Package ‘kelvin’

September 19, 2015

Type Package
Title Calculate Solutions to the Kelvin Differential Equation using
    Bessel Functions
Version 2.0-0
Date 2015-09-18
Author Andrew J Barbour
Maintainer Andrew J Barbour <andy.barbour@gmail.com>
Description Uses Bessel functions to calculate the
    fundamental and complementary analytic solutions to the
    Kelvin differential equation.
Depends R (>= 2.10.1)
Imports Bessel (>= 0.5-4)
Suggests knitr, testthat
License GPL (>= 2)
URL https://github.com/abarbour/kelvin
BugReports https://github.com/abarbour/kelvin/issues
LazyLoad TRUE
VignetteBuilder knitr
NeedsCompilation no
Repository CRAN
Date/Publication 2015-09-19 10:00:21

R topics documented:

kelvin-package .................................................. 2
Beir ................................................................. 3
Keir ............................................................... 4

Index 7
Description

The functions here use Bessel functions to calculate the analytic solutions to the Kelvin differential equation, namely the fundamental (Be) and equivalent (Ke) complex functions.

Details

The complex second-order ordinary differential equation, known as the Kelvin differential equation, is defined as

$$x^2 \ddot{y} + x \dot{y} - (ix^2 + \nu^2) y = 0$$

and has a suite of complex solutions. One set of solutions, $B_\nu$, is defined in the following manner:

$$B_\nu \equiv \text{Ber}_\nu(x) + i\text{Bei}_\nu(x)$$

\[= J_\nu(x \cdot \exp(3\pi i/4))\]

\[= \exp(\nu\pi i) \cdot J_\nu(x \cdot \exp(-\pi i/4))\]

\[= \exp(\nu\pi i/2) \cdot I_\nu(x \cdot \exp(\pi i/4))\]

\[= \exp(3\nu\pi i/2) \cdot I_\nu(x \cdot \exp(-3\pi i/4))\]

where $J_\nu$ is a Bessel function of the first kind, and $I_\nu$ is a modified Bessel function of the first kind.

Similarly, the complementary solutions, $K_\nu$, are defined as

$$K_\nu \equiv \text{Ker}_\nu(x) + i\text{Kei}_\nu(x)$$

\[= \exp(-\nu\pi i/2) \cdot K_\nu(x \cdot \exp(\pi i/4))\]

where $K_\nu$ is a modified Bessel function of the second kind.

The relationships between $y$ in the differential equation, and the solutions $B_\nu$ and $K_\nu$ are as follows

$$y = \text{Ber}_\nu(x) + i\text{Bei}_\nu(x)$$

$$= \text{Ber}_{-\nu}(x) + i\text{Bei}_{-\nu}(x)$$

$$= \text{Ker}_\nu(x) + i\text{Kei}_\nu(x)$$

$$= \text{Ker}_{-\nu}(x) + i\text{Kei}_{-\nu}(x)$$

In the case where $\nu = 0$, the differential equation reduces to

$$x^2 \ddot{y} + x \dot{y} - ix^2 y = 0$$

which has the set of solutions:

$$J_0 \left(i\sqrt{i} \cdot x\right)$$

\[= J_0 \left(\sqrt{2} \cdot (i - 1) \cdot x/2\right)\]

\[= \text{Ber}_0(x) + i\text{Bei}_0(x) \equiv B_0\]

This package has functions to calculate $B_\nu$ and $K_\nu$. 

---

**kelvin-package**

*Fundamental and equivalent solutions to the Kelvin differential equation using Bessel functions*
**Beir**

**Author(s)**
Andrew Barbour <andy.barbour@gmail.com>

**References**

Kelvin functions: [http://mathworld.wolfram.com/KelvinFunctions.html](http://mathworld.wolfram.com/KelvinFunctions.html)

Bessel functions: [http://mathworld.wolfram.com/BesselFunction.html](http://mathworld.wolfram.com/BesselFunction.html)

**See Also**
Fundamental solution: Beir
Equivalent solution: Keir

---

**Beir**

*Fundamental solution to the Kelvin differential equation (J)*

**Description**
This function calculates the complex solution to the Kelvin differential equation using modified Bessel functions of the *first kind*, specifically those produced by `besselJ`.

**Usage**

```r
Beir(x, ...) 
## default S3 method: 
Beir(x, nu = 0, nSeq. = 1, return.list = FALSE, ...)

Bei(...) 
Ber(...)```

**Arguments**

- `x` numeric; values to evaluate the complex solution at
- `...` additional arguments passed to `BesselK` or `Beir`
- `nu` numeric; value of $\nu$ in $B_\nu$ solutions
- `nSeq` positive integer; equivalent to nSeq in `BesselJ`
- `return.list` logical; Should the result be a list instead of matrix?
Details

Ber and Bei are wrapper functions which return the real and imaginary components of Beir, respectively.

Value

If return.list==FALSE (the default), a complex matrix with as many columns as using nSeq. creates. Otherwise the result is a list with matrices for Real and Imaginary components.

Author(s)

Andrew Barbour

References

http://mathworld.wolfram.com/KelvinFunctions.html

Imaginary: http://mathworld.wolfram.com/Bei.html

Real: http://mathworld.wolfram.com/Ber.html

See Also

keir-package, Keir, BesselJ

Examples

Beir(1:10)  # defaults to nu.=0
Beir(1:10, nu.=2)
Beir(1:10, nSeq.=2)
Beir(1:10, nSeq.=2, return.list=TRUE)
  # Imaginary component only
Bei(1:10)
  # Real component only
Ber(1:10)

Keir

Complementary solution to the Kelvin differential equation (\(K\))

Description

This function calculates the complex solution to the Kelvin differential equation using modified Bessel functions of the second kind, specifically those produced by BesselK.
Usage

Keir(x, ...)

## Default S3 method:
Keir(x, nu. = 0L, nSeq. = 1L, add.tol = TRUE,
     return.list = FALSE, show.scaling = FALSE, ...)

Kei(...)

Ker(...)

Arguments

- **x**: numeric; values to evaluate the complex solution at
- **...**: additional arguments passed to `BesselK` or `Keir`
- **nu.**: numeric; value of $\nu$ in $K_\nu$ solutions
- **nSeq.**: positive integer; equivalent to nSeq in `BesselK`
- **add.tol**: logical; Should a fudge factor be added to prevent an error for zero-values?
- **return.list**: logical; Should the result be a list instead of matrix?
- **show.scaling**: logical; Should the normalization values be given as a message?

Details

`Ker` and `Kei` are wrapper functions which return the real and imaginary components of `Keir`, respectively.

Value

If `return.list==FALSE` (the default), a complex matrix with as many columns as using `nSeq` creates. Otherwise the result is a list with matrices for Real and Imaginary components.

Author(s)

Andrew Barbour

References

- [http://mathworld.wolfram.com/KelvinFunctions.html](http://mathworld.wolfram.com/KelvinFunctions.html)
- Real: [http://mathworld.wolfram.com/Ker.html](http://mathworld.wolfram.com/Ker.html)

See Also

`kelvin-package`, `Beir`, `BesselK`
Examples

Keir(1:10)  # defaults to nu.=0, nSeq=1
Keir(1:10, nu.=2)
Keir(1:10, nSeq=2)
Keir(1:10, nSeq=2, return.list=TRUE)
# Imaginary component only
Kei(1:10)
# Real component only
Ker(1:10)
Index

Bei. 4
Bei (Beir), 3
Beir. 3, 3, 4, 5
Ber. 4
Ber (Beir), 3
BesselJ. 3, 4
BesselK. 3–5

Kei. 5
Kei (Keir). 4
Keir. 3, 4, 4, 5
kelvin (kelvin-package). 2
kelvin-package. 2
Ker. 5
Ker (Keir). 4