Package: lazymatrix (via r-universe)

October 12, 2026

Title Perform Complex Matrix Operations Symbolically on Sparse Matrices

Version 0.1.0.9000

Description Provides a framework for lazy computation on large sparse matrices. Enables lazy evaluation of normalized data matrices, preserving sparsity throughout operations without materializing dense intermediate objects. Implements statistical algorithms including LSQR for sparse least squares as described in Paige and Saunders (1982) <doi:10.1145/355984.355989> and partial singular value decomposition via the augmented implicitly restarted Lanczos bidiagonalization algorithm of Baglama and Reichel (2005) <doi:10.1137/04060593X>.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Suggests bench, dplyr, ggplot2, knitr, rmarkdown, scales, testthat (>= 3.0.0), tidyr

Config/testthat/edition 3

Imports Matrix, methods, stats, irlba, Rcpp

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo

URL https://vsegersall.github.io/lazymatrix/

Repository https://vsegersall.r-universe.dev

Date/Publication 2026-08-13 07:24:20 UTC

RemoteUrl https://github.com/vsegersall/lazymatrix

RemoteRef HEAD

RemoteSha a9feca82cbaadb54eb6b69c044dbff7f28188323

Contents

-,ANY,LazyColumn-method . . . 3
-,LazyColumn,ANY-method . . . 3
-,LazyColumn,LazyColumn-method . . . 4
*,LazyColumn,LazyColumn-method . . . 5
*,LazyColumn,numeric-method . . . 5
*,numeric,LazyColumn-method . . . 6
%*%,ANY,LazyMatrix-method . . . 7
%*%,LazyColumn,LazyColumn-method . . . 7
%*%,LazyColumn,numeric-method . . . 8
%*%,LazyMatrix,ANY-method . . . 9
%*%,LazyMatrix,matrix-method . . . 9
%*%,numeric,LazyColumn-method . . . 10
+,ANY,LazyColumn-method . . . 11
+,LazyColumn,ANY-method . . . 11
+,LazyColumn,LazyColumn-method . . . 12
colnames,LazyMatrix-method . . . 13
Compare,LazyColumn,numeric-method . . . 13
Compare,numeric,LazyColumn-method . . . 14
crossprod,LazyMatrix,ANY-method . . . 14
dim,LazyMatrix-method . . . 15
lazy_crossprod_vec . . . 16
lazy_crossprod_vec_sp . . . 16
LazyColumn-class . . . 17
LazyMatrix . . . 17
LazyMatrix-class . . . 18
length,LazyColumn-method . . . 18
lsqr . . . 19
ncol,LazyMatrix-method . . . 20
norm . . . 20
norm,LazyColumn-method . . . 21
norm,LazyMatrix-method . . . 21
nrow,LazyMatrix-method . . . 22
prcomp,LazyMatrix-method . . . 23
setNames,LazyColumn,character-method . . . 24
subset-LazyColumn . . . 25
subset-LazyMatrix . . . 26
svd,LazyMatrix-method . . . 27
t,LazyMatrix-method . . . 28

Index **29**

-,ANY,LazyColumn-method

Vector Subtraction between regular vector and LazyColumn

Description

Subtracts a LazyColumn vector from a base vector.

Usage

```
## S4 method for signature 'ANY,LazyColumn'
e1 - e2
```

Arguments

e1	A LazyColumn object.
e2	A numeric vector.

Value

A numeric vector.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
b <- rnorm(nrow(mat_a))
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- lazy_a[, 2]
b - lazy_col
```

-,LazyColumn,ANY-method

Vector subtraction between a LazyColumn *vector and a regular R vector.*

Description

Subtracts a numeric R vector from a LazyColumn vector.

Usage

```
## S4 method for signature 'LazyColumn,ANY'
e1 - e2
```

Arguments

e1	A LazyColumn object.
e2	A numeric vector.

Value

A numeric vector.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
b <- rnorm(nrow(mat_a))
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- lazy_a[,2]
lazy_col - b
```

-,LazyColumn,LazyColumn-method

Vector Subtraction between two LazyColumn *vectors.*

Description

Subtracts a LazyColumn vector from another LazyColumn vector.

Usage

```
## S4 method for signature 'LazyColumn,LazyColumn'
e1 - e2
```

Arguments

e1	A LazyColumn object.
e2	A LazyColumn object.

Value

A numeric vector.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow = 3, ncol = 4)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- lazy_a[, 2]
lazy_col_2 <- lazy_a[, 3]
lazy_col_2 - lazy_col
```

*,LazyColumn,LazyColumn-method

Multiply element-wise two LazyColumn vectors

Description

Computes the element-wise multiplication of two LazyColumn objects , preserving the lazy structure.

Usage

```
## S4 method for signature 'LazyColumn,LazyColumn'
e1 * e2
```

Arguments

e1	A LazyColumn object.
e2	A LazyColumn object.

Value

A numeric vector with the resulting vector.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
X <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- X[, 1]
lazy_col_2 <- X[, 2]
lazy_col_2 * lazy_col
```

*,LazyColumn,numeric-method

Multiply a LazyColumn by a numeric scalar or vector

Description

Computes the element-wise multiplication of a LazyColumn object with a numeric value, preserving the lazy structure. If e2 is a scalar, scalar multiplication is performed. If e2 is a vector of the same length as the column, element-wise multiplication is performed.

Usage

```
## S4 method for signature 'LazyColumn,numeric'
e1 * e2
```

Arguments

e1	A LazyColumn object.
e2	A numeric scalar or vector.

Value

A numeric vector with the resulting vector.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
X <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- X[, 1]
lazy_col * 2
lazy_col * rnorm(nrow(X))
```

---

*,numeric,LazyColumn-method

Multiply a numeric scalar or vector by a LazyColumn

Description

Computes the element-wise multiplication of a LazyColumn object with a numeric value, preserving the lazy structure. If e1 is a scalar, scalar multiplication is performed. If e1 is a vector of the same length as the column, element-wise multiplication is performed.

Usage

```
## S4 method for signature 'numeric,LazyColumn'
e1 * e2
```

Arguments

e1	A numeric scalar or vector.
e2	A LazyColumn object.

Value

A numeric vector with the resulting vector.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
X <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- X[, 1]
2* lazy_col
rnorm(nrow(X)) * lazy_col
```

%*%,ANY,LazyMatrix-method

Matrix multiplication for vector and LazyMatrix

Description

Multiplies a LazyMatrix object by a vector.

Usage

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

Arguments

x	A numeric vector.
y	A LazyMatrix object.

Value

A Matrix object of class dgeMatrix.

Examples

```
mat_a <- base::matrix(rep(1, 6), nrow=2, ncol=3)
b <- c(1, 2)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
b %*% lazy_a
```

%*%,LazyColumn,LazyColumn-method

Perform the dot product between two LazyColumn vectors

Description

Computes the inner product between two LazyColumn objects.

Usage

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

Arguments

x	A LazyColumn object.
y	A LazyColumn object.

Value

A numeric value with the resulting scalar.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
X <- LazyMatrix(mat_a, "sd", "mean")
b <- rnorm(nrow(X))
lazy_col <- X[, 1]
lazy_col_2 <- X[, 2]
lazy_col %*% lazy_col_2
```

%*%,LazyColumn,numeric-method

Perform the dot product between a LazyColumn and a numeric vector

---

Description

Computes the inner product of a `LazyColumn` object and a numeric vector. If `y` is a scalar, scalar multiplication is performed. If `y` is a vector of the same length as the column, the dot product is performed.

Usage

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

Arguments

`x`	A `LazyColumn` object.
`y`	A numeric scalar or vector.

Value

A numeric value with the resulting scalar.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
X <- LazyMatrix(mat_a, "sd", "mean")
b <- rnorm(nrow(X))
lazy_col <- X[, 1]
lazy_col %*% b
```

`%*%,LazyMatrix,ANY-method`

Matrix multiplication for `LazyMatrix` *and vector*

Description

Multiplies a `LazyMatrix` object by a vector.

Usage

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

Arguments

`x`	A `LazyMatrix` object.
`y`	A numeric vector.

Value

A numeric matrix.

Examples

```
mat_a <- base::matrix(rep(1, 6), nrow=2, ncol=3)
b <- c(1, 2, 3)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_a %*% b
```

`%*%,LazyMatrix,matrix-method`

Matrix multiplication for `LazyMatrix` *and matrix-object.*

Description

Multiplies a `LazyMatrix` object by a matrix

Usage

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

Arguments

`x`	A `LazyMatrix` object.
`y`	A matrix-object.

Value

A matrix-object with the product of the lazy and non lazy object.

Examples

```
mat_a <- matrix(rep(1, 6), nrow = 2, ncol = 3)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
set.seed(123)
m <- matrix(rnorm(6), nrow = 3, ncol = 2)
lazy_a %*% m
```

%*%,numeric,LazyColumn-method

Perform the dot product between a numeric vectorand a LazyColumn

---

Description

Computes the inner product of a numeric vector and a `LazyColumn` object. If `x` is a scalar, scalar multiplication is performed. If `x` is a vector of the same length as the column, the dot product is performed.

Usage

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

Arguments

`x`	A numeric scalar or vector.
`y`	A `LazyColumn` object.

Value

A numeric value with the resulting scalar.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
X <- LazyMatrix(mat_a, "sd", "mean")
b <- rnorm(nrow(X))
lazy_col <- X[, 1]
b %*% lazy_col
```

+,ANY,LazyColumn-method

Vector Addition between regular vector and LazyColumn

Description

Sums a regular base vector and a LazyColumn vector.

Usage

```
## S4 method for signature 'ANY,LazyColumn'
e1 + e2
```

Arguments

e1	A numeric vector.
e2	A LazyColumn object.

Value

A numeric vector.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
b <- rnorm(nrow(mat_a))
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- lazy_a[, 2]
b + lazy_col
```

+,LazyColumn,ANY-method

Vector Addition between LazyColumn *and regular vector*

Description

Sums a LazyColumn vector and a regular base vector.

Usage

```
## S4 method for signature 'LazyColumn,ANY'
e1 + e2
```

Arguments

e1	A LazyColumn object.
e2	A numeric vector.

Value

A numeric vector.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
b <- rnorm(nrow(mat_a))
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- lazy_a[,2]
lazy_col + b
```

+,LazyColumn,LazyColumn-method

Vector Addition between two LazyColumn *vectors.*

Description

Sums two LazyColumn vectors.

Usage

```
## S4 method for signature 'LazyColumn,LazyColumn'
e1 + e2
```

Arguments

e1	A LazyColumn object.
e2	A LazyColumn object.

Value

A numeric vector.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- lazy_a[,2]
lazy_col_2 <- lazy_a[, 3]
lazy_col_2 + lazy_col
```

colnames,LazyMatrix-method

Retrieve or set the row or column names of a LazyMatrix object.

Description

Retrieve or set the row or column names of a LazyMatrix object.

Usage

```
## S4 method for signature 'LazyMatrix'
colnames(x)
```

Arguments

x	A LazyMatrix object.

Value

A character vector of column names, or NULL if the matrix has no column names.

Examples

```
mat_a <- base::matrix(rep(1, 6), nrow=2, ncol=3)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
colnames(lazy_a)
```

Compare,LazyColumn,numeric-method

Comparison operators for LazyColumn

Description

Enables logical comparisons (>, <, >=, <=, ==, !=) on a LazyColumn, comparing against its scaled values.

Usage

```
## S4 method for signature 'LazyColumn,numeric'
Compare(e1, e2)
```

Arguments

e1	A LazyColumn or numeric.
e2	A LazyColumn or numeric.

Value

A logical vector.

`Compare,numeric,LazyColumn-method`
Comparison operators for LazyColumn

Description

Enables logical comparisons (>, <, >=, <=, ==, !=) on a `LazyColumn`, comparing against its scaled values.

Usage

```
## S4 method for signature 'numeric,LazyColumn'
Compare(e1, e2)
```

Arguments

`e1`	A `LazyColumn` or numeric.
`e2`	A `LazyColumn` or numeric.

Value

A logical vector.

`crossprod,LazyMatrix,ANY-method`
Crossproduct for `LazyMatrix`

Description

Computes the crossproduct of a `LazyMatrix` object as it's Gram matrix or computes the transposed matrix-vector multiplication.

Usage

```
## S4 method for signature 'LazyMatrix,ANY'
crossprod(x, y = NULL)
```

Arguments

`x`	A `LazyMatrix` object.
`y`	An optional numeric vector or matrix. If NULL, computes the Gram matrix of x.

Value

A matrix: the Gram matrix if y is NULL, otherwise the crossproduct result.

Examples

```
mat_a <- matrix(rep(1, 6), nrow=2, ncol=3)
b <- c(1, 2)
lazy_a <- LazyMatrix(mat_a, scale="sd", location="mean")
crossprod(lazy_a)
crossprod(lazy_a, b)
```

---

dim,LazyMatrix-method *Returns the dimension of a LazyMarix Object.*

Description

Returns the dimension of a LazyMarix Object.

Usage

```
## S4 method for signature 'LazyMatrix'
dim(x)
```

Arguments

x	A LazyMatrix object.

Value

For an array (and hence in particular, for a matrix) dim retrieves the dim attribute of the object. It is NULL or a vector of mode integer. The replacement method changes the "dim" attribute (provided the new value is compatible) and removes any "dimnames" and "names" attributes.

Examples

```
mat_a <- base::matrix(rep(1, 6), nrow=2, ncol=3)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
dim(lazy_a)
```

lazy_crossprod_vec *Fast crossprod for LazyMatrix (dense case)*

Description

Fast crossprod for LazyMatrix (dense case)

Usage

```
lazy_crossprod_vec(x, s, c, y)
```

Arguments

x	Dense matrix
s	Column scale inverse vector (1 / col_scales)
c	Column location vector (col_locations)
y	Vector to multiply

Value

Vector result of t(X_tilde) * y

lazy_crossprod_vec_sp *Fast crossprod for LazyMatrix (sparse case)*

Description

Fast crossprod for LazyMatrix (sparse case)

Usage

```
lazy_crossprod_vec_sp(x, s, c, y)
```

Arguments

x	Sparse matrix (dgCMatrix)
s	Column scale inverse vector (1 / col_scales)
c	Column location vector (col_locations)
y	Vector to multiply

Value

Vector result of t(X_tilde) * y

LazyColumn-class *LazyColumn S4 class*

Description

An S4 class to represent a column vector as a subset of a LazyMatrix-object

Value

An object of class `LazyColumn` with slots `data` (numeric vector), `scale` (numeric scalar), and `location` (numeric scalar); represents a column of a `LazyMatrix` (scaled via scale and location).

Slots

`data` The underlying data column vector.

`scale` Numeric scalar containing column-scale parameter.

`location` Numeric scalar containing the column-location parameter.

Examples

```
mat <- matrix(1:6, nrow = 2, ncol = 3)
lazy_mat <- LazyMatrix(mat, "sd", "mean")
lazy_column <- lazy_mat[, 2]
```

LazyMatrix *Constructs a LazyMatrix object.*

Description

Constructs a LazyMatrix object.

Usage

```
LazyMatrix(data, scale = NULL, location = NULL)
```

Arguments

`data`	a matrix object.
`scale`	optional scaling parameter.
`location`	optional location parameter.

Value

A LazyMatrix object.

Examples

```
mat_a <- matrix(1:6, nrow=3, ncol=2)
lazy_a <- LazyMatrix(mat_a, scale="sd", location="mean")
lazy_a
```

LazyMatrix-class *LazyMatrix S4 class*

Description

An S4 class to represent a lazily transformed matrix with scaling and location parameters.

Value

An object of class `LazyMatrix` with slots `data` (matrix, possibly sparse), `col_scales`, `row_scales`, `col_locations`, `row_locations`. Represents the original data matrix plus stored scaling/centering parameters used for lazy operations

Slots

`data` The underlying matrix.

`col_scales` Numeric vector of column scales.

`row_scales` Numeric vector of row scales.

`col_locations` Numeric vector of column locations.

`row_locations` Numeric vector of row locations.

Examples

```
mat <- matrix(1:6, nrow=2, ncol=3)
obj <- LazyMatrix(mat, "sd", "mean")
```

---

length,LazyColumn-method *Get the length of a LazyColumn*

Description

Get the length of a LazyColumn

Usage

```
## S4 method for signature 'LazyColumn'
length(x)
```

Arguments

x	A LazyColumn object.

Value

An integer value containing the number of elements within the vector.

Examples

```
mat_a <- base::matrix(rep(1, 6), nrow=2, ncol=3)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_col <- lazy_a[, 2]
length(lazy_col)
```

---

lsqr *Performs least squares estimation on LazyMatrix object using the iterative lsqr algorithm.*

Description

Performs least squares estimation on LazyMatrix object using the iterative lsqr algorithm.

Usage

```
lsqr(x, y, ...)

## S4 method for signature 'LazyMatrix'
lsqr(x, y)
```

Arguments

x	A LazyMatrix object.
y	A response vector.
...	Additional arguments (currently unused).

Value

A Matrix-object with the regression coefficients of the covariates.

Examples

```
set.seed(123)
mat_a <- base::matrix(rnorm(500), nrow = 50, ncol = 10)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
response_vector <- rnorm(nrow(mat_a))
lsqr(lazy_a, response_vector)
```

ncol,LazyMatrix-method

Returns the number of columns of the data matrix

Description

Returns the number of columns of the data matrix

Usage

```
## S4 method for signature 'LazyMatrix'
ncol(x)
```

Arguments

x	A LazyMatrix object.

Value

an integer of length 1 or NULL.

Examples

```
mat_a <- base::matrix(rep(1, 6), nrow=2, ncol=3)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
ncol(lazy_a)
```

norm

Compute the norm of a LazyMatrix or LazyColumn

Description

Dispatches to the appropriate method based on the class of `x`.

Usage

```
norm(x, ...)
```

Arguments

x	A `LazyMatrix` or `LazyColumn` object.
...	Additional arguments passed to methods, such as `type`.

Value

For `LazyColumn`, a numeric scalar containing the Euclidean norm. For `LazyMatrix`, a numeric scalar containing the Frobenius norm.

norm,LazyColumn-method

Perform the norm of a LazyColumn vector

Description

Computes the norm of a LazyColumn object.

Usage

```
## S4 method for signature 'LazyColumn'
norm(x, type = "2")
```

Arguments

x	A LazyColumn object.
type	A character value defining type of norm. Default is Euclidean norm.

Value

A numeric value with the resulting scalar.

Examples

```
mat_a <- base::matrix(rnorm(12), nrow=3, ncol=4)
X <- LazyMatrix(mat_a, "sd", "mean")
b <- rnorm(nrow(X))
lazy_col <- X[, 1]
norm(lazy_col)
```

norm,LazyMatrix-method

Computes the Frobenius norm of a LazyMatrix *object.*

Description

Computes the Frobenius norm of a LazyMatrix object.

Usage

```
## S4 method for signature 'LazyMatrix'
norm(x)
```

Arguments

x	A LazyMatrix object.

Value

A numeric scalar representing the Frobenius norm of the matrix.

Examples

```
mat_a <- base::matrix(rnorm(50), nrow = 10, ncol = 5)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
norm(lazy_a)
```

---

nrow,LazyMatrix-method

Returns the number of rows of the data matrix

Description

Returns the number of rows of the data matrix

Usage

```
## S4 method for signature 'LazyMatrix'
nrow(x)
```

Arguments

x	A LazyMatrix object.

Value

an integer of length 1 or NULL.

Examples

```
mat_a <- base::matrix(rep(1, 6), nrow=2, ncol=3)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
nrow(lazy_a)
```

prcomp,LazyMatrix-method

Performs a principal component analysis on the LazyMatrix object using irlba:s sparse svd.

Description

Performs a principal component analysis on the LazyMatrix object using irlba:s sparse svd.

Usage

```
## S4 method for signature 'LazyMatrix'
prcomp(x, retx = TRUE, tol = NULL, rank. = NULL, ...)
```

Arguments

x	a LazyMatrix object.
retx	a logical value indicating whether the rotated variables should be returned.
tol	a value indicating the magnitude below which components should be omitted. (Components are omitted if their standard deviations are less than or equal to tol times the standard deviation of the first component.) With the default null setting, no components are omitted (unless rank. is specified less than min(dim(x)).). Other settings for tol could be tol = 0 or tol = sqrt(.Machine$double.eps), which would omit essentially constant components.
rank.	optionally, a number specifying the maximal rank, i.e., maximal number of principal components to be used. Can be set as alternative or in addition to tol, useful notably when the desired rank is considerably smaller than the dimensions of the matrix.
...	Additional arguments passed to underlying methods.

Value

A list of class \"prcomp\" containing:

sdev	The standard deviations of the principal components (i.e., the square roots of the eigenvalues of the covariance/correlation matrix, calculated using the singular values of the data matrix).
rotation	The matrix of variable loadings (columns are eigenvectors).
x	If retx is TRUE, the value of the rotated data (centered and optionally scaled, multiplied by the rotation matrix).
center	The centering used, or FALSE.
scale	The scaling applied to the data, or FALSE

Examples

```
set.seed(123)
mat_a <- matrix(rnorm(500), nrow=50, ncol=10)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
pca_lazy <- prcomp(lazy_a)
```

---

setNames,LazyColumn,character-method

Set names for a LazyColumn

Description

Assigns names to the underlying data of a `LazyColumn`, returning a new `LazyColumn` with the same scale and location as the original. This enables name-based subsetting via `[` on `LazyColumn` objects, mirroring `stats::setNames()` for ordinary vectors.

Usage

```
## S4 method for signature 'LazyColumn,character'
setNames(object = nm, nm)
```

Arguments

`object`	A `LazyColumn` object.
`nm`	A character vector of names, with length equal to `length(object@data)`.

Value

A new `LazyColumn` with named data, preserving the original scale and location.

Examples

```
set.seed(123)
mat_a <- matrix(rnorm(500), nrow = 50, ncol = 10)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_c <- lazy_a[, 2]
lazy_named <- setNames(lazy_c[1:26], letters[1:26])
lazy_named["a"]
```

subset-LazyColumn *Subset a LazyColumn*

Description

Subsets a LazyColumn object using standard R indexing rules: positive integers, negative integers, logical vectors (with recycling), character vectors (if named), zero, or missing (nothing).

Usage

```
## S4 method for signature 'LazyColumn,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]
```

Arguments

x	A LazyColumn object.
i	Index: positive integers, negative integers, logical vector, character vector (if named), zero, or missing (nothing).
j	Not used for LazyColumn (1-dimensional). Must be missing.
...	Additional arguments (ignored).
drop	Logical. Currently ignored — single element extraction always returns a plain scaled numeric, consistent with vector subsetting.

Value

A LazyColumn when multiple elements are selected, or a scaled numeric value when a single element is selected.

Examples

```
set.seed(123)
mat_a <- matrix(rnorm(500), nrow = 50, ncol = 10)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_c <- lazy_a[, 2]

# Single element → plain scaled numeric
lazy_c[2]

# Multiple elements → LazyColumn
lazy_c[c(1, 3, 5)]
```

`subset-LazyMatrix` *Subset a LazyMatrix by columns*

Description

Subsets a `LazyMatrix` object by columns, returning either a `LazyColumn` or a new `LazyMatrix` depending on the number of columns selected. Row subsetting is not yet supported.

Usage

```
## S4 method for signature 'LazyMatrix,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]
```

Arguments

`x`	A `LazyMatrix` object.
`i`	Row index. Must be missing as row subsetting is not yet supported.
`j`	Column index. Either a single integer returning a `LazyColumn`, or a vector of integers returning a `LazyMatrix`.
`...`	Additional arguments (ignored).
`drop`	Logical. Currently ignored.

Value

A `LazyColumn` if a single column is selected, or a `LazyMatrix` if multiple columns are selected.

Examples

```
A <- Matrix::sparseMatrix(i = c(1,2,3), j = c(1,2,3), x = c(1,2,3))
lazy_m <- LazyMatrix(A, "sd", "mean")

# Single column → LazyColumn
lazy_col <- lazy_m[, 2]

# Multiple columns → LazyMatrix
lazy_subset <- lazy_m[, 1:3]
```

svd,LazyMatrix-method *Singular Value decomposition for* LazyMatrix.

Description

Performs lazy SVD using irlba for partial Singular value decomposition on sparse matrices.

Usage

```
## S4 method for signature 'LazyMatrix'
svd(x, nu = min(n, p), nv = min(n, p))
```

Arguments

x	A LazyMatrix object.
nu	number of left singular vectors to estimate (defaults to nv).
nv	number of right singular vectors to estimate.

Value

A list with entries:

d	max(nu, nv) approximate singular values
u	nu approximate left singular vectors (only when right_only=FALSE)
v	nv approximate right singular vectors
iter	The number of Lanczos iterations carried out
mprod	The total number of matrix vector products carried out

Examples

```
set.seed(123)
mat_a <- matrix(rnorm(500), nrow = 50, ncol = 10)
lazy_a <-LazyMatrix(mat_a, scale = "sd", location = "mean")
S <- svd(lazy_a)
# Receive singular values with
S$d
```

t,LazyMatrix-method *Given a* LazyMatrix *x, t returns the transpose of x.*

Description

Given a LazyMatrix x, t returns the transpose of x.

Usage

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

Arguments

x	A LazyMatrix object.

Value

A LazyMatrix object with the transposed data matrix.

Examples

```
mat_a <- base::matrix(rep(1, 6), nrow=2, ncol=3)
lazy_a <- LazyMatrix(mat_a, "sd", "mean")
lazy_t <- t(lazy_a)
```

Index

*,LazyColumn,LazyColumn-method, 5
*,LazyColumn,numeric-method, 5
*,numeric,LazyColumn-method, 6
+,ANY,LazyColumn-method, 11
+,LazyColumn,ANY-method, 11
+,LazyColumn,LazyColumn-method, 12
-,ANY,LazyColumn-method, 3
-,LazyColumn,ANY-method, 3
-,LazyColumn,LazyColumn-method, 4
[,LazyColumn,ANY,ANY,ANY-method
(subset-LazyColumn), 25
[,LazyMatrix,ANY,ANY,ANY-method
(subset-LazyMatrix), 26
%*%,ANY,LazyMatrix-method, 7
%*%,LazyColumn,LazyColumn-method, 7
%*%,LazyColumn,numeric-method, 8
%*%,LazyMatrix,ANY-method, 9
%*%,LazyMatrix,matrix-method, 9
%*%,numeric,LazyColumn-method, 10

colnames,LazyMatrix-method, 13
Compare,LazyColumn,numeric-method, 13
Compare,numeric,LazyColumn-method, 14
crossprod,LazyMatrix,ANY-method, 14

dim,LazyMatrix-method, 15

lazy_crossprod_vec, 16
lazy_crossprod_vec_sp, 16
LazyColumn-class, 17
LazyMatrix, 17
LazyMatrix-class, 18
length,LazyColumn-method, 18
lsqr, 19
lsqr,LazyMatrix-method (lsqr), 19

ncol,LazyMatrix-method, 20
norm, 20
norm,LazyColumn-method, 21
norm,LazyMatrix-method, 21
nrow,LazyMatrix-method, 22

prcomp,LazyMatrix-method, 23

setNames,LazyColumn,character-method,
24
subset-LazyColumn, 25
subset-LazyMatrix, 26
svd,LazyMatrix-method, 27

t,LazyMatrix-method, 28