Package ‘Brobdingnag’

August 13, 2018

Type Package
Title Very Large Numbers in R
Version 1.2-6
Date 2018-08-08
Author Robin K. S. Hankin
Depends R (>= 2.13.0), methods
Maintainer Robin K. S. Hankin <hankin.robin@gmail.com>
Description Handles very large numbers in R. Real numbers are held using their natural logarithms, plus a logical flag indicating sign. The package includes a vignette that gives a step-by-step introduction to using S4 methods.
LazyLoad yes
License GPL
Repository CRAN
URL https://github.com/RobinHankin/Brobdingnag.git
NeedsCompilation no
Date/Publication 2018-08-13 13:20:03 UTC

R topics documented:

  Brobdingnag-package .................................................. 2
  Arith-methods ........................................................... 4
  as.numeric ................................................................. 5
  brob ................................................................. 6
  brob-class ............................................................... 7
  cbrob ................................................................. 8
  Compare-methods ......................................................... 9
  Complex ................................................................. 9
  Extract.brob ............................................................ 10
  getP ................................................................. 11
  glub ................................................................. 12
Brobdingnag-package

Description

Handles very large numbers in R. Real numbers are held using their natural logarithms, plus a logical flag indicating sign. The package includes a vignette that gives a step-by-step introduction to using S4 methods.

Details

The DESCRIPTION file:

Package: Brobdingnag
Type: Package
Title: Very Large Numbers in R
Version: 1.2-6
Date: 2018-08-08
Author: Robin K. S. Hankin
Depends: R (>= 2.13.0), methods
Maintainer: Robin K. S. Hankin <hankin.robin@gmail.com>
Description: Handles very large numbers in R. Real numbers are held using their natural logarithms, plus a logical flag indicating sign.
LazyLoad: yes
License: GPL
Repository: CRAN
URL: https://github.com/RobinHankin/Brobdingnag.git

Index of help topics:

Arith-methods Methods for Function Arith in package Brobdingnag
Brobdingnag-package Very Large Numbers in R
Compare-methods Methods for Function Compare in Package Brobdingnag
Re Real and imaginary manipulation
[.brob Extract or Replace Parts of brobs or glubs
Real numbers are represented by two objects: a real, holding the logarithm of their absolute values; and a logical, indicating the sign. Multiplication and exponentiation are easy: the challenge is addition. This is achieved using the (trivial) identity \( \log(e^x + e^y) = x + \log(1 + e^{y-x}) \) where, \( \text{WLOG, } y < x \).

Complex numbers are stored as a pair of brobs: objects of class glub.

The package is a simple example of S4 methods.

However, it could be viewed as a cautionary tale: the underlying R concepts are easy yet the S4 implementation is long and difficult. I would not recommend using S4 methods for a package as simple as this; S3 methods would have been perfectly adequate. I would suggest that S4 methods should only be used when S3 methods are demonstrably inadequate.

Author(s)

Robin K. S. Hankin

Maintainer: Robin K. S. Hankin <hankin.robin@gmail.com>

References


Examples

googol <- as.brob(10)^100

googol

googol + googol/2

1/(googol + 1e99)

(1:10)^googol
```r
googolplex <- 10^googol

googolplex

googolplex * googol # practically the same as googolplex (!)
```

**Description**

Methods for Arithmetic functions in package Brobdingnag: `+`, `-`, `*`, `/`, `^`

**Note**

The unary arithmetic functions (viz `+` and `-`) do no coercion.

The binary arithmetic functions coerce numeric `<op>` `brob` to `brob`; and numeric `<op>` `glub`, complex `<op>` `brob`, and `brob `<op>` `glub`, to `glub`.

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
x <- as.brob(1:10)
y <- 1e10

x+y

as.numeric((x+y)-1e10)

x^(1/y)
```
Description

Coerces an object of class brob to numeric, or an object of class glub to complex.

Arguments

- **x**: Object of class brob or glub
- **...**: Further arguments (currently ignored)

Details

Function `as.numeric()` coerces a brob to numeric; if given a glub, the imaginary component is ignored (and a warning given).

Function `as.complex()` coerces to complex.

Note

If $|x|$ is greater than `.Machine$double.xmax`, then `as.numeric(x)` returns `Inf` or `-Inf` but no warning is given.

Author(s)

Robin K. S. Hankin

Examples

```r
a <- as.brob(1:10)
a <- cbrob(a, as.brob(10)^1e26)
a
as.numeric(a)
as.complex(10i + a)
```
**brob**

**Brobdingnagian numbers**

**Description**

Create, coerce to or test for a Brobdingnagian object

**Usage**

```
brob(x = double(), positive = FALSE)
```

as.brob(x)

**Arguments**

- **x**: Quantity to be tested, coerced in to Brobdingnagian form
- **positive**: In function `brob()`, logical indicating whether the number is positive (actually, positive or zero)

**Details**

Function `as.brob()` is the user's workhorse: use this to coerce numeric vectors to `brobs`.

Function `is.brob()` tests for its arguments being of class `brob`.

Function `brob()` takes argument `x` and returns a `brob` formally equal to $e^x$; set argument `positive` to `FALSE` to return $-e^x$. Thus calling function `exp(x)` simply returns `brob(x)`. This function is not really intended for the end user: it is confusing and includes no argument checking. In general numerical work, use function `as.brob()` instead, although be aware that if you really really want $e^{10^7}$, you should use `brob(1e7)`: this would be an **exact** representation.

**Note**

Real numbers are represented by two objects: a real, holding the logarithm of their absolute values; and a logical, indicating the sign. Multiplication and exponentiation are easy: the challenge is addition. This is achieved using the (trivial) identity $\log(e^x + e^y) = x + \log(1 + e^{y-x})$ where, WLOG, $y < x$.

Complex numbers are stored as a pair of `brobs`: objects of class `glub`.

The package is a simple example of S4 methods. However, it could be viewed as a cautionary tale: the underlying R concepts are easy yet the S4 implementation is long and difficult. I would not recommend using S4 methods for a package as simple as this; S3 methods would have been perfectly adequate. I would suggest that S4 methods should only be used when S3 methods are demonstrably inadequate.

The package has poor handling of `NA` and `NaN`. Currently, `as.brob(1) + as.brob(c(1, NA))` returns an error.
Author(s)

Robin K. S, Hankin

See Also

glob

Examples

```r
googol <- as.brob(10)^100
googolplex <- 10^googol

(googolplex/googol) / googolplex  
# Thus googolplex/googol == googolplex (!)

# use cbrob() instead of c() when Brobdingnagian numbers are involved:
cbrob(4,exp(as.brob(1e55)))
```

brob-class

Class "brob"

Description

The formal S4 class for Brobdingnagian numbers

Objects from the Class

Objects can be created by calls of the form new("brob", ...) but this is not encouraged. Use functions brob() and, especially, as.brob() instead.

Slots

- x: Object of class "numeric" holding the log of the absolute value of the number to be represented
- positive: Object of class "logical" indicating whether the number is positive (see Note, below)

Extends

Class "swift", directly.

Note

Slot positive indicates non-negativity, as zero is conventionally considered to be "positive".
Author(s)
Robin K. S. Hankin

See Also

*glub-class, swift-class*

Examples

```r
new("brob", x=5, positive=TRUE)  # not intended for the user
as.brob(5)  # Standard user-oriented idiom
```

---

**cbrob**  
*Combine Brobdingnagian vectors*

Description

Combine Brobdingnagian or Glubdubdribian vectors through concatenation

Usage

```r
cbrob(x, ...)
```

Arguments

- **x**: Brobdingnagian vector
- **...**: Other arguments coerced to brob form

Details

If any argument has class *glub*, all arguments are coerced to *glubs*. Otherwise, if any argument has class *brob*, all arguments are coerced to *brobs*.

Function `cbrob()` operates recursively, calling `.cPair()` repeatedly. Function `.cPair()` uses S4 method dispatch to call either `.Brob.cpair()` or `.Glub.cpair()` according to the classes of the arguments.

Note

As of R-2.4.0, it is apparently not possible to use S4 methods to redefine `c()` to coerce to class *brob* form and concatenate as expected. This would seem to be a reasonable interpretation of `c()` from the user's perspective.

Conceptually, the operation is simple: concatenate the value slot and the positive slot separately, then call `brob()` on the two resulting vectors. When concatenating *glub* objects, the real and imaginary components (being *brobs*) are concatenated using `.Brob.cpair()`.

The choice of name—`cbrob()`—is not entirely logical. Because it operates consistently on *brob* and *glub* objects, it might be argued that `cSwift()` would be a more appropriate name.
Author(s)

Robin K. S. Hankin; original idea due to John Chambers

Examples

```
a <- as.brob(2)^1e-40
cbrob(1:4,4:1,a)
cbrob(1:4,a,1i)
```

Description

Methods for comparision (greater than, etc) in package Brobdingnag

Note

As for `min()` and `max()`, comparison is not entirely straightforward in the presence of NAs.

The low-level workhorses are `.Brob.equal()` for equality and `.Brob.greater()` for ‘strictly greater than’. All other comparisons are calculated by combining these two.

Comparison [function `.Brob.compare()`] explicitly tests for a zero length argument and if given one returns `logical(0)` to match base behaviour.

Examples

```
a <- as.brob(10)^(0.5 + 97:103)
a < 1e100
```

Description

Get or set real and imaginary components of brobs or glubs.
Usage

```r
## S4 method for signature 'glub'
Re(z)
## S4 method for signature 'glub'
Im(z)
## S4 method for signature 'glub'
Mod(z)
## S4 method for signature 'glub'
Conj(z)
## S4 method for signature 'glub'
Arg(z)
Re(z) <- value
Im(z) <- value
```

Arguments

- `z` : object of class `glub` (or, in the case of `Im(-)` or `Im(z) <- value`, class `brob`
- `value` : object of class `numeric` or `brob`

Value

Functions `Re()` and `Im()` return an object of class `brob`; functions `Re<-()` and `Im<-()` return an object of class `glub`

Author(s)

Robin K. S. Hankin

Examples

```r
a <- cbrob(1:10,brob(1e100))
Im(a) <- 11:1
a
```

---

Extract.brob

Description

Methods for "[" and "]<-", i.e., extraction or subsetting of brobs and glubs.

Arguments

- `x` : Object of class `brob` or `glub`
- `i` : elements to extract or replace
- `value` : replacement value
**Value**

Always returns an object of the same class as \( x \).

**Note**

If \( x \) is a numeric vector and \( y \) a brob, one might expect typing \( x[1] \leftarrow y \) to result in \( x \) being a brob. This is impossible, according to John Chambers.

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
a <- as.brob(10)^c(-100, 0, 100, 1000, 1e32)
a[4]
a[4] <- 1e100
da
```

---

**getP**  
*Get and set methods for brob objects*

**Description**

Get and set methods for brobs: sign and value

**Usage**

```r
getP(x)  
getX(x)  
sign(x) <- value
```

**Arguments**

- \( x \)  
  Brobdingnagian object
- \( \text{value} \)  
  In function `sign<-()`, Boolean specifying wheter the brob object is positive

**Author(s)**

Robin K. S. Hankin

**See Also**

`brob`
Examples
x <- as.brob(-10:10)
sign(x) <- TRUE

Description
Create, coerce to or test for a Glubbdubdribian object

Usage
glut(real = double(), imag = double())
as.glut(x)
is.glut(x)

Arguments
real, imag Real and imaginary components of complex number: must be Brobdingnagian numbers
x object to be coerced to or tested for Glubbdubdribian form

Details
Function glut() takes two arguments that are coerced to Brobdingnagian numbers and returns a complex number. This function is not really intended for the end user: it is confusing and includes no argument checking. Use function as.glut() instead.

Function as.glut() is the user’s workhorse: use this to coerce numeric or complex vectors to Glubbdubdribian form.

Function is.glut() tests for its arguments being Glubbdubdribian.

Note
Function glut() uses recycling inherited from cbind().

Author(s)
Robin K. S. Hankin

See Also
brob
Examples

```r
a <- as.glub(1:10 + 5i)
a^2 - a*a

f <- function(x){sin(x) + x^4 - 1/x}
as.complex(f(a)) - f(as.complex(a))  # should be zero (in the first
                        # term, f() works with glubs and coerces to
                        # complex; in the second, f()
                        # works with complex numbers directly)
```

---

**Description**

Complex Brobdingnagian numbers

**Objects from the Class**

A `glub` object holds two slots, both brobs, representing the real and imaginary components of a complex vector.

**Slots**

- `real`: Object of class "brob" representing the real component
- `imag`: Object of class "brob" representing the imaginary component

**Extends**

Class "swift", directly.

**Methods**

- `.cPair` signature(x = "brob", y = "glub"): ...
- `.cPair` signature(x = "ANY", y = "glub"): ...
- `.cPair` signature(x = "glub", y = "glub"): ...
- `.cPair` signature(x = "glub", y = "ANY"): ...
- `.cPair` signature(x = "glub", y = "brob"): ...
- `Im<-` signature(x = "glub"): ...
- `Re<-` signature(x = "glub"): ...

**Author(s)**

Robin K. S. Hankin
See Also

`brob-class`, `swift-class`

Examples

```r
a <- as.brob(45)
new("glub", real=a, imag=a)

as.brob(5+5i)  # standard colloquial R idiom
```

Description

Get lengths of brob and glub vectors

Usage

```r
## S4 method for signature 'brob'
length(x)
## S4 method for signature 'glub'
length(x)
```

Arguments

- `x` vector of class brob or glub

Author(s)

Robin K. S. Hankin

Examples

```r
x <- as.brob(-10:10)
length(x)
```
**Logic**

**Logical operations on brobs**

**Description**

Logical operations on brobs are not supported

**Note**

The S4 group generic “Logic” appeared in R-2.4.0-patched.

Carrying out logical operations in this group will call `Brob.logic()`, which reports an error.

Negation, “!”, is not part of this group: attempting to negate a brob will not activate `Brob.logic()`; an “invalid argument type” error is given instead.

**Author(s)**

Robin K. S. Hankin

**Examples**

```r
## Not run:
!Brob(10)

## End(Not run)
```

---

**Math**

**Various logarithmic and circular functions for brobs**

**Description**

Various elementary functions for brobs

**Arguments**

- `x` Object of class brob (or sometimes glub)
- `base` In function `log()`, the base of the logarithm

**Details**

For brobs: apart from `abs()`, `log()`, `exp()`, `sinh()` and `cosh()`, these functions return `as.numeric(x)` so are numeric; the exceptional functions return brobs.

For glubs: mostly direct transliteration of the appropriate formula; one might note that `log(z)` is defined as `glub(log(Mod(x)), Arg(x))`. 
Author(s)
Robin K. S. Hankin

Examples
exp(as.brob(3000))  #exp(3000) is represented with zero error

plot                  Basic plotting of Brobs

Description
Plotting methods. Essentially, any brob is coerced to a numeric and any glub is coerced to a complex, and the argument or arguments are passed to plot().

Usage
plot(x, y, ...)

Arguments
x, y              Brob or glub
...               Further arguments passed to plot()

Author(s)
Robin K. S. Hankin

Examples
plot(as.brob(1:10))

Print                  Methods for printing brobs and glubs

Description
Methods for printing brobs and glubs nicely using exponential notation

Usage
## S3 method for class 'brob'
print(x, ...)
## S3 method for class 'glub'
print(x, ...)
Arguments

x   An object of class brob or glub
... Further arguments (currently ignored)

Author(s)

Robin K. S. Hankin

Examples

a <- as.brob(1:5)
dput(a)
a

sum  Various summary statistics for brobs and glubs

Description

Various summary statistics for brobs and glubs

Arguments

x, ... Objects of class brob or, in the case of sum() and prob(), class glub
na.rm Boolean, with default FALSE meaning to interpret NAs literally and TRUE meaning to ignore any such elements

Details

For a brob object, being NA is not entirely straightforward. The S4 method for is.na is too “strict” for some of the functions considered here. Consider max(a) where a includes only positive, fully specified, elements, and elements with known negative sign and exponents that include NA values. Here, max(a) is unambiguously determined.
Similar logic applies to min() and, by extension, range().

Note

Function prod() is very slow for long glub vectors. It has to compute four Brobdingnagian products and two Brobdingnagian sums per element of its argument, and this takes a long time.

Author(s)

Robin K. S. Hankin

See Also

is.na
Examples

a <- as.brob(1:10)
max(cbrob(1:10, brob(NA, FALSE)))

Description

A (virtual) class that extends brob and glub objects

Objects from the Class

A virtual Class: No objects may be created from it.

Methods

No methods defined with class "swift" in the signature.

Author(s)

Robin K. S. Hankin

See Also

brob-class, glub-class
Index

*Topic **classes**
  brob-class, 7
  glub-class, 13
  swift-class, 18

*Topic **math**
  Arith-methods, 4
  abs (Math), 15
  acos (Math), 15
  acosh (Math), 15
  as.numeric, 5
  arg (Complex), 9
  arg, brob-method (Complex), 9
  arg, glub-method (Complex), 9
  Arith, ANY, brob-method (Arith-methods), 4
  Arith, ANY, glub-method (Arith-methods), 4
  Arith, brob, ANY-method (Arith-methods), 4
  Arith, brob, complex-method (Arith-methods), 4
  Arith, brob, missing-method (Arith-methods), 4
  Arith, complex, brob-method (Arith-methods), 4
  Arith, complex, glub-method (Arith-methods), 4
  Arith, glub, ANY-method (Arith-methods), 4
  Arith, glub, brob-method (Arith-methods), 4
  Arith, glub, complex-method (Arith-methods), 4
  Arith, glub, missing-method (Arith-methods), 4
  Arith-methods, 4

*Topic **methods**
  Arith-methods, 4
  length-methods, 14

*Topic **package**
  Brobdingnag-package, 2
  .cPair, ANY, ANY-method (brob-class), 7
  .cPair, ANY, brob-method (brob-class), 7
  .cPair, ANY, glub-method (glub-class), 13
  .cPair, brob, ANY-method (brob-class), 7
  .cPair, brob, brob-method (brob-class), 7
  .cPair, brob, complex-method (brob-class), 7
  .cPair, brob, glub-method (glub-class), 13
  .cPair, complex, brob-method (brob-class), 7
  .cPair, glub, ANY-method (glub-class), 13
  .cPair, glub, brob-method (glub-class), 13
  .cPair, glub, glub-method (glub-class), 13
  [, brob-method (Extract.brob), 10
  [, , glub-method (Extract.brob), 10
  [, .brob (Extract.brob), 10
  [, .glub (Extract.brob), 10
  [<- , brob-method (Extract.brob), 10
  [<- , glub-method (Extract.brob), 10
  [<- .brob (Extract.brob), 10
  [<- .glub (Extract.brob), 10
  abs (Math), 15
  acos (Math), 15
  acosh (Math), 15
  Arg (Complex), 9
  Arg, brob-method (Complex), 9
  Arg, glub-method (Complex), 9
  Arith, ANY, brob-method (Arith-methods), 4
  Arith, ANY, glub-method (Arith-methods), 4
  Arith, brob, ANY-method (Arith-methods), 4
  Arith, brob, complex-method (Arith-methods), 4
  Arith, brob, missing-method (Arith-methods), 4
  Arith, complex, brob-method (Arith-methods), 4
  Arith, complex, glub-method (Arith-methods), 4
  Arith, glub, ANY-method (Arith-methods), 4
  Arith, glub, brob-method (Arith-methods), 4
  Arith, glub, complex-method (Arith-methods), 4
  Arith, glub, missing-method (Arith-methods), 4
  Arith-methods, 4
as.brob (brob), 6
as.complex (as.numeric), 5
as.complex, brob-method (as.numeric), 5
as.complex, glub-method (as.numeric), 5
as.glub (glub), 12
as.numeric, 5
as.numeric, brob-method (as.numeric), 5
as.numeric, glub-method (as.numeric), 5
asin (Math), 15
asinh (Math), 15
atan (Math), 15
atanh (Math), 15
brob, 6, 11, 12
brob-class, 7
Brobdingnag (Brobdingnag-package), 2
Brobdingnag-package, 2
cBrob (cbrob), 8
cbrob, 8
ceiling (Math), 15
c coerce, brob, complex-method
(as.numeric), 5
c coerce, brob, numeric-method
(as.numeric), 5
c coerce, glub, numeric-method
(as.numeric), 5
c coerce, glub, complex-method
(as.numeric), 5
c coerce, glub, numeric-method
(as.numeric), 5
Compare, ANY, brob-method
(Compare-methods), 9
Compare, ANY, glub-method
(Compare-methods), 9
Compare, brob, ANY-method
(Compare-methods), 9
Compare, brob, brob-method
(Compare-methods), 9
Compare, brob, glub-method
(Compare-methods), 9
Compare, glub, ANY-method
(Compare-methods), 9
Compare, glub, brob-method
(Compare-methods), 9
Compare, glub, glub-method
(Compare-methods), 9
Compare-methods, 9
Complex, 9
Complex, brob-method (Complex), 9
Complex, glub-method (Complex), 9
Complex-methods (Complex), 9
Conj (Complex), 9
Conj, brob-method (Complex), 9
Conj, glub-method (Complex), 9
cos (Math), 15
cosh (Math), 15
cumsum (Math), 15
exp (Math), 15
Extract.brob, 10
floor (Math), 15
gamma (Math), 15
getP, 11
getP, brob-method (brob-class), 7
getX (getP), 11
getX, brob-method (brob-class), 7
glub, 7, 12
glub-class, 13
Im (Complex), 9
Im, brob-method (Complex), 9
Im, glub-method (Complex), 9
Im<-, (Complex), 9
Im<-, brob-method (Complex), 9
Im<-, glub-method (Complex), 9
is.brob (brob), 6
is.glub (glub), 12
is.na, 17
length (length-methods), 14
length, brob-method (length-methods), 14
length, glub-method (length-methods), 14
length-methods, 14
lgamma (Math), 15
log (Math), 15
Logic, 15
Logic, ANY, swift-method (Logic), 15
Logic, swift, ANY-method (Logic), 15
Logic, swift, swift-method (Logic), 15
logic.brob (Logic), 15
Math, 15
Math, brob-method (Math), 15
Math, glub-method (Math), 15
max (sum), 17
min (sum), 17
Mod (Complex), 9
Mod, brob-method (Complex), 9
INDEX

Mod, glub-method (Complex), 9

plot, 16
plot, ANY, brob-method (plot), 16
plot, ANY, glub-method (plot), 16
plot, brob, ANY-method (plot), 16
plot, brob, missing-method (plot), 16
plot, brob-method (plot), 16
plot, glub, ANY-method (plot), 16
plot, glub, missing-method (plot), 16
plot, glub-method (plot), 16
Print, 16
print.brob (Print), 16
print.glub (Print), 16
prod (sum), 17

range (sum), 17
Re (Complex), 9
Re, brob-method (Complex), 9
Re, glub-method (Complex), 9
Re<- (Complex), 9
Re<-, glub-method (Complex), 9

show, brob-method (Print), 16
show, glub-method (Print), 16
sign<~ (getP), 11
sign<-, brob-method (brob-class), 7
sin (Math), 15
sinh (Math), 15
sqrt (Math), 15
sqrt, brob-method (Math), 15
sqrt, glub-method (Math), 15
sum, 17
Summary, brob-method (sum), 17
Summary, glub-method (sum), 17
swift-class, 18

tan (Math), 15
tanh (Math), 15
trunc (Math), 15