Package ‘contFracR’

August 19, 2023

Type Package

Title Continued Fraction Generators and Evaluators

Version 1.2.1

Date 2023-08-19

Author Carl Witthoft

Maintainer Carl Witthoft <cellocgw@gmail.com>

Description Converts numbers to continued fractions and back again. A solver for Pell's Equation is provided. The method for calculating roots in continued fraction form is provided without published attribution in such places as Professor Emeritus Jonathan Lubin, <http://www.math.brown.edu/jlubin/> and his post to StackOverflow, <https://math.stackexchange.com/questions/2215918> , or Professor Ron Knott, e.g., <https://r-knott.surrey.ac.uk/Fibonacci/cfINTRO.html> .

License LGPL-3

Imports Rmpfr, go2bigq, gmp, methods

NeedsCompilation no

Repository CRAN

Date/Publication 2023-08-19 15:32:33 UTC

R topics documented:

`contFracR-package` *Functions to generate and evaluate continued fractions*

Description

This package includes functions to convert a rational number to its continued fraction form, to convert any defined continued fraction to a numeric value, and to generate estimates of some common irrationals (e, pi, phi) via continued fraction representations. There's also a function which generates LaTeX code to draw the continued fraction itself.

Author(s)

Carl Witthoft, with attributions as noted in the individual help pages

Maintainer:Carl Witthoft carl@witthoft.com

`cfrac2latex` *Function which builds the LaTeX formula to display the continued fraction*

Description

Take the vectors of numerators and denominators and create the LaTeX formula which will draw the "cascading" continued fraction.

Usage

```
cfrac2latex(denomvals, numvals = 1, denrepeats = 1, doellipsis = FALSE,  ...)
```

Arguments

`denomvals` A vector containing the denominator values, starting with the not-actually-denominator integer part of the continued fraction. If the denominator has a repeat sequence, specify the number of repeats using the argument `denrepeats` . See the Details section for more information.

`numvals` A vector containing the numerator values. If the length of this vector is less than the length of the (perhaps repeated) `denomvals`, it will be recycled as necessary.

`denrepeats` If the denominator sequence has a repeat pattern, repeat that pattern this many times. See the Details section for more information.

`doellipsis` If TRUE, an ellipsis is printed in the deepest denominator, indicating an infinite sequence continues. If FALSE, not added.

`...` Reserved for future upgrades

Details

The standard notation for a continued fraction defines a sequence [a0; b1, b2, b3,...bn] indicating the formula is x = a0 + 1/(b1 + 1/(b2 + 1/(b3...))) (or replace the numerators with a specified sequence of values of length n). To save input effort, if there's a repeat pattern, e.g. [a0,b1,b2,b3,b1,b2,b3,...] then the user can enter the vector `denomvals = c(a0,b1,b2,b3)` and enter the desired number of repeats with the argument `denrepeats`.

Value

Three versions are returned. `texeqn` contains a text string compatible with the R-console's parser. Use this to write the correct LaTeX string to a file, or to pass it to tools such as `title`. `tex2copy` can be used if you wish to use Select/Copy/Paste operations on the text displayed in the console. Do not enter this into any R command, as the parser will interpret the backslashes as escape characters, e.g. "backslash-f" turns into a newline. Finally, eqn returns an ASCII-only string along the lines of "1/(1+1/(2+1/(3+...)))" .

Author(s)

Carl Witthoft, `<carl@witthoft.com>`

Examples

```
cfrac2latex( 1:5,1, doellipsis= FALSE)
#$eqn
#[1] "1 + 1/(2 + 1/(3 + 1/(4 + 1/5 ...)))"
#$tex2copy
#[1] "1 + \frac{1}{2 + \frac{1}{3 + \frac{1}{4 + \frac{1}{5 ...}}}}"

# Notice the additional backslashes to make the console parser happy.
#$texeqn
#[1] "1 + \frac{1}{2 + \frac{1}{3 + \frac{1}{4 + \frac{1}{5 ...}}}}"
```

`cfrac2num` *Function to convert a defined set of numerators and denominators into the equivalent number.*

Description

Given a vector of denominators and optionally a vector of numerators, calculate the `bigq` fraction and the `mpfr` extended precision decimal value.

Usage

```
cfrac2num(denom, num = 1, init = 0, nreps= 1, msgrate = NULL )
```

Arguments

`denom` A vector in standard continued fraction form `[b0; a1,a2,a3,...]` where `b0` represents the integer part of the number and the `a_j` are the denominator terms. If all terms, including `b0` , are the same, optionally enter a single value and specify the length with the `numterms` value

`num` A vector of numerator values. If all numerators are the same, a single value will suffice. The default is `1`

`init` In those cases where the denominator values repeat (e.g., for continued fraction form of square roots), it is possible to take the output of a previous run and use it to initialize a new run. This input must be a `bigq` fraction. Use with care.

`nreps` If `denoms` is a single value, repeat that value `nreps` times to set up the continued fraction. Otherwise, assume a periodic denominator sequence was submitted, and repeat the `denom[-1]` section `nreps` times (the first denominator is the integer term) .

`msgrate` If desired enter an integer indicating how often a status message should be displayed. Leave as `NULL` to suppress messages. For example, setting thi to 10 will generate a message every 10 sub-fraction reductions.

Details

All calculations are done with `bigq` fractions to preserve full precision. Naturally, a finite input will not yield the exact value for any irrational number.

Value

The exact numeric value is provided in `cgmp`, a `bigq` fraction. The `mpfr` value is calculated using `.bigq2mpfr`, which generates a decimal number of sufficient precision to match the implied precision of the fraction. `fracpart` equals `cgmp` minus the integer part, if any. `denom` and `num` provide the vectors of denominator and numerator values used in the calculation.

Author(s)

Carl Witthoft, `<carl@witthoft.com>`

See Also

`num2cfrac`

Examples

```
cfrac2num(rep(1,10)) # approximate phi

frac2 <- sqrt2periodicCfrac(2)
cfrac2num(frac2$numericdenom)
#simple cases
cfrac2num(denom=1)
cfrac2num(denom = c(0,2),num=1)
```

cfrac2simple *Function to Convert Continued Fraction to Simple Fraction.*

Description

Given a vector of denominators and optionally numerators for a continued fraction, return the "simple" continued fraction where all numerators are 1. In addition, return the exact "simple" fraction x/y where both are integers.

Usage

```
cfrac2simple(denom, num =1, ...)
```

Arguments

denom	A vector of values representing the standard [b0; a1,a2,a3...] continued fraction denominators, starting with the integer part.
num	A vector of values representing the numerators in a continued fraction. The default is nums = 1 as is the case for cfrac2num .
...	Reserved for future use

Value

A list containing: intfrac ,the integer fraction as a bigq value. denom, the denominator sequence for the continued fraction with all numerators equal to one.

Author(s)

Carl Witthoft, <carl@witthoft.com>

See Also

cfrac2num, num2cfrac

Examples

```
foon <- c(1,3,5,7,9)
food <- c(1,2,4,6,8,10)
foosimple <- cfrac2simple(denom =food, num=foon)
# compare with these two:
cfrac2num(den=food, num= foon)
cfrac2num(den=foosimple$denom)
```

e2cfrac *Function to Calculate e, Euler's Number With Continued Fractions*

Description

Select a particular formula to get an approximate value for Euler's number or any power `e^(x/y)` .

Usage

```
e2cfrac(nterms, pownum = 1, powdenom = 1, method = c('euler','wall','gauss'), ...)
```

Arguments

`nterms`	How many terms of the continued fraction formula to use.
`pownum`	An integer identifying the numerator of the (possibly fractional) power to raise `e` to. See Details.
`powdenom`	An integer identifying the denominator of the (possibly fractional) power to raise `e` to. See Details.
`method`	Select the method, i.e. the continued fraction formula, for the calculation. If `pownum/powdenom != 1`, the method 'gauss' is automatically chosen. See Details.
`...`	Reserved for future use

Details

The two methods which calculate `e` but not to any power, use the following formulas: "euler" denominators = 2; 1,2,1,1,4,1,1,6,1,1,8,... "wall" denominators = 2; 1,2,3,4,... and numerators = 1,1,2,3,4,... The third method, which can calculate powers of `e` is variously listed as derived by Euler or proved by Gauss based on hypergeometric functions, is automatically invoked if `pownum/powdenom != 1` . "gauss" denominators = 1; 2y-x,6y,10y,14y, 18y, 22y,.. and numerators = 2x,x^2,x^2,x^2,... where the exponent is x/y .

Due to the cyclic formula for the "euler" case, the exact number of terms used in the calculation may differ from the input `nterms` by 1 or 2 counts.

Value

edec contains the `mpfr` extended decimal representation. `egmp` contains the fraction representation. `nterms, pownum, powdenom` echo back the input arguments used. `method, nums, denoms` report the actual values used in the calculation.

Author(s)

Carl Witthoft, <`carl@witthoft.com`>

References

https://mathworld.wolfram.com/eContinuedFraction.html https://en.wikipedia.org/wiki/Euler's_continued_fraction_formula https://en.wikipedia.org/wiki/Gauss's_continued_fraction

Examples

```
e2cfrac(nterms = 10)
e2cfrac(nterms = 10, method = 'wall')
e2cfrac(nterms = 10, pownum = 2)
e2cfrac(nterms  = 10, powdenom = 2)
```

---

num2cfrac *Function to Generate The Continued Fraction For a Number*

Description

This function takes a numeric value or a character string version of a number and returns the continued fraction equivalent representation.

Usage

```
num2cfrac(num, denom = 1, ...)
```

Arguments

num	A value of class numeric, bigz, bigq, or character representing the numerator of the value to be converted. If the class is bigq then the input argument denom is ignored. Warning: if a numeric value with a fractional part is entered, errors will occur due to decimal - binary precision errors. Use either the character string form or enter integers for both numerator and denominator. A character string must be of the form [+,-][X][.][Y][[E,e][+,-]Z] where X,Y, and Z are strings of numerals. See the Details section for more information.
denom	Optional argument representing the denominator of the value to be converted. The default is denom = 1 . A character string must be of the form [+,][X][.][Y][[E,e][+,-]Z] where X,Y, and Z are strings of numerals. See the Details section for more information.
...	Reserved for future use.

Details

The use of mpfr - class numbers is deliberately disallowed, because the small but generally present binary vs. decimal rounding error will lead to horrific continued fraction representations. As noted above, character strings must be in the form [+,-][X][.][Y][[E,e][+,-]Z] where X,Y, and Z are strings of numerals. To be explicit: if an exponent is desired, the "+, -" sign is optional but the numeric value Z is required.

While irrationals can't be expressed as a finite-digit number, by entering as many digits of a known approximation as desired, the equivalent close approximation as a finite continued fraction can be generated. See the functions pi2cfrac, phi2cfrac, e2cfrac to generate as many elements of these known generators as desired.

Value

The denominators are returned as a bigz integer vector in denom. The actual numerator and denominator values used in the continued fraction creation are echoed back as bigz integers in numdenom . For example, if the input numerator and denominator were 6 and 15, the values 2 and 5 (reduced fraction) would be returned. Or, if the numerator input is the character string "1.25" , then the values 5 and 4 will be returned. The numerators (A) and denominators (B) of the convergents are provided in convA and convB

Author(s)

Carl Witthoft, <carl@witthoft.com>

See Also

go2bigq, cfrac2num

Examples

```
num2cfrac(num = 35, denom = 11)
num2cfrac(num = "-5.4967E+2")
num2cfrac(num = "1.3579")
# but this will be "wrong" :
num2cfrac(num = 1.3579)
```

pell *Function to Solve Pell's Equation With Continued Fractions*

Description

Solve Pell's equation, $x^2 - d * y^2 = 1$, using continued fractions.

Usage

```
pell(d, maxrep=10 , A = NULL, B = NULL)
```

Arguments

`d` The integer which multiplies y in the equation. For obvious reasons, there is no value to selecting a perfect square.

`maxrep` The maximum number of times to repeat the periodic section of the continued fraction while generating convergents. This exists only to avoid the possibility of an infinite "while" loop should something have gone wrong.

`A,B` Should a run of `pell` fail to find a solution (due, most likely, to a value of the argument `maxrep`), enter the convergents `A` and `B` from the previous run. This simply reduces runtime by avoiding the need to recalculate these convergent pairs.

Details

As is nicely presented in the Wikipedia entry for Pell's Equation, Let h[i] / k[i] denote the sequence ofconvergents to the regular continued fraction for sqrt(n). This sequence is unique. Then the pairsolving Pell's equation and minimizing x satisfies x1 = h[i] and y1 = k[i] for some i. This pair is called thefundamental solution. Thus, the fundamental solution may be found by performing the continued fraction expansion and testing each successive convergent until a solution to Pell's equation is found. The sequence of convergents for the square root of seven are h/k(convergent) h^2-7*k^2(Pell-type approximation) 2/1 3 3/1 +2 5/2 3 8/3 +1 so 8 / 3 is the one we want

Value

A list. `wins` contains the solutions found `d` echo back the input argument used. `cfracdata` contains the output generated with `sqrt2periodicCfrac` `A,B` the numerators and denominators of the convergents calculated.

Author(s)

Carl Witthoft, `<carl@witthoft.com>`

References

`https://mathworld.wolfram.com/eContinuedFraction.html` `https://en.wikipedia.org/wiki/Pell's_equation`

`phi2cfrac` *Function to Calculate Phi And Powers of Phi In Continued Fraction Form.*

Description

This function generates the continued fraction form of the "golden ratio", `phi^N` for integer powers N.

Usage

```
phi2cfrac( nterms = 10, exponent = 1,  ...)
```

Arguments

`nterms` How many denominators to calculate.

`exponent` An positive integer indicating the power of `phi` desired. The default is 1.

`...` Reserved for future use.

Details

The 'golden ratio' , equal to `(1 + sqrt(5))/2`, is the ratio of two sides x < y of a rectangle such that, by removing a square of side x, the remaining rectangle has the same ratio.

It turns out, in one of those mathematical curiosities, the denominators of the continued fraction form of `phi` are all equal to one. Some people use this to state, humorously, that this makes `phi` "the most irrational irrational number." It also happens that the continued fraction form for powers of `phi` consist of Lucas Numbers (see References).

Value

The continued fraction denominators are provided in `denom`. The inputs `nterms` and `exponent` are echoed back for reference.

Author(s)

Carl Witthoft, <carl@witthoft.com>

References

https://en.wikipedia.org/wiki/Lucas_number https://en.wikipedia.org/wiki/Golden_ratio

See Also

`num2cfrac`

Examples

```
phi2cfrac(nterms = 10)
phi2cfrac(exponent = 3)
foop <- phi2cfrac(nterms = 20)
cfrac2num(denom  = foop$denom)
# compare with:
library(Rmpfr)
(1 + sqrt(mpfr(5,1000)))/2
```

pi2cfrac *Function to Calculate Pi In Continued Fraction Form*

Description

This function generates the continued fraction coefficients for pi, with three different algorithms to choose from.

Usage

```
pi2cfrac(nterms, method = c('brouncker','stern','coleman'),...)
```

Arguments

nterms	The number of terms (basically depth of denominators) to generate.
method	The method to use, entered as a string. The choice can be abbreviated so long as it's unambiguous. See the Details section for discussion of the methods and their discoverers.
...	Reserved for future use.

Details

The three methods are named for their discoverers: Brouncker in 1655, Stern in 1833, and Coleman & Pickett in 2008. There are others which can be found in various papers; these may be added in a future release.

Value

A list containing: The numerators and denominators of the continued fraction in `nums` and `denoms` , as `bigz` integers, `nterms` and `method` echoed back for reference

Author(s)

Carl Witthoft, `<carl@witthoft.com>`

References

http://scihi.org/william-brouncker-approximation-pi/ https://divisbyzero.com/2008/12/09/a-new-continued-fraction-for-pi/ https://mathworld.wolfram.com/PiContinuedFraction.html

See Also

`cfrac2num`

Examples

```
pi2cfrac(nterms = 100, method='stern')
pi2cfrac(nterms = 100, method = 'coleman')
```

recipCfrac *Function to Generate the Reciprocal of A Continued Fraction*

Description

Given the numerators and denominators of a continued fraction, generate the numerators and denominators of the reciprocal of that fraction.

Usage

```
recipCfrac( denom, num = 1, ...)
```

Arguments

denom	A vector of the continued fraction denominators.
num	A vector of the continued fraction numerators. Default is 1, indicating that all numerators have the value 1.
...	Reserved for future use.

Details

This is a Q&D tool to perform a simple operation. It can easily be shown that the reciprocal of a continued fraction is obtained by prepending `0` to the current denominator sequence and prepending 1 to the current numerator sequence. To be explicit, given the standard denominator notation `[b0; a1,a2,a3...]` , the new denominator vector is `[ 0; b0, a1,a2,a3...]`

Value

A list containing `denom` and `num` (numerators) for the reciprocal value.

Author(s)

Carl Witthoft, <carl@witthoft.com>

Examples

```
foon <- c(1,3,5,7,9)
food <- c(1,2,4,6,8,10)
foor <- recipCfrac(num = foon, den=food)
# compare:
cfrac2num(num=foor$num, denom = foor$denom)$cgmp
original <- cfrac2num(num=foon, denom=food)
original$cgmp *cfrac2num(num=foor$num, denom = foor$denom)$cgmp
```

---

root2cfrac *Function To Generate Continued Fraction For Arbitrary Roots*

Description

This function generates the generalized continued fraction for any input value `x^(m/n)`.

Usage

```
root2cfrac(x, n, m = 1, nterms = 10, ...)
```

Arguments

`x`	The number itself. Integers, doubles, bigz, and bigq classes are allowed.
`n`	The integer denominator of the power to which x is raised. That is, when `m` is 1, the n-th root of x is generated.
`m`	The integer numerator of the power to which x is raised. The default is 1.
`nterms`	How many terms (denominators) to calculate.
`...`	Reserved for future use

Details

The generalized continued fraction for arbitrary roots will not be periodic, and may not even show a pattern in the denominator values. By comparison, `sqrt2periodicCfrac` generates a simple continued fraction with a periodic sequence for square roots only. That periodic sequence tends to converge more slowly than the aperiodic sequence produced here.

Value

A list, containing: The continued fraction numerators and denominators in `bigz` form `num` , `denom` . The continued fraction numerators and denominators in numeric form `numericnum`, `numericdenom` . In the extreme case that a value exceeds the machine size of a numeric, `NA` is returned. The inputs `x` , `n` , `m` are echoed back.

Author(s)

Carl Witthoft, `<carl@witthoft.com>`

References

https://en.wikipedia.org/wiki/Generalized_continued_fraction

See Also

sqrt2periodicCfrac, cfrac2num

Examples

```
root2cfrac(x = 2, n = 3)
root2cfrac(x=17, n= 5, m= 2)
root2cfrac(x = 2, n = 2, nterms = 20)
#compare with
sqrt2periodicCfrac(num = 2, nterms = 20)
```

---

sqrt2periodicCfrac *Function To Generate Periodic Continued Fraction For Square Roots*

Description

This function produces the denominators of the continued fraction form of any square root.

Usage

```
sqrt2periodicCfrac(num, denom = 1,  nterms = 50, ...)
```

Arguments

num	An integer, bigz integer, or a character string form of an integer, representing the number or the numerator of the number for which the continued fraction form of the square root is to be generated.
denom	An integer ,bigz integer, or a character string form of an integer,representing the denominator of the number for which the continued fraction form of the square root is to be generated. The default value is 1. If denom is not 1, the values returned must be divided by the denominator. See the Details section.
nterms	The maximum number of terms (denominators) to calculate. This is a "safety" limit in case the denominator repeat pattern turns out to be extremely long. See the Details section.
...	Reserved for future use.

Details

As discussed in the references, this algorithm will produce a periodic sequence of denominator values for any rational input value. Important note: the algorithm actually calculates the square root of num * denom because it only works properly with integers. Thus, if num/denom is not an integer, you must divide the results by the value of the denominator to get the correct square root. If the returned value of repeated is "FALSE" then increase the input argument nterms. The default value (50) exists so that the function can terminate rather than spend (possibly undesired) time calculating extremely long denominator repeat sequences.

Value

A list, with: The continued fraction numerators and denominators in bigz form num , denom . The continued fraction numerators and denominators in numeric form numericnum, numericdenom . In the extreme case that a value exceeds the machine size of a numeric, NA is returned. repeated returns TRUE if the denominator repeat argument is found, and FALSE if not. input echoes back the input num and denom arguments. The numerators (A) and denominators (B) of the convergents are provided in convA and convB

Author(s)

Carl Witthoft, <carl@witthoft.com>

References

https://r-knott.surrey.ac.uk/Fibonacci/cfINTRO.html section6.2 https://math.stackexchange.com/questions/2215918

Proof of periodicity: https://web.math.princeton.edu/mathlab/jr02fall/Periodicity/mariusjp.pdf

Examples

```
sqrt2periodicCfrac(12)
sqrt2periodicCfrac('12')
sqrt2periodicCfrac(12,7)
```

Index