Package ‘Bessel’

February 19, 2015

Version 0.5-5
Date 2013-12-10
Title Bessel -- Bessel Functions Computations and Approximations
Author Martin Maechler
Maintainer Martin Maechler <maechler@stat.math.ethz.ch>
Depends Rmpfr
Suggests gsl
Description Bessel Function Computations for complex and real numbers; notably interfacing TOMS 644; approximations for large arguments, experiments, etc.
License GPL (>= 2)
URL http://specfun.r-forge.r-project.org/
NeedsCompilation yes
Repository CRAN
Date/Publication 2013-12-10 10:36:57

R topics documented:

Airy ................................................................. 2
Bessel ............................................................ 3
BesselH ............................................................ 4
bessell.nuAsym .............................................. 6
bessellasym ...................................................... 7
bi .............................................................. 9

Index 10
Description

Compute the Airy functions $Ai$ or $Bi$ or their first derivatives, $\frac{d}{dz}Ai(z)$ and $\frac{d}{dz}Bi(z)$.

Usage

AiryA(z, deriv = 0, expon.scaled = FALSE)
AiryB(z, deriv = 0, expon.scaled = FALSE)

Arguments

z complex or numeric vector.
deriv order of derivative; must be 0 or 1.
expon.scaled logical indicating if the result should be scaled by an exponential factor (typically to avoid under- or over-flow).

Details

By default, when expon.scaled is false, AiryA() computes the complex Airy function $Ai(z)$ or its derivative $\frac{d}{dz}Ai(z)$ on deriv=0 or deriv=1 respectively.
When expon.scaled is true, it returns $exp(\zeta)Ai(z)$ or $exp(\zeta)\frac{d}{dz}Ai(z)$, effectively removing the exponential decay in $-\pi/3 < \arg(z) < \pi/3$ and the exponential growth in $\pi/3 < |\arg(z)| < \pi$, where $\zeta = \frac{2}{3}z\sqrt{z}$.

While the Airy functions $Ai(z)$ and $d/dzAi(z)$ are analytic in the whole $z$ plane, the corresponding scaled functions (for expon.scaled=TRUE) have a cut along the negative real axis.

By default, when expon.scaled is false, AiryB() computes the complex Airy function $Bi(z)$ or its derivative $\frac{d}{dz}Bi(z)$ on deriv=0 or deriv=1 respectively.
When expon.scaled is true, it returns $exp(-|\Re(\zeta)|)Bi(z)$ or $exp(-|\Re(\zeta)|)\frac{d}{dz}Bi(z)$, to remove the exponential behavior in both the left and right half planes where, as above, $\zeta = \frac{2}{3}z\sqrt{z}$.

Value

a complex or numeric vector of the same length (and class) as z.

Author(s)

Donald E. Amos, Sandia National Laboratories, wrote the original fortran code. Martin Maechler did the R interface.

References

see BesselI.
Bessel Functions of Complex Arguments $I()$, $J()$, $K()$, and $Y()$

**Description**

Compute the Bessel functions $I()$, $J()$, $K()$, and $Y()$, of complex arguments $z$ and real $\nu$.

**Usage**

- `BesselI(z, nu, expon.scaled = FALSE, nSeq = 1)`
- `BesselJ(z, nu, expon.scaled = FALSE, nSeq = 1)`
- `BesselK(z, nu, expon.scaled = FALSE, nSeq = 1)`
- `BesselY(z, nu, expon.scaled = FALSE, nSeq = 1)`

**Arguments**

- $z$: complex or numeric vector.
- $nu$: numeric (scalar).
- `expon.scaled`: logical indicating if the result should be scaled by an exponential factor (typically to avoid under- or over-flow).
- `nSeq`: positive integer; if > 1, computes the result for a whole sequence of $nu$ values; if $nu \geq 0$, $nu$, $nu+1$, ..., $nu+nSeq-1$; if $nu < 0$, $nu$, $nu-1$, ..., $nu-nSeq+1$.

**Details**

The case $nu < 0$ is handled by using simple formula from Abramowitz and Stegun.

---

See Also

- `BesselI` etc; the Hankel functions `Hankel`.

**Examples**

```r
### The AiryA() := Ai() function
curve(AiryA, -20, 100, n=1001)
curve(AiryA, -1, 100, n=1001, log="y")
curve(AiryA(x, expon.scaled=TRUE), -1, 50, n=1001)
curve(AiryA(x, expon.scaled=TRUE), 1, 10000, n=1001, log="xy")

### The AiryB() := Bi() function
curve(AiryB, -20, 2, n=1001); abline(h=0,v=0, col="gray", lty=2)
curve(AiryB, -1, 20, n=1001, log="y") # exponential growth (x > 0)
curve(AiryB(x, expon.scaled=TRUE), -1, 20, n=1001)
curve(AiryB(x, expon.scaled=TRUE), 1, 10000, n=1001, log="x")
```
BesselH

Value

A complex or numeric vector (or matrix with nSeq columns if nSeq > 1) of the same length (or nrow when nSeq > 1) and mode as z.

Author(s)

Donald E. Amos, Sandia National Laboratories, wrote the original fortran code. Martin Maechler did the R interface.

References


See Also

The base R functions besseli, etc.

Examples

```r
## For real small arguments, BesselI() gives the same as base::besseli() :
set.seed(47); x <- sort(round(rlnorm(20), 2))
M <- cbind(x, b = besseli(x, 3), B = BesselI(x, 3))
stopifnot(all.equal(M[,"b"], M[,"B"]))
M
```

BesselH

Hankel (H-Bessel) Function (of Complex Argument)

Description

Compute the Hankel functions $H(1, \ast)$ and $H(2, \ast)$, also called ‘H-Bessel’ function (of the third kind), of complex arguments.

Usage

```r
BesselH(m, z, nu, expon.scaled = FALSE, nSeq = 1)
```
Arguments

- **m**: integer, either 1 or 2, indicating the kind of Hankel function.
- **z**: complex or numeric vector of values **different from 0**.
- **nu**: numeric, must currently be non-negative.
- **expon.scaled**: logical indicating if the result should be scaled by an exponential factor (typically to avoid under- or over-flow).
- **nSeq**: positive integer, ...

Details

By default (when `expon.scaled` is false), the resulting sequence (of length `nSeq`) is

\[ y_j = H(m, \nu + j - 1, z), \]

computed for \( j = 1, \ldots, nSeq \).

If `expon.scaled` is true, the sequence is

\[ y_j = \exp(-\tilde{m}zi) \cdot H(m, \nu + j - 1, z), \]

where \( \tilde{m} = 3 - 2m \) (and \( i^2 = -1 \)), for \( j = 1, \ldots, nSeq \).

Value

A complex or numeric vector (or matrix if \( nSeq > 1 \)) of the same length and mode as \( z \).

Author(s)

Donald E. Amos, Sandia National Laboratories, wrote the original fortran code. Martin Maechler did the R interface.

References

see `besseli`.

See Also

`BesselI` etc; the Airy function `Airy`.

Examples

```r
#----------------------- H(1, x) -----------------------
nus <- c(1, 2, 5, 10)
for(i in seq_along(nus))
  curve(BesselH(1, x, nu=nus[i]), -10, 10, add= i > 1, col=i, n=1000)
legend("topleft", paste("nu = ", format(nus)), col = seq_along(nus), lty=1)

# nu = 10 looks a bit "special": hmm...
curve(BesselH(1, x, nu=10), -.3, .3, col=4,
    ylim = c(-10,10), n=1000)
```
besselI.nuAsym

Asymptotic Expansion of Bessel I(x,nu) and K(x,nu) for Large nu (and x)

Description

Compute Bessel functions I_\nu(x) and K_\nu(x) for large \nu and possibly large x, using asymptotic expansions in Debye polynomials.

Usage

\[
\text{besselI.nuAsym}(x, \text{nu}, k.\text{max}, \text{expon.scaled} = \text{FALSE}, \text{log} = \text{FALSE})
\]

\[
\text{besselK.nuAsym}(x, \text{nu}, k.\text{max}, \text{expon.scaled} = \text{FALSE}, \text{log} = \text{FALSE})
\]

Arguments

x numeric, \geq 0.

nu numeric; The order (maybe fractional!) of the corresponding Bessel function.

k.max integer number of terms in the expansion. Must be in 0:4, currently.

expon.scaled logical; if TRUE, the results are exponentially scaled in order to avoid overflow (I_\nu) or underflow (K_\nu), respectively.

log logical; if TRUE, log(f(.)) is returned instead of f.

Details

Abramowitz & Stegun, page 378, has formula 9.7.7 and 9.7.8 for the asymptotic expansions of I_\nu(x) and K_\nu(x), respectively.

The Debye polynomials u_k(x) are defined in 9.3.9 and 9.3.10 (page 366).

Value

a numeric vector of the same length as the long of x and nu. (usual argument recycling is applied implicitly.)

Author(s)

Martin Maechler
besseliasym

References


See Also

From this package Bessel BesselI(); further, besseliasym() for the case when \( x \) is large and \( \nu \) is small or moderate; further base besseli, etc

Examples

```r
x <- c(1:10, 20, 50, 100, 10000)
nu <- c(1, 10, 20, 50, 10^2:10))

sapply(0:4, function(k.)
sapply(nu, function(n.)
  besseli.nuAsym(x, nu=n., k.max = k., log = TRUE))
sapply(0:4, function(k.)
sapply(nu, function(n.)
  besselK.nuAsym(x, nu=n., k.max = k., log = TRUE))
```

besseliasym

Asymptotic Expansion of besselI(x,\( \nu \)) For Large \( x \)

Description

Compute Bessel function \( I_\nu(x) \) and \( K_\nu(x) \) for large \( x \) and small or moderate \( \nu \), using the asymptotic expansion (9.7.1), p.377 of Abramowitz & Stegun, for \( x \rightarrow \infty \), even valid for complex \( x \),

\[
I_\nu(x) = \frac{\exp(x)}{\sqrt{2\pi x}} \cdot f(x, a),
\]

where

\[
f(x, a) = 1 - \frac{\mu - 1}{8x} + \frac{(\mu - 1)(\mu - 9)}{2!(8x)^2} - \ldots,
\]

and \( \mu = 4a^2 \) and \( |arg(x)| < \pi/2 \).

Whereas besseliasym(x, a) computes \( I_\nu(x) \), besseli.ftrms returns the corresponding terms in the series expansion of \( f(x, a) \) above.

Usage

```r
besseliasym (x, nu, k.max = 10, expon.scaled = FALSE, log = FALSE)
besseli.ftrms(x, nu, K = 20)
```
Arguments

- **x** numeric, \( \geq 0 \).
- **nu** numeric; The order (maybe fractional!) of the corresponding Bessel function.
- **k.max, K** integer number of terms in the expansion.
- **expon.scaled** logical; if TRUE, the results are exponentially scaled in order to avoid overflow.
- **log** logical; if TRUE, \( \log(f(.)) \) is returned instead of \( f \).

Details

\[ \ldots \quad \text{FIXME} \quad \ldots \]

Value

a numeric vector of the same length as \( x \).

Author(s)

Martin Maechler

References


See Also

From this package **Bessel()** **Besseli()**; further, **besseli.nuAsym()** which is useful when \( \nu \) is large (as well); further **base** **besseli**, etc.

Examples

```r
x <- c(1:10, 20, 50, 100*2:10))
nu <- c(1, 10, 20, 50, 100)

r <- lapply(c(0:4,10,20), function(k.)
   sapply(nu, function(n.)
      besseli asym(x, nu=n., k.max = k., log = TRUE)))

warnings()
```
**bI**

**Bessel I() function Simple Series Representation**

**Description**

Computes the modified Bessel $I$ function, using one of its basic definitions as an infinite series. The implementation is pure R, working for numeric, complex, but also e.g., for objects of class "mpfr" from package Rmpfr.

**Usage**

```
bi(x, nu, nterm = 800, exponent.scaled = FALSE, log = FALSE, 
    Ceps = if (isNum) 8e-16 else 2^(-x@.Data[[1]]@prec))
```

**Arguments**

- **x**: numeric of complex vector, or of another class for which arithmetic methods are defined, notably objects of class mpfr.
- **nu**: non-negative numeric (scalar).
- **nterm**: integer indicating the number of terms to be used. should be in the order of abs(x), but can be smaller for large x. A warning is given, when nterm was chosen too small.
- **exponent.scaled**: logical indicating if the result should be scaled by $\exp(-\text{abs}(x))$.
- **log**: logical indicating if the logarithm logI() is required. is not yet implemented!
- **Ceps**: a relative error tolerance for checking if nterm has been sufficient. The default is “correct” for double precision and also for multiprecision objects.

**Value**

a “numeric” (or complex or ...) vector of the same class and length as x.

**Author(s)**

Martin Maechler

**References**


**See Also**

This package *BesselI*, base *besseli*, etc

**Examples**

```
stopifnot(all.equal(bi(1:10, 1), # R code
                     besseli(1:10, 1)))# internal C code w/ different algorithm
```
Index

*Topic math
   Airy, 2
   Bessel, 3
   BesselH, 4
   besselI.nuAsym, 6
   besselIasym, 7
   bi, 9
   Airy, 2, 5
   AiryA (Airy), 2
   AiryB (Airy), 2
   Bessel, 3
   BesselH, 4
   BesselI, 2, 3, 5, 7–9
   BesselI (Bessel), 3
   besselI, 4, 7–9
   besselI.ftrms (besselIasym), 7
   besselI.nuAsym, 6, 8
   besselIasym, 7, 7
   BesselJ (Bessel), 3
   BesselK (Bessel), 3
   besselK.nuAsym (besselI.nuAsym), 6
   BesselY (Bessel), 3
   bi, 9
   class, 9
   complex, 7, 9
   Hankel, 3
   Hankel (BesselH), 4
   matrix, 4, 5
   mode, 4, 5
   mpfr, 9
   nrow, 4
   numeric, 9