-\infty, -1/4),$$

one can choose

$$w_N := \frac{a_{N+1}}{1} + \frac{a_{N+1}}{1} + \frac{a_{N+1}}{1} + \ldots = \frac{\sqrt{4a_{N+1}+1}-1}{2}. \tag{12}$$

where we take $\sqrt{4a_{N+1}+1}$ with $\Re(\sqrt{4a_{N+1}+1}) > 0$.

For a continued fraction of the form $\mathrm{K}(c_m/d_m)$, a suitable modification is obtained by setting $\tilde{w}_N := d_N w_N$ for $N \geq 1$ where w_N is a modification for the equivalent continued fraction $\mathrm{K}(a_m/1)$.

The `tailestimate` command is used to automatically compute a modification w_N for a given continued fraction. To determine which case applies from the list given above, this command makes use of the `assuming` facility provided by Maple. That is, it computes the limiting behavior of the partial numerators a_m under the assumption that the constraints of the given formula are satisfied. When specifying `improved` as the last argument of `tailestimate`, the command tries to compute the improved estimate (11) in case it applies.

For example, applying this to the continued fraction given in (3) for the exponential function, one gets:

```
> w := tailestimate( expCF, improved );
```

$$w := \mathit{nthdenom}\left(\mathit{expCF}, m\right) \mathit{nthpnumer}\left(\mathit{transform}\left(\mathit{expCF}, \mathit{simregular}\right), m+1\right).$$

Subsequently, this result can be assigned to the `modification` argument of the `nthnumer`, `nthdenom` or `nthapprox` commands, in order to compute a modified numerator, denominator or approximant, respectively. This amounts to starting the backward recurrence (7) with $F_{N+1} = w$ instead of the classical $F_{N+1} = 0$. For example, the modified approximant can now be computed as follows:

```
> nthapprox( expCF, 10, modification = w, z = 1.0 );
```

$$2.71828182845904523536028747150335798417 1$$

This value has 28 significant digits, while the classical approximant, computed in section 4.4, is correct to only 25 digits.

Sometimes, `tailestimate` initially fails in computing a modification. In this case, it helps to introduce an additional constraint to the `constraints` argument of the given formula. One example is to add a constraint of the form `z::positive`, indicating that one is only interested in the positive real case (with z being the variable of the given formula). Unfortunately, due to the nature and limitations

of the `assuming` facility as well as the large amount of possibilities for specifying the constraints, it is not always possible to automatically compute a symbolic modification.

For numerical computations, the `tailestimate` command can also be assigned directly to the `modification` argument of the `nthnumer`, `nthdenom` or `nthapprox` commands. Since in numerical computations, values are known for all the parameters as well as for the variable of a given formula, the `tailestimate` command has no problem in automatically computing the modification. For example, the following statement computes the fifth modified approximant of (4) with $a = 1/2$ and $z = 2 + 3\,\mathrm{I}$ in 20 digit decimal arithmetic:

```
> evalf[20]( nthapprox( CIGammaCF, 5, { a = 1/2 },
  modification = tailestimate(), z = 2+3*I ) );
```

$$-0.063490399330654569450 + 0.017403965788340933772\,\mathrm{I}$$

5. EQUIVALENCE TRANSFORMATIONS AND CONTRACTIONS

For many special functions or constants, several continued fraction expansions exist, with some having all numerators or denominators equal to 1, some equivalent to a power series, and so on. Here we give some commands to transform between expansions or continued fraction representations.

Transformations are automatically computed with the `transform` command. This command takes a formula as its first argument and the name of the requested transformation as its second argument (which can be any of `simregular`, `even_contraction`, `odd_contraction` or `Euler`).

5.1 Simregular equivalence transformation

Two continued fractions $b_0 + \mathrm{K}(a_m/b_m)$ and $d_0 + \mathrm{K}(c_m/d_m)$ are called *equivalent* if they have the same sequence of approximants. This equivalence holds if and only if there exist complex numbers $\{r_m\}$ with $r_0 = 1$ and $r_m \neq 0$ for $m \geq 1$ such that

$$d_0 = b_0, \quad c_m = r_m r_{m-1} a_m, \quad d_m = r_m b_m.$$

In particular, if $b_m \neq 0$ for $m \geq 1$, one can take $r_m = 1/b_m$ so that $d_m = 1$ and

$$c_1 = \frac{a_1}{b_1}, \quad c_m = \frac{a_m}{b_m b_{m-1}}, \quad m \geq 2,$$

to get the equivalent fractions

$$b_0 + \operatorname*{K}_{m=1}^{\infty} \left(\frac{a_m}{b_m} \right) \equiv b_0 + \operatorname*{K}_{m=1}^{\infty} \left(\frac{c_m}{1} \right) = \frac{a_1/b_1}{1} \; + \; \frac{a_2/(b_2 b_1)}{1} \; + \; \ldots . \tag{13}$$

A continued fraction in which all partial denominators are equal to 1, is said to be in *simregular* form.

For example, the following statement constructs a continued fraction of the form (13) for the continued fraction given in (4):

```
> CIGammaCF_simregular := transform( CIGammaCF, simregular );
```

$$
\begin{aligned}
&\mathit{formula}\Bigg(\Bigg[\, \mathit{type} = \mathit{contfrac},\ \mathit{parameters} = \{a\}\,,\\
&\mathit{lhs} = \Gamma(a,z),\ \mathit{factor} = z^a e^{-z},\ \mathit{begin} = \left[\left[\frac{1}{z},1\right],\left[\frac{1-a}{z},1\right]\right],\\
&\mathit{general} = \left[\left[\frac{(m-1)/2}{z},1\right],\left[\frac{m/2-a}{z},1\right]\right],\\
&\mathit{constraints} = \{\,|\mathit{functions}\text{:-}\mathit{argument}(z)| < \pi\,\},\\
&\mathit{variable} = z,\ \mathit{index} = m,\\
&\mathit{comment} = \text{“simregular form of “CIGammaCF””}\ \Bigg]\Bigg).
\end{aligned}
$$

The equivalence of two continued fractions can be checked using the `equivalent` function. This function checks if their sequence of approximants are the same. Since we are dealing with limit-periodic continued fractions, this comes down to comparing all approximants up to the least common multiple of the numbers of general elements increased with the maximum number of begin elements. For instance, to check whether `CIGammaCF` and `CIGammaCF_simregular` are equivalent, one uses the following statement:

```
> equivalent( CIGammaCF, CIGammaCF_simregular );
```

$$\mathit{true}$$

This functionality allows to compare newly discovered or constructed fractions versus existing ones.

5.2 Even and odd contractions

Another type of transformations can be obtained using contractions. A continued fraction $d_0 + \mathrm{K}(c_m/d_m)$ is called a *contraction* of another continued fraction $b_0 + \mathrm{K}(a_m/b_m)$ if and only if its approximants are a subset of the approximants of the latter. Conversely, the continued fraction $b_0 + \mathrm{K}(a_m/b_m)$ is then called an *extension* of $d_0 + \mathrm{K}(c_m/d_m)$. If in particular $b_{2k} \neq 0$ for $k \geq 1$, then an even contraction of $b_0 + K(a_m/b_m)$ is given by

$$
\begin{aligned}
d_0 + \underset{m=1}{\overset{\infty}{\mathrm{K}}}\left(\frac{c_m}{d_m}\right) = b_0 + \frac{a_1 b_2}{b_1 b_2 + a_2} \;-\; &\frac{a_2 a_3 b_4/b_2}{a_4 + b_3 b_4 + a_3 b_4/b_2}\\
&- \frac{a_4 a_5 b_6/b_4}{a_6 + b_5 b_6 + a_5 b_6/b_4} \;-\; \ldots
\end{aligned}
\tag{14}
$$

with

$$
\begin{aligned}
&d_0 = b_0, \quad c_1 = a_1 b_2, \quad d_1 = a_2 + b_1 b_2,\\
&c_m = -\frac{a_{2m-2} a_{2m-1} b_{2m}}{b_{2m-2}}, \quad m \geq 2,\\
&d_m = a_{2m} + b_{2m-1} b_{2m} + \frac{a_{2m-1} b_{2m}}{b_{2m-2}}, \quad m \geq 2.
\end{aligned}
$$

An odd contraction can be constructed in a similar way (under the condition that $b_{2k+1} \neq 0$).

For example, an even contraction of (4) can be constructed using the statement:

```
> CIGammaCF_even := transform( CIGammaCF, even_contraction );
```

$$\begin{gathered} \mathit{formula}\big(\big[\,\mathit{type} = \mathit{contfrac},\ \mathit{parameters} = \{a\}\,, \\ \mathit{lhs} = \Gamma(a,z),\ \mathit{begin} = [[z^a e^{-z}, 1-a+z]], \\ \mathit{general} = [[-(m-1-a)\,(m-1)\,, 2\,m-a+z-1]], \\ \mathit{constraints} = \{\,|\mathit{functions}\text{:-}\mathit{argument}(z)| < \pi\,\}, \\ \mathit{variable} = z,\ \mathit{index} = m, \\ \mathit{comment} = \text{“even contraction of “CIGammaCF””}\,\big]\big). \end{gathered}$$

This is a real J-fraction with a single general element.

Computing an even (or odd) contraction means constructing new elements by combining elements from the original formula. For the even contraction in (5.2), its elements c_m/d_m are expressions which involve the three successive elements a_{m-2}/b_{m-2}, a_{m-1}/b_{m-1} and a_m/b_m from the original formula. If a continued fraction has an even number of general elements, then its even contraction has only half that number of general elements. If on the other hand a continued fraction has an odd number of general elements, then its even contraction has the same amount of general elements.

5.3 Connection with series

The `transform` command can also be used to construct the continued fraction representation of a given series and vice versa. This transformation is often called the Euler transformation or representation. Given a sequence $\{c_k\}_k$ in $\mathbb{C} \setminus \{0\}$ such that

$$f_N := \sum_{k=0}^{N} c_k, \tag{15}$$

there exists a continued fraction $b_0 + \mathrm{K}(a_m/b_m)$ of the form

$$\begin{gathered} b_0 = c_0, \quad a_1 = c_1, \quad b_1 = 1, \\ a_m = -\frac{c_m}{c_{m-1}}, \quad b_m = 1 + \frac{c_m}{c_{m-1}}, \quad m \geq 2, \end{gathered}$$

which has f_N as its N-th approximant for all N.

For example, if `expSeries` is the McLaurin series of $\exp(z)$, then the Euler representation can be computed with

```
> expEuler := transform( expSeries, Euler );
```

$$\begin{gathered} \mathit{formula}\Bigg(\Bigg[\,\mathit{type} = \mathit{contfrac},\ \mathit{front} = 1,\ \mathit{variable} = z,\ \mathit{index} = k, \\ \mathit{lhs} = e^z,\ \mathit{function} = \exp,\ \mathit{begin} = [[z,1]],\ \mathit{general} = \left[\left[-\frac{z}{k}, \frac{z+k}{k}\right]\right], \\ \mathit{comment} = \text{“Euler form of “expSeries””}\,\Bigg]\Bigg) \end{gathered}$$

Conversely, for a continued fraction $b_0 + \mathrm{K}(a_m/b_m)$ with finite approximants f_N, the sequence $\{c_k\}_k$ defined by

$$c_0 = b_0, \quad c_k = \frac{(-1)^{k-1} \prod_{j=1}^{k} a_j}{B_k B_{k-1}}, \quad k \geq 1$$

where B_k denotes the k-th denominator of the continued fraction, satisfies (15). Since this connection between series and continued fractions implies the same convergence or divergence behavior, it is only of limited interest.

5.4 Variable transformation

Finally, a variable transformation $z \to T(z)$ can be incorporated by using the `varsubs` command. Given a formula as its first argument and an argument of the form *variable = expression* as its second argument, this command returns a new formula where all occurrences of the variable (except for the variable itself) are substituted with the supplied expression. The newly created formula remains being defined in function of the original variable. This allows the evaluation of the new continued fraction without having to explicitly repeat the expression every time.

For instance, one can apply the variable transformation $T(z) = -z^2 + 3$ to `CIGammaCFeven` using the statement:

```
> varsubs( CIGammaCF_even, z=-z^2+3 );
```

$$\begin{aligned} &\mathit{formula}\Bigg(\Bigg[\, \mathit{type} = \mathit{contfrac},\ \mathit{parameters} = \{a\}\,, \\ &\mathit{lhs} = \Gamma(a,z),\ \mathit{begin} = \left[\left[\left(-z^2+3\right)^a e^{z^2-3}, 4-a-z^2\right]\right], \\ &\mathit{general} = \left[\left[-\left(m-1-a\right)\left(m-1\right), 2\,m+2-a-z^2\right]\right], \\ &\mathit{variable} = z,\ \mathit{index} = m\Bigg]\Bigg) \end{aligned}$$

6. A LIBRARY OF CONTINUED FRACTIONS

Besides providing all the functionality needed to create continued fractions and to compute their approximants, the CFSF package also includes a library of continued fractions and series for a lot of special functions. Table 1 contains a list of special functions and the available continued fraction representations, classified in families as given in section 3.

The CFSF package provides two commands, `query` and `formula`, for querying this list and retrieving the predefined continued fractions or series. Both commands use the `function`, `category` and `label` arguments introduced in section 4.1.

6.1 Querying the library

Using the `query` command, one can search the library for continued fractions or series of a specific special function. The name of the function is assigned to the `function` parameter, while the continued fraction family is assigned to the `category` parameter. When executed, this command returns a set of `label` arguments for the formulas (continued fractions and/or series) which satisfy the search. These labels refer to the full list of formulas at `www.cfsf.ua.ac.be`.

For example, the set of all library formula labels related to the gamma and incomplete gamma functions can be retrieved by using the following statement:

```
> query( function = GAMMA );
```

{"GA.lincgamma.cfrac.01", "GA.lincgamma.power.02",
"GA.lincgamma.mfrac.01", "GA.uincgamma.asymp.01",
"GA.uincgamma.cfrac.02", "GA.uincgamma.jfrac.02",
"GA.uincgamma.sfrac.01", "GA.uincgamma.jfrac.01",
"GA.lincgamma.power.01", "GA.uincgamma.mfrac.01",
"GA.uincgamma.cfrac.01"}

Additionally, one can restrict this set to include only those labels that correspond to S-fractions by adding the `category` parameter to the previous statement:

```
> query( function = GAMMA, category = "S-fraction" );
```

{"GA.uincgamma.sfrac.01"}

If neither the `function` nor the `category` parameter is specified, the set of labels for all predefined formulas is returned.

6.2 Retrieving formulas

To retrieve a formula from the library, the `formula` command is used. It is given a label as its first argument. For example, one can retrieve the formula with label `GA.uincgamma.sfrac.01` using the following statement:

```
> f := formula( "GA.uincgamma.sfrac.01" );
```

$$f := \mathit{formula}(\text{"GA.uincgamma.sfrac.01"});$$

This label corresponds to the modified S-fraction of the incomplete gamma function, obtained from (4) by adding the constraint $-\infty < a < 1$. To see the full definition of this formula one uses the `eval` command, which returns the underlying table structure:

```
> eval(f);
```

$$\begin{gathered}
\mathit{table}\big(\big[\, \mathit{type} = \mathit{contfrac},\ \mathit{parameters} = \{a\},\ \mathit{factor} = z^a e^{-z},\\
\mathit{begin} = [[1, z]]\,,\ \mathit{general} = [[m/2 - a, 1]\,, [(m-1)/2, z]]\,,\\
\mathit{function} = \{\, \mathit{functions}{:-}\mathit{uincgamma}, \Gamma\,\},\\
\mathit{lhs} = \mathit{functions}{:-}\mathit{uincgamma}(a, z),\ \mathit{category} = \text{"S} - \text{fraction"},\\
\mathit{constraints} = \{\, -\infty < a \text{ and } a < 1,\ |\mathit{functions}{:-}\mathit{argument}(z)| < \pi\,\},\\
\mathit{variable} = z,\ \mathit{index} = m,\ \mathit{label} = \text{"GA.uincgamma.sfrac.01"}\,\big]\big)
\end{gathered}$$

The `formula` command also allows parameters to be substituted immediately when retrieving the formula. This is done by adding a set of parameter substitutions as a second argument. If a given parameter substitution violates a constraint, an error is raised. For example, one can replace a by $1/2$ in the continued fraction with label `GA.uincgamma.sfrac.01` for $\Gamma(1/2, z)$ through:

```
> g := formula( "GA.uincgamma.sfrac.01", { a = 1/2 } );
```

$$table\big(\big[\, type = contfrac,\, factor = \sqrt{z}e^{-z},$$
$$begin = [[1, z]]\,,\, general = [[m/2 - 1/2, 1]\,, [m/2 - 1/2, z]]\,,$$
$$function = \{functions{:-}uincgamma, \Gamma\,\},$$
$$lhs = functions{:-}uincgamma(1/2, z),\, category = \text{“S} - \text{fraction”},$$
$$constraints = \{\,|functions{:-}argument(z)| < \pi\,\},$$
$$variable = z,\, index = m\,\big]\big)$$

6.3 The submodule CFSF:−functions

Note that the `function` argument of the previous formula is a set of two functions. The first function, $functions{:-}uincgamma$, is defined as an alias for $\Gamma(\alpha, z)$ in the submodule CFSF:−functions. Adding functions to the `function` argument makes the `query` command more flexible. For example, one retrieves only those formulas that are related to $functions :- uincgamma$, which is a subset of the previous `query` statement, by using:

```
> query( function = functions:-uincgamma );
```

{“GA.uincgamma.mfrac.01”, “GA.uincgamma.jfrac.02”,
“GA.uincgamma.asymp.01”, “GA.uincgamma.cfrac.02”,
“GA.uincgamma.sfrac.01”, “GA.uincgamma.jfrac.01”,
“GA.uincgamma.cfrac.01”}

The submodule CFSF:−functions also defines functions for which no function exists directly in Maple. For example, the $functions{:-}lincgamma$ function is defined by

$$functions{:-}lincgamma := (\alpha, z) \to \Gamma(\alpha) - \Gamma(\alpha, z).$$

Since the Maple function `argument` incorrectly returns 0 for Arg(0), a correct implementation for Arg(z) is provided as $functions :- argument$ which returns `undefined` when $z = 0$.

7. SUMMARY AND FUTURE WORK

In total 220 series and continued fraction representations for the special functions listed in Table 1 are available. We have outlined how these representations can be retrieved and which functionality is offered in the CFSF package. A full list of the 220 implemented formulas with their labels is shown on `www.cfsf.ua.ac.be`.

The project is an ongoing project and future work includes the inventory and implementation of representations for the Coulomb wave functions, the Legendre functions, the Riemann zeta function and other number theoretic functions. As the work evolves, the webpage will be updated.

Acknowledgement. The authors are very grateful to Stefan Becuwe for transferring these 220 formulas from the continued fractions handbook [Cuyt et al. 2008] to the CFSF package. He has also thoroughly tested the package and verified all implemented series and continued fraction representations.

REFERENCES

Abramowitz, M. and Stegun, I. 1964. *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables*. U.S. Government Printing Office, NBS, Washington, D. C.

Cuyt, A., Brevik Petersen, V., Verdonk, B., Waadeland, H., and Jones, W. 2008. *Handbook of Continued Fractions for Special Functions*. Springer Verlag.

Cuyt, A., Verdonk, B., and Waadeland, H. 2006. Efficient and reliable multiprecision implementation of elementary and special functions. *SIAM J. Sci. Comput. 28*, 1437–1462.

Erdélyi, A., Magnus, W., Oberhettinger, F., and Tricomi, F. 1953a. *Higher transcendental functions*. Vol. 1. McGraw-Hill, New York.

Erdélyi, A., Magnus, W., Oberhettinger, F., and Tricomi, F. 1953b. *Higher transcendental functions*. Vol. 2. McGraw-Hill, New York.

Erdélyi, A., Magnus, W., Oberhettinger, F., and Tricomi, F. 1955. *Higher transcendental functions*. Vol. 3. McGraw-Hill, New York.

Jacobsen, L. and Waadeland, H. 1988. Convergence acceleration of limit periodic continued fractions under asymptotic side conditions. *Numer. Math. 53*, 285–298.

Lorentzen, L. and Waadeland, H. 1992. *Continued fractions with applications*. North-Holland Publishing Co., Amsterdam.

Lozier, D. 2000. The DMLF project: A new initiative in classical special functions. In *Special Functions: Proceedings of the International Workshop (Hong Kon, 1999)*, C. Dunkl, M. Ismail, and R. Wong, Eds. World Scientific, Signapore, 207–220.