Package ‘jacobi’

July 19, 2022

Type Package

Title Jacobi Theta Functions and Related Functions

Version 2.0.0

Description Evaluation of the Jacobi theta functions and related functions: Weierstrass elliptic function, Weierstrass sigma function, Weierstrass zeta function, Klein j-function, Dedekind eta function, lambda modular function, Jacobi elliptic functions, Neville theta functions, and Eisenstein series. Complex values of the variable are supported.

License GPL-3

URL https://github.com/stla/jacobi

BugReports https://github.com/stla/jacobi/issues

Imports Carlson, Rcpp (>= 1.0.8), rgl, Rvcg

Suggests testthat (>= 3.0.0), elliptic

LinkingTo Rcpp

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.2.0

NeedsCompilation yes

Author Stéphane Laurent [aut, cre],
Mikael Fremling [aut] (author of the original Fortran code for the theta functions)

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Repository CRAN

Date/Publication 2022-07-19 12:00:09 UTC

R topics documented:

`agm` *Arithmetic-geometric mean*

Description

Evaluation of the arithmetic-geometric mean of two complex numbers.

Usage

```
agm(x, y)
```

Arguments

`x, y` complex numbers

Value

A complex number, the arithmetic-geometric mean of `x` and `y`.

Examples

```
agm(1, sqrt(2))
2*pi^(3/2)*sqrt(2) / gamma(1/4)^2
```

`am` *Amplitude function*

Description

Evaluation of the amplitude function.

Usage

```
am(u, m)
```

Arguments

`u`	complex number
`m`	square of elliptic modulus, a complex number

Value

A complex number.

Examples

```
library(Carlson)
phi <- 1 + 1i
m <- 2
u <- elliptic_F(phi, m)
am(u, m) # should be phi
```

`CostaMesh` *Costa surface*

Description

Computes a mesh of the Costa surface.

Usage

```
CostaMesh(nu = 50L, nv = 50L)
```

Arguments

`nu, nv`	numbers of subdivisions

Value

A triangle **rgl** mesh (object of class `mesh3d`).

Examples

```
library(jacobi)
library(rgl)

mesh <- CostaMesh(nu = 250, nv = 250)
open3d(windowRect = c(50, 50, 562, 562), zoom = 0.9)
bg3d("#15191E")
shade3d(mesh, color = "darkred", back = "cull")
shade3d(mesh, color = "orange", front = "cull")
```

EisensteinE *Eisenstein series*

Description

Evaluation of Eisenstein series with weight 2, 4 or 6.

Usage

```
EisensteinE(n, q)
```

Arguments

`n`	the weight, can be 2, 4 or 6
`q`	nome, complex number with modulus smaller than one, but not a negative real number

Value

A complex number, the value of the Eisenstein series.

---

eta *Dedekind eta function*

Description

Evaluation of the Dedekind eta function.

Usage

```
eta(tau)
```

Arguments

`tau`	a complex number with strictly positive imaginary part

Value

A complex number.

Examples

```
eta(2i)
gamma(1/4) / 2^(11/8) / pi^(3/4)
```

`jellip` *Jacobi elliptic functions*

Description

Evaluation of the Jacobi elliptic functions.

Usage

```
jellip(kind, u, tau = NULL, m = NULL)
```

Arguments

`kind`	a string with two characters among `"s"`, `"c"`, `"d"` and `"n"`; this string specifies the function: the two letters respectively denote the basic functions sn, cn, dn and 1, and the string specifies the ratio of two such functions, e.g. $ns = 1/sn$ and $cd = cn/dn$
`u`	a complex number, vector or matrix
`tau`	complex number with strictly positive imaginary part; it is related to `m` and only one of them must be supplied
`m`	the "parameter", square of the elliptic modulus; it is related to `tau` and only one of them must be supplied

Value

A complex number, vector or matrix.

Examples

```
u <- 2 + 2i
tau <- 1i
jellip("cn", u, tau)^2 + jellip("sn", u, tau)^2 # should be 1
```

`jtheta1` *Jacobi theta function one*

Description

Evaluates the first Jacobi theta function.

Usage

```
jtheta1(z, tau = NULL, q = NULL)

ljtheta1(z, tau = NULL, q = NULL)
```

Arguments

`z`	complex number, vector, or matrix
`tau`	lattice parameter, a complex number with strictly positive imaginary part; the two complex numbers `tau` and `q` are related by `q = exp(1i*pi*tau)`, and only one of them must be supplied
`q`	the nome, a complex number whose modulus is strictly less than one, and which is not zero nor a negative real number

Value

A complex number, vector or matrix; `jtheta1` evaluates the first Jacobi theta function and `ljtheta1` evaluates its logarithm.

Examples

```
jtheta1(1 + 1i, q = exp(-pi/2))
```

`jtheta2` *Jacobi theta function two*

Description

Evaluates the second Jacobi theta function.

Usage

```
jtheta2(z, tau = NULL, q = NULL)

ljtheta2(z, tau = NULL, q = NULL)
```

Arguments

z	complex number, vector, or matrix
tau	lattice parameter, a complex number with strictly positive imaginary part; the two complex numbers tau and q are related by q = exp(1i*pi*tau), and only one of them must be supplied
q	the nome, a complex number whose modulus is strictly less than one, and which is not zero nor a negative real number

Value

A complex number, vector or matrix; jtheta2 evaluates the second Jacobi theta function and ljtheta2 evaluates its logarithm.

Examples

```
jtheta2(1 + 1i, q = exp(-pi/2))
```

---

jtheta3 *Jacobi theta function three*

Description

Evaluates the third Jacobi theta function.

Usage

```
jtheta3(z, tau = NULL, q = NULL)

ljtheta3(z, tau = NULL, q = NULL)
```

Arguments

z	complex number, vector, or matrix
tau	lattice parameter, a complex number with strictly positive imaginary part; the two complex numbers tau and q are related by q = exp(1i*pi*tau), and only one of them must be supplied
q	the nome, a complex number whose modulus is strictly less than one, and which is not zero nor a negative real number

Value

A complex number, vector or matrix; jtheta3 evaluates the third Jacobi theta function and ljtheta3 evaluates its logarithm.

Examples

```
jtheta3(1 + 1i, q = exp(-pi/2))
```

jtheta4 *Jacobi theta function four*

Description

Evaluates the fourth Jacobi theta function.

Usage

```
jtheta4(z, tau = NULL, q = NULL)

ljtheta4(z, tau = NULL, q = NULL)
```

Arguments

z	complex number, vector, or matrix
tau	lattice parameter, a complex number with strictly positive imaginary part; the two complex numbers `tau` and `q` are related by `q = exp(1i*pi*tau)`, and only one of them must be supplied
q	the nome, a complex number whose modulus is strictly less than one, and which is not zero nor a negative real number

Value

A complex number, vector or matrix; `jtheta4` evaluates the fourth Jacobi theta function and `ljtheta4` evaluates its logarithm.

Examples

```
jtheta4(1 + 1i, q = exp(-pi/2))
```

kleinj *Klein j-function and its inverse*

Description

Evaluation of the Klein j-invariant function and its inverse.

Usage

```
kleinj(tau, transfo = FALSE)

kleinjinv(j)
```

Arguments

`tau`	a complex number with strictly positive imaginary part, or a vector or matrix of such complex numbers; missing values allowed
`transfo`	Boolean, whether to use a transformation of the values of `tau` close to the real line; using this option can fix some failures of the computation (at the cost of speed), e.g. when the algorithm reaches the maximal number of iterations
`j`	a complex number

Value

A complex number, vector or matrix.

Note

The Klein-j function is the one with the factor 1728.

Examples

```
( j <- kleinj(2i) )
66^3
kleinjinv(j)
```

`lambda` *Lambda modular function*

Description

Evaluation of the lambda modular function.

Usage

```
lambda(tau, transfo = FALSE)
```

Arguments

`tau`	a complex number with strictly positive imaginary part, or a vector or matrix of such complex numbers; missing values allowed
`transfo`	Boolean, whether to use a transformation of the values of `tau` close to the real line; using this option can fix some failures of the computation (at the cost of speed), e.g. when the algorithm reaches the maximal number of iterations

Value

A complex number, vector or matrix.

Note

The lambda function is the square of the elliptic modulus.

Examples

```
x <- 2
lambda(1i*sqrt(x)) + lambda(1i*sqrt(1/x)) # should be one
```

`theta.s` *Neville theta functions*

Description

Evaluation of the Neville theta functions.

Usage

```
theta.s(z, tau = NULL, m = NULL)

theta.c(z, tau = NULL, m = NULL)

theta.n(z, tau = NULL, m = NULL)

theta.d(z, tau = NULL, m = NULL)
```

Arguments

`z`	a complex number, vector, or matrix
`tau`	complex number with strictly positive imaginary part; it is related to `m` and only one of them must be supplied
`m`	the "parameter", square of the elliptic modulus; it is related to `tau` and only one of them must be supplied

Value

A complex number, vector or matrix.

`wp` *Weierstrass elliptic function*

Description

Evaluation of the Weierstrass elliptic function and its derivatives.

Usage

```
wp(z, g = NULL, omega = NULL, tau = NULL, derivative = 0L)
```

Arguments

`z`	complex number, vector or matrix
`g`	the elliptic invariants, a vector of two complex numbers; only one of `g`, `omega` and `tau` must be given
`omega`	the half-periods, a vector of two complex numbers; only one of `g`, `omega` and `tau` must be given
`tau`	the half-periods ratio; supplying `tau` is equivalent to supply `omega = c(1/2, tau/2)`
`derivative`	differentiation order, an integer between 0 and 3

Value

A complex number, vector or matrix.

Examples

```
omega1 <- 1.4 - 1i
omega2 <- 1.6 + 0.5i
omega <- c(omega1, omega2)
e1 <- wp(omega1, omega = omega)
e2 <- wp(omega2, omega = omega)
e3 <- wp(-omega1-omega2, omega = omega)
e1 + e2 + e3 # should be 0
```

`wpinv` *Inverse of Weierstrass elliptic function*

Description

Evaluation of the inverse of the Weierstrass elliptic function.

Usage

```
wpinv(w, g = NULL, omega = NULL, tau = NULL)
```

Arguments

`w`	complex number
`g`	the elliptic invariants, a vector of two complex numbers; only one of `g`, `omega` and `tau` must be given
`omega`	the half-periods, a vector of two complex numbers; only one of `g`, `omega` and `tau` must be given
`tau`	the half-periods ratio; supplying `tau` is equivalent to supply `omega = c(1/2, tau/2)`

Value

A complex number.

Examples

```
library(jacobi)
omega <- c(1.4 - 1i, 1.6 + 0.5i)
w <- 1 + 1i
z <- wpinv(w, omega = omega)
wp(z, omega = omega) # should be w
```

---

wsigma *Weierstrass sigma function*

Description

Evaluation of the Weierstrass sigma function.

Usage

```
wsigma(z, g = NULL, omega = NULL, tau = NULL)
```

Arguments

z	a complex number, vector or matrix
g	the elliptic invariants, a vector of two complex numbers; only one of g, omega and tau must be given
omega	the half-periods, a vector of two complex numbers; only one of g, omega and tau must be given
tau	the half-periods ratio; supplying tau is equivalent to supply omega = c(1/2, tau/2)

Value

A complex number, vector or matrix.

Examples

```
wsigma(1, g = c(12, -8))
# should be equal to:
sin(1i*sqrt(3))/(1i*sqrt(3)) / sqrt(exp(1))
```

`wzeta` *Weierstrass zeta function*

Description

Evaluation of the Weierstrass zeta function.

Usage

```
wzeta(z, g = NULL, omega = NULL, tau = NULL)
```

Arguments

`z`	complex number, vector or matrix
`g`	the elliptic invariants, a vector of two complex numbers; only one of `g`, `omega` and `tau` must be given
`omega`	the half-periods, a vector of two complex numbers; only one of `g`, `omega` and `tau` must be given
`tau`	the half-periods ratio; supplying `tau` is equivalent to supply `omega = c(1/2, tau/2)`

Value

A complex number, vector or matrix.

Examples

```
# Mirror symmetry property:
z <- 1 + 1i
g <- c(1i, 1+2i)
wzeta(Conj(z), Conj(g))
Conj(wzeta(z, g))
```

Index