Package 'lazyNumbers'

November 21, 2022

Type Package

Title Exact Floating-Point Arithmetic

Version 1.2.1

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Description Provides a new type of numbers called the lazy numbers. Arithmetic on lazy numbers is exact, contrary to ordinary floating-point arithmetic. The lazy numbers are implemented in 'C++' with the 'CGAL' library (<https://www.cgal.org/>).

License GPL-3

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

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

Imports methods, Rcpp (>= 1.0.9), stats

LinkingTo BH, Rcpp, RcppCGAL, RcppEigen

Encoding UTF-8

RoxygenNote 7.2.2

SystemRequirements C++ 17, gmp, mpfr

Collate 'classes.R' 'Arith.R' 'Compare.R' 'Concat.R' 'Extract.R' 'Math.R' 'RcppExports.R' 'Summary.R' 'Transpose.R' 'aaa.R' 'lazyMatrices.R' 'lazyNumbers.R'

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation yes

Author Stéphane Laurent [aut, cre]

Repository CRAN

Date/Publication 2022-11-21 21:50:06 UTC

R topics documented:

`bind2-lazyMatrices` *Concatenation of lazy matrices*

Description

Concatenate two `lazyMatrix` objects.

Usage

```
## S4 method for signature 'lazyMatrix,missing'
cbind2(x, y)

## S4 method for signature 'lazyVector,missing'
cbind2(x, y)

## S4 method for signature 'lazyMatrix,lazyMatrix'
cbind2(x, y)

## S4 method for signature 'lazyVector,lazyMatrix'
cbind2(x, y)

## S4 method for signature 'lazyMatrix,lazyVector'
cbind2(x, y)

## S4 method for signature 'lazyVector,lazyVector'
```

```
cbind2(x, y)

## S4 method for signature 'lazyVector,numeric'
cbind2(x, y)

## S4 method for signature 'numeric,lazyVector'
cbind2(x, y)

## S4 method for signature 'lazyVector,matrix'
cbind2(x, y)

## S4 method for signature 'matrix,lazyVector'
cbind2(x, y)

## S4 method for signature 'matrix,lazyMatrix'
cbind2(x, y)

## S4 method for signature 'lazyMatrix,numeric'
cbind2(x, y)

## S4 method for signature 'numeric,lazyMatrix'
cbind2(x, y)

## S4 method for signature 'lazyMatrix,missing'
rbind2(x, y)

## S4 method for signature 'lazyVector,missing'
rbind2(x, y)

## S4 method for signature 'lazyMatrix,lazyMatrix'
rbind2(x, y)

## S4 method for signature 'lazyVector,lazyMatrix'
rbind2(x, y)

## S4 method for signature 'lazyMatrix,lazyVector'
rbind2(x, y)

## S4 method for signature 'lazyVector,lazyVector'
rbind2(x, y)

## S4 method for signature 'lazyVector,numeric'
rbind2(x, y)

## S4 method for signature 'numeric,lazyVector'
rbind2(x, y)

## S4 method for signature 'lazyVector,matrix'
```

```
rbind2(x, y)

## S4 method for signature 'matrix,lazyVector'
rbind2(x, y)

## S4 method for signature 'lazyMatrix,matrix'
rbind2(x, y)

## S4 method for signature 'matrix,lazyMatrix'
rbind2(x, y)

## S4 method for signature 'lazyMatrix,numeric'
rbind2(x, y)

## S4 method for signature 'numeric,lazyMatrix'
rbind2(x, y)
```

Arguments

x, y	lazyMatrix or lazyVector objects

Value

A lazyMatrix object.

---

concat-lazyObjects *Concatenation of lazy vectors*

Description

Concatenate two or more lazyVector or lazyMatrix objects.

Usage

```
## S4 method for signature 'lazyVector'
c(x, ...)

## S4 method for signature 'lazyMatrix'
c(x, ...)
```

Arguments

x	a lazyVector object or a lazyMatrix object
...	lazyVector objects or lazyMatrix objects or numeric vectors or numeric matrices

Value

A lazyVector object.

diag *Extract/replace diagonal of a lazy matrix*

Description

Extract or replace the diagonal of a square lazy matrix.

Usage

```
## S4 method for signature 'lazyMatrix'
diag(x)

## S4 replacement method for signature 'lazyMatrix,lazyVector'
diag(x) <- value
```

Arguments

x	a square lazy matrix
value	a lazy vector

Value

The diagonal of x as a lazy vector, or the modified matrix.

intervals *Intervals for lazy numbers*

Description

For each lazy number in a lazyVector object or a lazyMatrix object, this function computes an interval containing this lazy number.

Usage

```
intervals(x)
```

Arguments

x	a lazyVector object or a lazyMatrix object

Value

A named list ("inf" and "sup") containing: two numeric vectors if x is a lazy vector, two numeric matrices if x is a lazy matrix.

Examples

```
library(lazyNumbers)
x <- lazynb(22) / lazynb(7)
itrv <- intervals(x)
print(itrv, digits = 17L)
x_dbl <- as.double(x)
itrv$inf <= x_dbl & x_dbl <= itrv$sup
```

`is.na` *Missing lazy values*

Description

Check whether values are missing in lazy vectors and lazy matrices.

Usage

```
## S4 method for signature 'lazyVector'
is.na(x)

## S4 method for signature 'lazyMatrix'
is.na(x)

## S4 method for signature 'lazyVector'
anyNA(x, recursive = FALSE)

## S4 method for signature 'lazyMatrix'
anyNA(x, recursive = FALSE)
```

Arguments

`x`	a lazy vector or a lazy matrix
`recursive`	ignored

Value

The `is.na` function returns a logical vector or a logical matrix, and the `anyNA` function returns a logical value.

Note

The `is.na` function does not detect lazy NaN numbers; see the note in `isNaN_or_Inf`.

Examples

```
is.na(NA_lazy_)
is.na(lazyvec(c(1, 2, NA, NaN, Inf)))
anyNA(lazyvec(c(1, 2, NA)))
```

isNaN_or_Inf *Lazy infinite or NaN numbers*

Description

Check whether values are infinite or NaN in lazy vectors and lazy matrices.

Usage

```
isNaN_or_Inf(x)
```

Arguments

x	a lazy vector or a lazy matrix

Value

A logical vector or a logical matrix.

Note

If you want to check whether a lazy number is infinite or whether a lazy number is NaN, you have to call ‘as.double‘. There is no way to distinguish an infinite lazy number from a NaN lazy number without resorting to its double approximation.

Examples

```
isNaN_or_Inf(lazyvec(c(1, NaN, NA, Inf)))
```

lazyDet *Determinant of lazy matrix*

Description

Compute the determinant of a lazy matrix.

Usage

```
lazyDet(M)
```

Arguments

M	a `lazyMatrix` object corresponding to a square matrix

Value

A lazy number (`lazyVector` object with length 1).

Examples

```
M <- lazymat(toeplitz(c(3, 2, 1)))
as.double(lazyDet(M))
```

`lazyInv` *Inverse of lazy matrix*

Description

Compute the inverse of a lazy matrix.

Usage

```
lazyInv(M)
```

Arguments

`M`	a `lazyMatrix` object corresponding to a square matrix

Value

A `lazyMatrix` object.

Note

This function does not check the invertibility. If the matrix is not invertible, you will get some `NaN` in the result (after calling `as.double`).

Examples

```
library(lazyNumbers)
set.seed(666L)
M <- lazymat(matrix(rpois(9L, lambda = 4), nrow = 3L, ncol = 3L))
invM <- lazyInv(M)
I3 <- M %*% invM
as.double(I3) == diag(3)
```

lazyMatrix *Lazy matrices*

Description

Create a lazy matrix.

Usage

```
as.lazyMatrix(x)

lazymat(x, dim = NULL)
```

Arguments

x	a numeric matrix, a numeric vector, a lazyVector object, or a lazyMatrix object
dim	ignored if x is a (possibly lazy) matrix; otherwise, i.e. if x is a (possibly lazy) vector, then dim must be NULL or a vector of two integers, and NULL is equivalent to c(length(x), 1) (a column matrix)

Value

An object of class lazyMatrix.

Examples

```
library(lazyNumbers)
M <- lazymat(toeplitz(c(1, 2)))
as.double(M + M)
as.double(M * M)
as.double(M %*% M)
```

lazyMatrix-product *Matricial product of lazy matrices*

Description

Matricial product of lazy matrices.

Usage

```
## S4 method for signature 'lazyMatrix,lazyMatrix'
x %*% y
```

Arguments

x, y	objects of class lazyMatrix

Value

A lazyMatrix object.

---

lazyMatrix-unary *Unary operators for lazy matrices*

Description

Unary operators for lazy matrices.

Usage

```
## S4 method for signature 'lazyMatrix,missing'
e1 + e2

## S4 method for signature 'lazyMatrix,missing'
e1 - e2
```

Arguments

e1	object of class lazyMatrix
e2	nothing

Value

A lazyMatrix object.

---

lazyResolve *Resolve lazy numbers*

Description

Resolve the lazy numbers in a lazy vector or a lazy matrix; see details.

Usage

```
lazyResolve(x)
```

Arguments

x	a lazyVector object or a lazyMatrix object

Details

When an operation between lazy numbers is performed, the resulting lazy number is not the result of the operation, it is the unevaluated operation (wherefrom the word "lazy"). This function performs the evaluation of the operations contained in the lazy numbers of the vector/matrix; the returned lazy vector/matrix has the same values as the input lazy vector/matrix. Applying this function can help to avoid a stack overflow.

Value

Invisibly returns the lazy vector or matrix `x`, resolved.

Note

Once you call `as.double` on a lazy number, then this number is resolved (see the example).

Examples

```
library(lazyNumbers)
n <- 500
p <- seq(1, n, by = 1)
q <- seq(3, 2*n + 1, by = 2)
# fast, because the operations are not evaluated:
x1 <- 2 * (1 + sum(cumprod(lazynb(p) / lazynb(q))))
x2 <- 2 * (1 + sum(cumprod(lazynb(p) / lazynb(q))))
x3 <- 2 * (1 + sum(cumprod(lazynb(p) / lazynb(q))))
# slow, because this evaluates the operations:
lazyResolve(x1)
# fast, because `x1` is resolved now:
as.double(x1)
# slow, because `x2` must be resolved:
as.double(x2)
# fast, because the call to `as.double` has resolved `x2`
as.double(x2)
# slow, because `x3` is not resolved:
x1 == x3
# fast, because `x3` has been resolved by the equality test:
as.double(x3)
```

`lazyVector` *Lazy vector*

Description

Create a vector of lazy numbers.

Usage

```
as.lazyVector(x)

as.lazyNumber(x)

lazyvec(x)

lazynb(x)
```

Arguments

x	a numeric vector or a lazy matrix (lazyMatrix object)

Value

An object of class lazyVector.

Examples

```
library(lazyNumbers)
x <- lazynb(1) - lazynb(7) * lazynb(0.1)
as.double(x)
# shorter:
x <- 1 - lazynb(7) * 0.1
```

---

lazyVector-unary *Unary operators for lazy vectors*

Description

Unary operators for lazy vectors.

Usage

```
## S4 method for signature 'lazyVector,missing'
e1 + e2

## S4 method for signature 'lazyVector,missing'
e1 - e2
```

Arguments

e1	object of class lazyVector
e2	nothing

Value

A lazyVector object.

NA_lazy_ *The missing lazy value.*

Description

The missing lazy value.

Usage

```
NA_lazy_
```

Format

An object of class `lazyVector` of length 1.

Submatrix *Extract lazy submatrix*

Description

Extract a submatrix of a lazy matrix.

Usage

```
## S4 method for signature 'lazyMatrix,numeric,numeric,logical'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'lazyMatrix,numeric,numeric,missing'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'lazyMatrix,numeric,missing,missing'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'lazyMatrix,numeric,missing,logical'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'lazyMatrix,missing,numeric,logical'
x[i, j, drop = TRUE]

## S4 method for signature 'lazyMatrix,missing,numeric,missing'
x[i, j, drop]
```

Arguments

x	a lazyMatrix object
i, j	indices
...	ignored
drop	Boolean, whether to drop the matrix structure if i or j has only one element

Value

A lazyMatrix object or a lazyVector object.

Subvector *Extract/replace in a lazy vector*

Description

Extract or replace elements in a lazy vector.

Usage

```
## S4 method for signature 'lazyVector,numeric,ANY,ANY'
x[i]

## S4 method for signature 'lazyVector,logical,ANY,ANY'
x[i]

## S4 replacement method for signature 'lazyVector,numeric,missing,lazyVector'
x[i, j] <- value
```

Arguments

x	a lazyVector object
i	indices
j	nothing
value	a lazyVector object

Value

A lazyVector object.

transpose-lazyMatrix *Transposition of lazy matrices*

Description

Transpose a lazyMatrix object.

Usage

```
## S4 method for signature 'lazyVector'
t(x)

## S4 method for signature 'lazyMatrix'
t(x)
```

Arguments

x	a lazyMatrix or lazyVector object

Value

A lazyMatrix object.

Index