

# Package ‘sparseMatrixStats’

May 19, 2026

**Type** Package

**Title** Summary Statistics for Rows and Columns of Sparse Matrices

**Version** 1.25.0

**Description** High performance functions for row and column operations on sparse matrices.

For example: col / rowMeans2, col / rowMedians, col / rowVars etc. Currently, the optimizations are limited to data in the column sparse format.

This package is inspired by the matrixStats package by Henrik Bengtsson.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.3.1

**LinkingTo** Rcpp

**Imports** Rcpp, Matrix, matrixStats (>= 0.60.0), methods

**Depends** MatrixGenerics (>= 1.5.3)

**Suggests** testthat (>= 2.1.0), knitr, bench, rmarkdown, BiocStyle

**SystemRequirements** C++11

**VignetteBuilder** knitr

**URL** <https://github.com/const-ae/sparseMatrixStats>

**BugReports** <https://github.com/const-ae/sparseMatrixStats/issues>

**biocViews** Infrastructure, Software, DataRepresentation

**git\_url** <https://git.bioconductor.org/packages/sparseMatrixStats>

**git\_branch** devel

**git\_last\_commit** 9d61c99

**git\_last\_commit\_date** 2026-04-28

**Repository** Bioconductor 3.24

**Date/Publication** 2026-05-18

**Author** Constantin Ahlmann-Eltze [aut, cre] (ORCID:  
<<https://orcid.org/0000-0002-3762-068X>>)

**Maintainer** Constantin Ahlmann-Eltze <artjom31415@gmail.com>

## Contents

colAlls,xgCMatrix-method . . . . .	2
colAnyNAs,xgCMatrix-method . . . . .	4
colAnys,xgCMatrix-method . . . . .	5
colAvgPerRowSet,xgCMatrix-method . . . . .	6
colCollapse,xgCMatrix-method . . . . .	8
colCounts,xgCMatrix-method . . . . .	9
colCummaxs,dgCMatrix-method . . . . .	10
colCummins,dgCMatrix-method . . . . .	11
colCumprods,xgCMatrix-method . . . . .	12
colCumsums,xgCMatrix-method . . . . .	13
colDiffs,dgCMatrix-method . . . . .	14
colIQRDiffs,dgCMatrix-method . . . . .	16
colIQRs,xgCMatrix-method . . . . .	17
colLogSumExps,xgCMatrix-method . . . . .	18
colMadDiffs,dgCMatrix-method . . . . .	19
colMads,dgCMatrix-method . . . . .	21
colMaxs,dgCMatrix-method . . . . .	22
colMeans2,xgCMatrix-method . . . . .	23
colMedians,dgCMatrix-method . . . . .	24
colMins,dgCMatrix-method . . . . .	25
colOrderStats,dgCMatrix-method . . . . .	26
colProds,xgCMatrix-method . . . . .	27
colQuantiles,xgCMatrix-method . . . . .	29
colRanges,dgCMatrix-method . . . . .	30
colRanks,dgCMatrix-method . . . . .	31
colSdDiffs,dgCMatrix-method . . . . .	33
colSds,xgCMatrix-method . . . . .	34
colSums2,xgCMatrix-method . . . . .	36
colTabulates,xgCMatrix-method . . . . .	37
colVarDiffs,dgCMatrix-method . . . . .	38
colVars,xgCMatrix-method . . . . .	39
colWeightedMads,dgCMatrix-method . . . . .	41
colWeightedMeans,xgCMatrix-method . . . . .	42
colWeightedMedians,dgCMatrix-method . . . . .	44
colWeightedSds,xgCMatrix-method . . . . .	45
colWeightedVars,xgCMatrix-method . . . . .	46
xgCMatrix-class . . . . .	48
<b>Index</b>	<b>49</b>

---

colAlls,xgCMatrix-method

*Check if all elements in a row (column) of a matrix-like object are equal to a value*

---

### Description

Check if all elements in a row (column) of a matrix-like object are equal to a value.

## Usage

```
## S4 method for signature 'xgCMatrix'
colAlls(
  x,
  rows = NULL,
  cols = NULL,
  value = TRUE,
  na.rm = FALSE,
  useNames = TRUE
)

## S4 method for signature 'xgCMatrix'
rowAlls(
  x,
  rows = NULL,
  cols = NULL,
  value = TRUE,
  na.rm = FALSE,
  useNames = TRUE
)
```

## Arguments

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
value	The value to search for.
na.rm	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

## Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowAlls` / `matrixStats::colAlls`.

## Value

Returns a [logical vector](#) of length N (K).

## See Also

- `matrixStats::rowAlls()` and `matrixStats::colAlls()` which are used when the input is a [matrix](#) or [numeric vector](#).
- For checks if *any* element is equal to a value, see [rowAnys\(\)](#).
- `base::all()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowAlls(mat)
colAlls(mat)
```

---

colAnyNAs,xgCMatrix-method

*Check if any elements in a row (column) of a matrix-like object is missing*

---

**Description**

Check if any elements in a row (column) of a matrix-like object is missing.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colAnyNAs(x, rows = NULL, cols = NULL, useNames = TRUE)

## S4 method for signature 'xgCMatrix'
rowAnyNAs(x, rows = NULL, cols = NULL, useNames = TRUE)
```

**Arguments**

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

**Details**

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowAnyNAs` / `matrixStats::colAnyNAs`.

**Value**

Returns a [logical vector](#) of length N (K).

**See Also**

- `matrixStats::rowAnyNAs()` and `matrixStats::colAnyNAs()` which are used when the input is a matrix or numeric vector.
- For checks if any element is equal to a value, see [rowAny\(\)](#).
- `base::is.na()` and `base::any()`.

**Examples**

```
mat <- matrix(0, nrow=10, ncol=5)
mat[sample(seq_len(5 * 10), 5)] <- NA
sp_mat <- as(mat, "dgCMatrix")
colAnyNAs(sp_mat)
```

---

colAnys,xgCMatrix-method

*Check if any elements in a row (column) of a matrix-like object is equal to a value*

---

**Description**

Check if any elements in a row (column) of a matrix-like object is equal to a value.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colAnys(
  x,
  rows = NULL,
  cols = NULL,
  value = TRUE,
  na.rm = FALSE,
  useNames = TRUE
)

## S4 method for signature 'xgCMatrix'
rowAnys(
  x,
  rows = NULL,
  cols = NULL,
  value = TRUE,
  na.rm = FALSE,
  useNames = TRUE
)
```

**Arguments**

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
value	The value to search for.
na.rm	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

**Details**

The S4 methods for `x` of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowAnys` / `matrixStats::colAnys`.

**Value**

Returns a `logical vector` of length `N` (`K`).

**See Also**

- `matrixStats::rowAnys()` and `matrixStats::colAnys()` which are used when the input is a `matrix` or `numeric vector`.
- For checks if *all* elements are equal to a value, see `rowAlls()`.
- `base::any()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowAnys(mat)
colAnys(mat)
```

---

colAvgPerRowSet, xgCMatrix-method

*Calculates for each row (column) a summary statistic for equally sized subsets of columns (rows)*

---

**Description**

Calculates for each row (column) a summary statistic for equally sized subsets of columns (rows)

**Usage**

```
## S4 method for signature 'xgCMatrix'
colAvgPerRowSet(
  X,
  W = NULL,
  cols = NULL,
  S,
  FUN = colMeans2,
  ...,
  na.rm = NA,
  tFUN = FALSE
)

## S4 method for signature 'xgCMatrix'
```

```

rowAvgPerColSet(
  X,
  W = NULL,
  rows = NULL,
  S,
  FUN = rowMeans2,
  ...,
  na.rm = NA,
  tFUN = FALSE
)

```

### Arguments

X	An NxM matrix-like object.
W	An optional numeric NxM matrix of weights.
S	An <a href="#">integer</a> KxJ matrix that specifying the J subsets. Each column hold K column (row) indices for the corresponding subset. The range of values is [1, M] ([1,N]).
FUN	A row-by-row (column-by-column) summary statistic function. It is applied to each column (row) subset of X that is specified by S.
...	Additional arguments passed to FUN.
na.rm	(logical) Argument passed to FUN() as na.rm = na.rm. If NA (default), then na.rm = TRUE is used if X or S holds missing values, otherwise na.rm = FALSE.
tFUN	If TRUE, X is transposed before it is passed to FUN.
rows, cols	A <a href="#">vector</a> indicating the subset (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.

### Details

**\*\*Note\*\*:** the handling of missing parameters differs from [matrixStats::colAvgPerRowSet()]. The 'matrixStats' version always removes 'NA's if there are any in the data. This method however does whatever is passed in the '...' parameter.

### Value

Returns a numeric JxN (MxJ) matrix.

### See Also

- `matrixStats::rowAvgPerColSet()` and `matrixStats::colAvgPerRowSet()` which are used when the input is a matrix or numeric vector.

### Examples

```

mat <- matrix(rnorm(20), nrow = 5, ncol = 4)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

S <- matrix(1:ncol(mat), ncol = 2)
print(S)

```

```
rowAvgPerColSet(mat, S = S, FUN = rowMeans)
rowAvgPerColSet(mat, S = S, FUN = rowVars)
```

---

colCollapse,xgCMatrix-method

*Extract one cell from each row (column) of a matrix-like object*

---

## Description

Extract one cell from each row (column) of a matrix-like object.

## Usage

```
## S4 method for signature 'xgCMatrix'
colCollapse(x, idxs, cols = NULL, useNames = TRUE)
```

```
## S4 method for signature 'xgCMatrix'
rowCollapse(x, idxs, rows = NULL, useNames = TRUE)
```

## Arguments

x	An NxK matrix-like object.
idxs	An index <a href="#">vector</a> with the position to extract. It is recycled to match the number of rows (column)
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.

## Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowCollapse` / `matrixStats::colCollapse`.

## Value

Returns a [numeric vector](#) of length N (K).

## See Also

- `matrixStats::rowCollapse()` and `matrixStats::colCollapse()` which are used when the input is a matrix or numeric vector.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowCollapse(mat, idxs = 2)
rowCollapse(mat, idxs = c(1,1,2,3,2))

colCollapse (mat, idxs = 4)
```

---

colCounts,xgCMatrix-method

*Count how often an element in a row (column) of a matrix-like object is equal to a value*

---

**Description**

Count how often an element in a row (column) of a matrix-like object is equal to a value.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colCounts(
  x,
  rows = NULL,
  cols = NULL,
  value = TRUE,
  na.rm = FALSE,
  useNames = TRUE
)

## S4 method for signature 'xgCMatrix'
rowCounts(
  x,
  rows = NULL,
  cols = NULL,
  value = TRUE,
  na.rm = FALSE,
  useNames = TRUE
)
```

**Arguments**

x	An $N \times K$ matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
value	The value to search for.
na.rm	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.

useNames            If **TRUE** (default), names attributes of result are set. Else if **FALSE**, no naming support is done.

### Details

The S4 methods for *x* of type **matrix**, **array**, **table**, or **numeric** call `matrixStats::rowCounts / matrixStats::colCounts`.

### Value

Returns a **integer vector** of length *N* (*K*).

### See Also

- `matrixStats::rowCounts()` and `matrixStats::colCounts()` which are used when the input is a **matrix** or **numeric vector**.
- For checks if any element is equal to a value, see `rowAnys()`. To check if all elements are equal, see `rowAlls()`.

### Examples

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowCounts(mat)
colCounts(mat)
rowCounts(mat, value = 0)
colCounts(mat, value = Inf, na.rm = TRUE)
```

---

colCummaxs,dgCMatrix-method

*Calculates the cumulative maxima for each row (column) of a matrix-like object*

---

### Description

Calculates the cumulative maxima for each row (column) of a matrix-like object.

### Usage

```
## S4 method for signature 'dgCMatrix'
colCummaxs(x, rows = NULL, cols = NULL, useNames = TRUE)

## S4 method for signature 'dgCMatrix'
rowCummaxs(x, rows = NULL, cols = NULL, useNames = TRUE)
```

**Arguments**

<code>x</code>	An NxK matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
<code>useNames</code>	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

**Details**

The S4 methods for `x` of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowCummaxs` / `matrixStats::colCummaxs`.

**Value**

Returns a [numeric matrix](#) with the same dimensions as `x`.

**See Also**

- `matrixStats::rowCummaxs()` and `matrixStats::colCummaxs()` which are used when the input is a [matrix](#) or [numeric vector](#).
- For single maximum estimates, see `rowMaxs()`.
- `base::cummax()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowCummaxs(mat)
colCummaxs(mat)
```

---

colCummins,dgCMatrix-method

*Calculates the cumulative minima for each row (column) of a matrix-like object*

---

**Description**

Calculates the cumulative minima for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'dgCMatrix'
colCummins(x, rows = NULL, cols = NULL, useNames = TRUE)

## S4 method for signature 'dgCMatrix'
rowCummins(x, rows = NULL, cols = NULL, useNames = TRUE)
```

**Arguments**

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

**Details**

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowCummins` / `matrixStats::colCummins`.

**Value**

Returns a [numeric matrix](#) with the same dimensions as x.

**See Also**

- `matrixStats::rowCummins()` and `matrixStats::colCummins()` which are used when the input is a [matrix](#) or [numeric vector](#).
- For single minimum estimates, see `rowMins()`.
- `base::cummin()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowCummins(mat)
colCummins(mat)
```

---

colCumprods,xgCMatrix-method

*Calculates the cumulative product for each row (column) of a matrix-like object*

---

**Description**

Calculates the cumulative product for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colCumprods(x, rows = NULL, cols = NULL, useNames = TRUE)

## S4 method for signature 'xgCMatrix'
rowCumprods(x, rows = NULL, cols = NULL, useNames = TRUE)
```

**Arguments**

<code>x</code>	An NxK matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
<code>useNames</code>	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

**Details**

The S4 methods for `x` of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowCumprods` / `matrixStats::colCumprods`.

**Value**

Returns a [numeric matrix](#) with the same dimensions as `x`.

**See Also**

- `matrixStats::rowCumprods()` and `matrixStats::colCumprods()` which are used when the input is a matrix or numeric vector.
- `base::cumprod()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowCumprods(mat)
colCumprods(mat)
```

---

colCumsums,xgCMatrix-method

*Calculates the cumulative sum for each row (column) of a matrix-like object*

---

**Description**

Calculates the cumulative sum for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colCumsums(x, rows = NULL, cols = NULL, useNames = TRUE)

## S4 method for signature 'xgCMatrix'
rowCumsums(x, rows = NULL, cols = NULL, useNames = TRUE)
```

**Arguments**

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

**Details**

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowCumsums` / `matrixStats::colCumsums`.

**Value**

Returns a [numeric matrix](#) with the same dimensions as x.

**See Also**

- `matrixStats::rowCumsums()` and `matrixStats::colCumsums()` which are used when the input is a matrix or numeric vector.
- `base::cumsum()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowCumsums(mat)
colCumsums(mat)
```

---

colDiffs,dgCMatrix-method

*Calculates the difference between each element of a row (column) of a matrix-like object*

---

**Description**

Calculates the difference between each element of a row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'dgCMatrix'
colDiffs(
  x,
  rows = NULL,
  cols = NULL,
  lag = 1L,
```

```

    differences = 1L,
    useNames = TRUE
  )

## S4 method for signature 'dgCMatrix'
rowDiffs(
  x,
  rows = NULL,
  cols = NULL,
  lag = 1L,
  differences = 1L,
  useNames = TRUE
)

```

### Arguments

<code>x</code>	An $N \times K$ matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
<code>lag</code>	An integer specifying the lag.
<code>differences</code>	An integer specifying the order of difference.
<code>useNames</code>	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

### Details

The S4 methods for `x` of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowDiffs` / `matrixStats::colDiffs`.

### Value

Returns a [numeric matrix](#) with one column (row) less than `x`:  $N \times (K - 1)$  or  $(N - 1) \times K$ .

### See Also

- `matrixStats::rowDiffs()` and `matrixStats::colDiffs()` which are used when the input is a [matrix](#) or [numeric vector](#).
- `base::diff()`.

### Examples

```

mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowDiffs(mat)
colDiffs(mat)

```

---

 colIQRDiffs,dgCMatrix-method

*Calculates the interquartile range of the difference between each element of a row (column) of a matrix-like object*

---

## Description

Calculates the interquartile range of the difference between each element of a row (column) of a matrix-like object.

## Usage

```
## S4 method for signature 'dgCMatrix'
colIQRDiffs(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  diff = 1L,
  trim = 0,
  useNames = TRUE
)

## S4 method for signature 'dgCMatrix'
rowIQRDiffs(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  diff = 1L,
  trim = 0,
  useNames = TRUE
)
```

## Arguments

<code>x</code>	An NxK matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
<code>na.rm</code>	If <a href="#">TRUE</a> , missing values (NA or NaN) are omitted from the calculations.
<code>diff</code>	An integer specifying the order of difference.
<code>trim</code>	A double in [0,1/2] specifying the fraction of observations to be trimmed from each end of (sorted) x before estimation.
<code>useNames</code>	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

## Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowIQRDiffs` / `matrixStats::colIQRDiffs`.

**Value**

Returns a [numeric vector](#) of length N (K).

**See Also**

- `matrixStats::rowIQRDiffs()` and `matrixStats::colIQRDiffs()` which are used when the input is a matrix or numeric vector.
- For the direct interquartile range see also [rowIQRs](#).

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowIQRDiffs(mat)
colIQRDiffs(mat)
```

---

colIQRs,xgCMatrix-method

*Calculates the interquartile range for each row (column) of a matrix-like object*

---

**Description**

Calculates the interquartile range for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colIQRs(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)

## S4 method for signature 'xgCMatrix'
rowIQRs(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

**Arguments**

<code>x</code>	An NxK matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <code>NULL</code> , no subsetting is done.
<code>na.rm</code>	If <code>TRUE</code> , missing values ( <code>NA</code> or <code>NaN</code> ) are omitted from the calculations.
<code>useNames</code>	If <code>TRUE</code> (default), names attributes of result are set. Else if <code>FALSE</code> , no naming support is done.

**Details**

The S4 methods for `x` of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowIQRs` / `matrixStats::colIQRs`.

**Value**

Returns a [numeric vector](#) of length N (K).

**See Also**

- `matrixStats::rowIQRs()` and `matrixStats::colIQRs()` which are used when the input is a matrix or numeric vector.
- For a non-robust analog, see `rowSds()`. For a more robust version see `rowMads()`
- `stats::IQR()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowIQRs(mat)
colIQRs(mat)
```

---

colLogSumExps,xgCMatrix-method

*Accurately calculates the logarithm of the sum of exponentials for each row (column) of a matrix-like object*

---

**Description**

Accurately calculates the logarithm of the sum of exponentials for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colLogSumExps(lx, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)

## S4 method for signature 'xgCMatrix'
rowLogSumExps(lx, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

**Arguments**

<code>lx</code>	An NxK matrix-like object. Typically <code>lx</code> are $\log(x)$ values.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset (and/or columns) to operate over. If <code>NULL</code> , no subsetting is done.
<code>na.rm</code>	If <code>TRUE</code> , missing values ( <code>NA</code> or <code>NaN</code> ) are omitted from the calculations.
<code>useNames</code>	If <code>TRUE</code> (default), names attributes of result are set. Else if <code>FALSE</code> , no naming support is done.

**Details**

The S4 methods for `x` of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowLogSumExps` / `matrixStats::colLogSumExps`.

**Value**

Returns a `numeric vector` of length `N (K)`.

**See Also**

- `matrixStats::rowLogSumExps()` and `matrixStats::colLogSumExps()` which are used when the input is a matrix or numeric vector.
- `rowSums2()`

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowLogSumExps(mat)
colLogSumExps(mat)
```

---

colMadDiffs,dgCMatrix-method

*Calculates the mean absolute deviation of the difference between each element of a row (column) of a matrix-like object*

---

**Description**

Calculates the mean absolute deviation of the difference between each element of a row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'dgCMatrix'
colMadDiffs(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  diff = 1L,
  trim = 0,
  constant = 1.4826,
  ...,
  useNames = TRUE
)
```

```
## S4 method for signature 'dgCMatrix'
rowMadDiffs(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  diff = 1L,
  trim = 0,
  constant = 1.4826,
  ...,
  useNames = TRUE
)
```

### Arguments

<code>x</code>	An $N \times K$ matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <code>NULL</code> , no subsetting is done.
<code>na.rm</code>	If <code>TRUE</code> , missing values ( <code>NA</code> or <code>NaN</code> ) are omitted from the calculations.
<code>diff</code>	An integer specifying the order of difference.
<code>trim</code>	A double in $[0, 1/2]$ specifying the fraction of observations to be trimmed from each end of (sorted) <code>x</code> before estimation.
<code>constant</code>	A scale factor. See <a href="#">mad</a> for details.
<code>...</code>	Additional arguments passed to specific methods.
<code>useNames</code>	If <code>TRUE</code> (default), names attributes of result are set. Else if <code>FALSE</code> , no naming support is done.

### Details

The S4 methods for `x` of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowMadDiffs` / `matrixStats::colMadDiffs`.

### Value

Returns a [numeric vector](#) of length  $N$  ( $K$ ).

### See Also

- `matrixStats::rowMadDiffs()` and `matrixStats::colMadDiffs()` which are used when the input is a `matrix` or `numeric` vector.

### Examples

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowMadDiffs(mat)
colMadDiffs(mat)
```

---

 colMads,dgCMatrix-method

*Calculates the median absolute deviation for each row (column) of a matrix-like object*

---

## Description

Calculates the median absolute deviation for each row (column) of a matrix-like object.

## Usage

```
## S4 method for signature 'dgCMatrix'
colMads(
  x,
  rows = NULL,
  cols = NULL,
  center = NULL,
  constant = 1.4826,
  na.rm = FALSE,
  useNames = TRUE
)

## S4 method for signature 'dgCMatrix'
rowMads(
  x,
  rows = NULL,
  cols = NULL,
  center = NULL,
  constant = 1.4826,
  na.rm = FALSE,
  useNames = TRUE
)
```

## Arguments

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
center	(optional) the center, defaults to the row means
constant	A scale factor. See <code>stats::mad()</code> for details.
na.rm	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

## Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowMads / matrixStats::colMads`.

**Value**

Returns a [numeric vector](#) of length N (K).

**See Also**

- `matrixStats::rowMads()` and `matrixStats::colMads()` which are used when the input is a `matrix` or `numeric vector`.
- For mean estimates, see `rowMeans2()` and `rowMeans()`.
- For non-robust standard deviation estimates, see `rowSds()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowMads(mat)
colMads(mat)
```

---

colMaxs,dgCMatrix-method

*Calculates the maximum for each row (column) of a matrix-like object*

---

**Description**

Calculates the maximum for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'dgCMatrix'
colMaxs(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)

## S4 method for signature 'dgCMatrix'
rowMaxs(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

**Arguments**

<code>x</code>	An NxK matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <code>NULL</code> , no subsetting is done.
<code>na.rm</code>	If <code>TRUE</code> , missing values ( <code>NA</code> or <code>NaN</code> ) are omitted from the calculations.
<code>useNames</code>	If <code>TRUE</code> (default), names attributes of result are set. Else if <code>FALSE</code> , no naming support is done.

**Details**

The S4 methods for `x` of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowMaxs / matrixStats::colMaxs`.

**Value**

Returns a [numeric vector](#) of length N (K).

**See Also**

- `matrixStats::rowMaxs()` and `matrixStats::colMaxs()` which are used when the input is a [matrix](#) or [numeric vector](#).
- For min estimates, see `rowMins()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowMaxs(mat)
colMaxs(mat)
```

---

colMeans2,xgCMatrix-method

*Calculates the mean for each row (column) of a matrix-like object*

---

**Description**

Calculates the mean for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colMeans2(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)

## S4 method for signature 'xgCMatrix'
rowMeans2(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

**Arguments**

<code>x</code>	An $N \times K$ matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
<code>na.rm</code>	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
<code>useNames</code>	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

**Details**

The S4 methods for `x` of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowMeans2 / matrixStats::colMeans2`.

**Value**

Returns a [numeric vector](#) of length N (K).

**See Also**

- `matrixStats::rowMeans2()` and `matrixStats::colMeans2()` which are used when the input is a [matrix](#) or [numeric vector](#).
- See also [rowMeans\(\)](#) for the corresponding function in base R.
- For variance estimates, see [rowVars\(\)](#).
- See also the base R version `base::rowMeans()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowMeans2(mat)
colMeans2(mat)
```

---

colMedians,dgCMatrix-method

*Calculates the median for each row (column) of a matrix-like object*

---

**Description**

Calculates the median for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'dgCMatrix'
colMedians(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)

## S4 method for signature 'dgCMatrix'
rowMedians(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

**Arguments**

<code>x</code>	An NxK matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
<code>na.rm</code>	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
<code>useNames</code>	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

**Details**

The S4 methods for `x` of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowMedians` / `matrixStats::colMedians`.

**Value**

Returns a `numeric vector` of length `N (K)`.

**See Also**

- `matrixStats::rowMedians()` and `matrixStats::colMedians()` which are used when the input is a `matrix` or `numeric vector`.
- For mean estimates, see `rowMeans2()` and `rowMeans()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowMedians(mat)
colMedians(mat)
```

---

colMins,dgCMatrix-method

*Calculates the minimum for each row (column) of a matrix-like object*

---

**Description**

Calculates the minimum for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'dgCMatrix'
colMins(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)

## S4 method for signature 'dgCMatrix'
rowMins(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

**Arguments**

<code>x</code>	An <code>NxK</code> matrix-like object.
<code>rows, cols</code>	A <code>vector</code> indicating the subset of rows (and/or columns) to operate over. If <code>NULL</code> , no subsetting is done.
<code>na.rm</code>	If <code>TRUE</code> , missing values ( <code>NA</code> or <code>NaN</code> ) are omitted from the calculations.
<code>useNames</code>	If <code>TRUE</code> (default), names attributes of result are set. Else if <code>FALSE</code> , no naming support is done.

**Details**

The S4 methods for `x` of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowMins` / `matrixStats::colMins`.

**Value**

Returns a `numeric vector` of length `N` (`K`).

**See Also**

- `matrixStats::rowMins()` and `matrixStats::colMins()` which are used when the input is a `matrix` or `numeric vector`.
- For max estimates, see `rowMaxs()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowMins(mat)
colMins(mat)
```

---

colOrderStats,dgCMatrix-method

*Calculates an order statistic for each row (column) of a matrix-like object*

---

**Description**

Calculates an order statistic for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'dgCMatrix'
colOrderStats(
  x,
  rows = NULL,
  cols = NULL,
  which = 1,
  na.rm = FALSE,
  useNames = TRUE
)

## S4 method for signature 'dgCMatrix'
rowOrderStats(
  x,
  rows = NULL,
```

```

  cols = NULL,
  which = 1,
  na.rm = FALSE,
  useNames = TRUE
)

```

### Arguments

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
which	An integer index in [1,K] ([1,N]) indicating which order statistic to be returned
na.rm	If TRUE, NAs are excluded first, otherwise not.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

### Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowOrderStats` / `matrixStats::colOrderStats`.

### Value

Returns a [numeric vector](#) of length N (K).

### See Also

- `matrixStats::rowOrderStats()` and `matrixStats::colOrderStats()` which are used when the input is a [matrix](#) or [numeric vector](#).

### Examples

```

mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- 2
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowOrderStats(mat, which = 1)
colOrderStats(mat, which = 3)

```

---

colProds, xgCMatrix-method

*Calculates the product for each row (column) in a matrix*

---

### Description

Calculates the product for each row (column) in a matrix

**Usage**

```
## S4 method for signature 'xgCMatrix'
colProds(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)

## S4 method for signature 'xgCMatrix'
rowProds(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

**Arguments**

<code>x</code>	An $N \times K$ matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <code>NULL</code> , no subsetting is done.
<code>na.rm</code>	If <code>TRUE</code> , missing values ( <code>NA</code> or <code>NaN</code> ) are omitted from the calculations.
<code>useNames</code>	If <code>TRUE</code> (default), names attributes of result are set. Else if <code>FALSE</code> , no naming support is done.

**Details**

Attention: This method ignores the order of the values, because it assumes that the product is commutative. Unfortunately, for 'double' this is not true. For example `'NaN * NA = NaN'`, but `'NA * NaN = NA'`. This is relevant for this function if there are `'+-Inf'`, because `'Inf * 0 = NaN'`. This function returns `'NA'` whenever there is `'NA'` in the input. This is different from `'matrixStats::colProds()'`.

**Value**

Returns a [numeric vector](#) of length  $N$  ( $K$ ).

**See Also**

- `matrixStats::rowProds()` and `matrixStats::colProds()` which are used when the input is a matrix or numeric vector.
- For sums across rows (columns), see `rowSums2()` (`colSums2()`)
- `base::prod()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowProds(mat)
colProds(mat)
```

---

 colQuantiles,xgCMatrix-method

*Calculates quantiles for each row (column) of a matrix-like object*


---

### Description

Calculates quantiles for each row (column) of a matrix-like object.

### Usage

```
## S4 method for signature 'xgCMatrix'
colQuantiles(
  x,
  rows = NULL,
  cols = NULL,
  probs = seq(from = 0, to = 1, by = 0.25),
  na.rm = FALSE,
  type = 7L,
  useNames = TRUE,
  drop = TRUE
)

## S4 method for signature 'xgCMatrix'
rowQuantiles(
  x,
  rows = NULL,
  cols = NULL,
  probs = seq(from = 0, to = 1, by = 0.25),
  na.rm = FALSE,
  type = 7L,
  useNames = TRUE,
  drop = TRUE
)
```

### Arguments

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
probs	A numeric vector of J probabilities in [0, 1].
na.rm	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
type	An integer specifying the type of estimator. See <code>stats::quantile()</code> . for more details.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.
drop	If <a href="#">TRUE</a> a vector is returned if J == 1.

### Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowQuantiles` / `matrixStats::colQuantiles`.

**Value**

a numeric  $N \times J$  ( $K \times J$ ) *matrix*, where  $N$  ( $K$ ) is the number of rows (columns) for which the  $J$  values are calculated.

**See Also**

- `matrixStats::rowQuantiles()` and `matrixStats::colQuantiles()` which are used when the input is a matrix or numeric vector.
- `stats::quantile`

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowQuantiles(mat)
colQuantiles(mat)
```

---

colRanges,dgCMatrix-method

*Calculates the minimum and maximum for each row (column) of a matrix-like object*

---

**Description**

Calculates the minimum and maximum for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'dgCMatrix'
colRanges(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)

## S4 method for signature 'dgCMatrix'
rowRanges(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

**Arguments**

<code>x</code>	An $N \times K$ matrix-like object.
<code>rows, cols</code>	A <i>vector</i> indicating the subset of rows (and/or columns) to operate over. If <code>NULL</code> , no subsetting is done.
<code>na.rm</code>	If <code>TRUE</code> , missing values ( <code>NA</code> or <code>NaN</code> ) are omitted from the calculations.
<code>useNames</code>	If <code>TRUE</code> (default), names attributes of result are set. Else if <code>FALSE</code> , no naming support is done.

**Details**

The S4 methods for `x` of type *matrix*, *array*, *table*, or *numeric* call `matrixStats::rowRanges / matrixStats::colRanges`.

**Value**

a numeric  $N \times 2$  ( $K \times 2$ ) *matrix*, where  $N$  ( $K$ ) is the number of rows (columns) for which the ranges are calculated.

**Note**

Unfortunately for the argument list of the `rowRanges()` generic function we cannot follow the scheme used for the other row/column matrix summarization generic functions. This is because we need to be compatible with the historic `rowRanges()` getter for `RangedSummarizedExperiment` objects. See `?SummarizedExperiment::rowRanges`.

**See Also**

- `matrixStats::rowRanges()` and `matrixStats::colRanges()` which are used when the input is a *matrix* or numeric vector.
- For max estimates, see `rowMaxs()`.
- For min estimates, see `rowMins()`.
- `base::range()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowRanges(mat)
colRanges(mat)
```

---

colRanks,dgCMatrix-method

*Calculates the rank of the elements for each row (column) of a matrix-like object*

---

**Description**

Calculates the rank of the elements for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'dgCMatrix'
colRanks(
  x,
  rows = NULL,
  cols = NULL,
  ties.method = c("max", "average", "min"),
  preserveShape = FALSE,
  na.handling = c("keep", "last"),
  ...,
```

```

    useNames = TRUE
  )

## S4 method for signature 'dgCMatrix'
rowRanks(
  x,
  rows = NULL,
  cols = NULL,
  ties.method = c("max", "average", "min"),
  preserveShape = TRUE,
  na.handling = c("keep", "last"),
  ...,
  useNames = TRUE
)

```

### Arguments

<code>x</code>	An NxK matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <code>NULL</code> , no subsetting is done.
<code>ties.method</code>	A character string specifying how ties are treated. Note that the default specifies fewer options than the original <code>matrixStats</code> package.
<code>preserveShape</code>	a boolean that specifies if the returned matrix has the same dimensions as the input matrix. By default this is true for <code>'rowRanks()'</code> , but false for <code>'colRanks()'</code> .
<code>na.handling</code>	string specifying how 'NA's are handled. They can either be preserved with an 'NA' rank ('keep') or sorted in at the end ('last'). Default is 'keep' derived from the behavior of the equivalent
<code>...</code>	Additional arguments passed to specific methods.
<code>useNames</code>	If <code>TRUE</code> (default), names attributes of result are set. Else if <code>FALSE</code> , no naming support is done.

### Details

There are three different methods available for handling ties:

**'max'** for values with identical values the maximum rank is returned

**'average'** for values with identical values the average of the ranks they cover is returned. Note, that in this case the return value is of type 'numeric'.

**'min'** for values with identical values the minimum rank is returned.

### Value

a matrix of type [integer](#) is returned unless `ties.method = "average"`. It has dimensions 'NxJ (KxJ) [matrix](#), where N (K) is the number of rows (columns) of the input x.

### See Also

- `matrixStats::rowRanks()` and `matrixStats::colRanks()` which are used when the input is a `matrix` or `numeric` vector.
- [base::rank](#)

**Examples**

```

mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowRanks(mat)
colRanks(mat)

```

---

colSdDiffs,dgCMatrix-method

*Calculates the standard deviation of the difference between each element of a row (column) of a matrix-like object*

---

**Description**

Calculates the standard deviation of the difference between each element of a row (column) of a matrix-like object.

**Usage**

```

## S4 method for signature 'dgCMatrix'
colSdDiffs(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  diff = 1L,
  trim = 0,
  useNames = TRUE
)

## S4 method for signature 'dgCMatrix'
rowSdDiffs(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  diff = 1L,
  trim = 0,
  useNames = TRUE
)

```

**Arguments**

<code>x</code>	An <code>NxK</code> matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <code>NULL</code> , no subsetting is done.
<code>na.rm</code>	If <code>TRUE</code> , missing values ( <code>NA</code> or <code>NaN</code> ) are omitted from the calculations.

diff	An integer specifying the order of difference.
trim	A double in $[0, 1/2]$ specifying the fraction of observations to be trimmed from each end of (sorted) $x$ before estimation.
useNames	If <code>TRUE</code> (default), names attributes of result are set. Else if <code>FALSE</code> , no naming support is done.

### Details

The S4 methods for  $x$  of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowSdDiffs` / `matrixStats::colSdDiffs`.

### Value

Returns a `numeric vector` of length  $N$  ( $K$ ).

### See Also

- `matrixStats::rowSdDiffs()` and `matrixStats::colSdDiffs()` which are used when the input is a `matrix` or `numeric vector`.
- for the direct standard deviation see `rowSds()`.

### Examples

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowSdDiffs(mat)
colSdDiffs(mat)
```

---

colSds, xgCMatrix-method

*Calculates the standard deviation for each row (column) of a matrix-like object*

---

### Description

Calculates the standard deviation for each row (column) of a matrix-like object.

### Usage

```
## S4 method for signature 'xgCMatrix'
colSds(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  center = NULL,
  useNames = TRUE
```

```

)

## S4 method for signature 'xgCMatrix'
rowSds(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  center = NULL,
  useNames = TRUE
)

```

### Arguments

<code>x</code>	An NxK matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
<code>na.rm</code>	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
<code>center</code>	(optional) the center, defaults to the row means
<code>useNames</code>	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

### Details

The S4 methods for `x` of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowSds` / `matrixStats::colSds`.

### Value

Returns a [numeric vector](#) of length N (K).

### See Also

- `matrixStats::rowSds()` and `matrixStats::colSds()` which are used when the input is a [matrix](#) or [numeric vector](#).
- For mean estimates, see [rowMeans2\(\)](#) and [rowMeans\(\)](#).
- For variance estimates, see [rowVars\(\)](#).

### Examples

```

mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowSds(mat)
colSds(mat)

```

---

 colSums2,xgCMatrix-method

*Calculates the sum for each row (column) of a matrix-like object*


---

### Description

Calculates the sum for each row (column) of a matrix-like object.

### Usage

```
## S4 method for signature 'xgCMatrix'
colSums2(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

```
## S4 method for signature 'xgCMatrix'
rowSums2(x, rows = NULL, cols = NULL, na.rm = FALSE, useNames = TRUE)
```

### Arguments

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
na.rm	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

### Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowSums2 / matrixStats::colSums2`.

### Value

Returns a [numeric vector](#) of length N (K).

### See Also

- `matrixStats::rowSums2()` and `matrixStats::colSums2()` which are used when the input is a [matrix](#) or [numeric vector](#).
- For mean estimates, see `rowMeans2()` and `rowMeans()`.
- `base::sum()`.

### Examples

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowSums2(mat)
colSums2(mat)
```

---

 colTabulates, xgCMatrix-method

*Tabulates the values in a matrix-like object by row (column)*


---

## Description

Tabulates the values in a matrix-like object by row (column).

## Usage

```
## S4 method for signature 'xgCMatrix'
colTabulates(x, rows = NULL, cols = NULL, values = NULL, useNames = TRUE)

## S4 method for signature 'xgCMatrix'
rowTabulates(x, rows = NULL, cols = NULL, values = NULL, useNames = TRUE)
```

## Arguments

x	An NxK matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
values	the values to search for.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

## Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowTabulates` / `matrixStats::colTabulates`.

## Value

a [numeric](#) NxJ (KxJ) [matrix](#), where N (K) is the number of rows (columns) for which the J values are calculated.

## See Also

- `matrixStats::rowTabulates()` and `matrixStats::colTabulates()` which are used when the input is a [matrix](#) or [numeric](#) vector.
- `base::table()`

## Examples

```
mat <- matrix(rpois(15, lambda = 3), nrow = 5, ncol = 3)
mat[2, 1] <- NA_integer_
mat[3, 3] <- 0L
mat[4, 1] <- 0L

print(mat)

rowTabulates(mat)
```

```
colTabulates(mat)

rowTabulates(mat, values = 0)
colTabulates(mat, values = 0)
```

---

```
colVarDiffs,dgCMatrix-method
```

*Calculates the variance of the difference between each element of a row (column) of a matrix-like object*

---

## Description

Calculates the variance of the difference between each element of a row (column) of a matrix-like object.

## Usage

```
## S4 method for signature 'dgCMatrix'
colVarDiffs(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  diff = 1L,
  trim = 0,
  useNames = TRUE
)

## S4 method for signature 'dgCMatrix'
rowVarDiffs(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  diff = 1L,
  trim = 0,
  useNames = TRUE
)
```

## Arguments

<code>x</code>	An $N \times K$ matrix-like object.
<code>rows, cols</code>	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
<code>na.rm</code>	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
<code>diff</code>	An integer specifying the order of difference.
<code>trim</code>	A double in $[0, 1/2]$ specifying the fraction of observations to be trimmed from each end of (sorted) <code>x</code> before estimation.
<code>useNames</code>	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

**Details**

The S4 methods for `x` of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowVarDiffs` / `matrixStats::colVarDiffs`.

**Value**

Returns a `numeric vector` of length `N (K)`.

**See Also**

- `matrixStats::rowVarDiffs()` and `matrixStats::colVarDiffs()` which are used when the input is a `matrix` or `numeric vector`.
- for the direct variance see `rowVars()`.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowVarDiffs(mat)
colVarDiffs(mat)
```

---

colVars, xgCMatrix-method

*Calculates the variance for each row (column) of a matrix-like object*

---

**Description**

Calculates the variance for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colVars(
  x,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  center = NULL,
  useNames = TRUE
)

## S4 method for signature 'xgCMatrix'
rowVars(
  x,
  rows = NULL,
  cols = NULL,
```

```

na.rm = FALSE,
center = NULL,
useNames = TRUE
)

```

### Arguments

x	An N×K matrix-like object.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
na.rm	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
center	(optional) the center, defaults to the row means.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

### Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowVars / matrixStats::colVars`.

### Value

Returns a [numeric vector](#) of length N (K).

### See Also

- `matrixStats::rowVars()` and `matrixStats::colVars()` which are used when the input is a [matrix](#) or [numeric vector](#).
- For mean estimates, see `rowMeans2()` and `rowMeans()`.
- For standard deviation estimates, see `rowSds()`.
- `stats::var()`.

### Examples

```

mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)

rowVars(mat)
colVars(mat)

```

---

 colWeightedMads,dgCMatrix-method

*Calculates the weighted median absolute deviation for each row (column) of a matrix-like object*

---

## Description

Calculates the weighted median absolute deviation for each row (column) of a matrix-like object.

## Usage

```
## S4 method for signature 'dgCMatrix'
colWeightedMads(
  x,
  w = NULL,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  constant = 1.4826,
  center = NULL,
  useNames = TRUE
)

## S4 method for signature 'dgCMatrix'
rowWeightedMads(
  x,
  w = NULL,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  constant = 1.4826,
  center = NULL,
  useNames = TRUE
)
```

## Arguments

<code>x</code>	An NxK matrix-like object.
<code>w</code>	A <b>numeric</b> vector of length K (N) that specifies by how much each element is weighted.
<code>rows, cols</code>	A <b>vector</b> indicating the subset of rows (and/or columns) to operate over. If <b>NULL</b> , no subsetting is done.
<code>na.rm</code>	If <b>TRUE</b> , missing values ( <b>NA</b> or <b>NaN</b> ) are omitted from the calculations.
<code>constant</code>	A scale factor. See <code>stats::mad()</code> for details.
<code>center</code>	Not supported at the moment.
<code>useNames</code>	If <b>TRUE</b> (default), names attributes of result are set. Else if <b>FALSE</b> , no naming support is done.

**Details**

The S4 methods for `x` of type `matrix`, `array`, `table`, or `numeric` call `matrixStats::rowWeightedMads` / `matrixStats::colWeightedMads`.

**Value**

Returns a `numeric vector` of length `N (K)`.

**See Also**

- `matrixStats::rowWeightedMads()` and `matrixStats::colWeightedMads()` which are used when the input is a `matrix` or `numeric vector`.
- See also `rowMads` for the corresponding unweighted function.

**Examples**

```
mat <- matrix(0, nrow=10, ncol=5)
mat[sample(prod(dim(mat)), 25)] <- rpois(n=25, 5)
sp_mat <- as(mat, "dgCMatrix")
weights <- rnorm(10, mean=1, sd=0.1)

# sparse version
sparseMatrixStats::colWeightedMads(sp_mat, weights)

# Attention the result differs from matrixStats
# because it always uses 'interpolate=FALSE'.
matrixStats::colWeightedMads(mat, weights)
```

---

colWeightedMeans, xgCMatrix-method

*Calculates the weighted mean for each row (column) of a matrix-like object*

---

**Description**

Calculates the weighted mean for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colWeightedMeans(
  x,
  w = NULL,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  useNames = TRUE
)

## S4 method for signature 'xgCMatrix'
rowWeightedMeans(
```

```

x,
w = NULL,
rows = NULL,
cols = NULL,
na.rm = FALSE,
useNames = TRUE
)

```

### Arguments

x	An NxK matrix-like object.
w	A <a href="#">numeric</a> vector of length K (N) that specifies by how much each element is weighted.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
na.rm	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

### Details

The S4 methods for x of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowWeightedMeans` / `matrixStats::colWeightedMeans`.

### Value

Returns a [numeric vector](#) of length N (K).

### See Also

- `matrixStats::rowWeightedMeans()` and `matrixStats::colWeightedMeans()` which are used when the input is a [matrix](#) or [numeric vector](#).
- See also [rowMeans2](#) for the corresponding unweighted function.

### Examples

```

mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)
w <- rnorm(n = 5, mean = 3)
rowWeightedMeans(mat, w = w[1:3])
colWeightedMeans(mat, w = w)

```

---

colWeightedMedians,dgCMatrix-method

*Calculates the weighted median for each row (column) of a matrix-like object*

---

### Description

Calculates the weighted median for each row (column) of a matrix-like object.

### Usage

```
## S4 method for signature 'dgCMatrix'
colWeightedMedians(
  x,
  w = NULL,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  useNames = TRUE
)

## S4 method for signature 'dgCMatrix'
rowWeightedMedians(
  x,
  w = NULL,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  useNames = TRUE
)
```

### Arguments

x	An $N \times K$ matrix-like object.
w	A <a href="#">numeric</a> vector of length $K$ ( $N$ ) that specifies by how much each element is weighted.
rows, cols	A <a href="#">vector</a> indicating the subset of rows (and/or columns) to operate over. If <a href="#">NULL</a> , no subsetting is done.
na.rm	If <a href="#">TRUE</a> , missing values ( <a href="#">NA</a> or <a href="#">NaN</a> ) are omitted from the calculations.
useNames	If <a href="#">TRUE</a> (default), names attributes of result are set. Else if <a href="#">FALSE</a> , no naming support is done.

### Details

The S4 methods for  $x$  of type [matrix](#), [array](#), [table](#), or [numeric](#) call `matrixStats::rowWeightedMedians` / `matrixStats::colWeightedMedians`.

### Value

Returns a [numeric vector](#) of length  $N$  ( $K$ ).

**See Also**

- `matrixStats::rowWeightedMedians()` and `matrixStats::colWeightedMedians()` which are used when the input is a matrix or numeric vector.
- See also [rowMedians](#) for the corresponding unweighted function.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)
w <- rnorm(n = 5, mean = 3)
rowWeightedMedians(mat, w = w[1:3])
colWeightedMedians(mat, w = w)
```

---

colWeightedSds,xgCMatrix-method

*Calculates the weighted standard deviation for each row (column) of a matrix-like object*

---

**Description**

Calculates the weighted standard deviation for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colWeightedSds(
  x,
  w = NULL,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  useNames = TRUE
)

## S4 method for signature 'xgCMatrix'
rowWeightedSds(
  x,
  w = NULL,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  useNames = TRUE
)
```

**Arguments**

x	An NxK matrix-like object.
w	A <b>numeric</b> vector of length K (N) that specifies by how much each element is weighted.
rows, cols	A <b>vector</b> indicating the subset of rows (and/or columns) to operate over. If <b>NULL</b> , no subsetting is done.
na.rm	If <b>TRUE</b> , missing values ( <b>NA</b> or <b>NaN</b> ) are omitted from the calculations.
useNames	If <b>TRUE</b> (default), names attributes of result are set. Else if <b>FALSE</b> , no naming support is done.

**Details**

The S4 methods for x of type **matrix**, **array**, **table**, or **numeric** call `matrixStats::rowWeightedSds` / `matrixStats::colWeightedSds`.

**Value**

Returns a **numeric vector** of length N (K).

**See Also**

- `matrixStats::rowWeightedSds()` and `matrixStats::colWeightedSds()` which are used when the input is a **matrix** or **numeric** vector.
- See also `rowSds` for the corresponding unweighted function.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
mat[4, 1] <- 0

print(mat)
w <- rnorm(n = 5, mean = 3)
rowWeightedSds(mat, w = w[1:3])
colWeightedSds(mat, w = w)
```

---

`colWeightedVars, xgCMatrix-method`

*Calculates the weighted variance for each row (column) of a matrix-like object*

---

**Description**

Calculates the weighted variance for each row (column) of a matrix-like object.

**Usage**

```
## S4 method for signature 'xgCMatrix'
colWeightedVars(
  x,
  w = NULL,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  useNames = TRUE
)

## S4 method for signature 'xgCMatrix'
rowWeightedVars(
  x,
  w = NULL,
  rows = NULL,
  cols = NULL,
  na.rm = FALSE,
  useNames = TRUE
)
```

**Arguments**

x	An NxK matrix-like object.
w	A <b>numeric</b> vector of length K (N) that specifies by how much each element is weighted.
rows, cols	A <b>vector</b> indicating the subset of rows (and/or columns) to operate over. If <b>NULL</b> , no subsetting is done.
na.rm	If <b>TRUE</b> , missing values (NA or NaN) are omitted from the calculations.
useNames	If <b>TRUE</b> (default), names attributes of result are set. Else if <b>FALSE</b> , no naming support is done.

**Details**

The S4 methods for x of type **matrix**, **array**, **table**, or **numeric** call `matrixStats::rowWeightedVars` / `matrixStats::colWeightedVars`.

**Value**

Returns a **numeric vector** of length N (K).

**See Also**

- `matrixStats::rowWeightedVars()` and `matrixStats::colWeightedVars()` which are used when the input is a matrix or numeric vector.
- See also **rowVars** for the corresponding unweighted function.

**Examples**

```
mat <- matrix(rnorm(15), nrow = 5, ncol = 3)
mat[2, 1] <- NA
mat[3, 3] <- Inf
```

```
mat[4, 1] <- 0

print(mat)
w <- rnorm(n = 5, mean = 3)
rowWeightedVars(mat, w = w[1:3])
colWeightedVars(mat, w = w)
```

---

xgCMatrix-class

*Union of double and logical column-sparse matrices*

---

### **Description**

Union of dgCMatrix and lgCMatrix

# Index

all, [3](#)  
any, [4](#), [6](#)  
array, [3](#), [4](#), [6](#), [8](#), [10–17](#), [19–23](#), [25–27](#), [29](#), [30](#),  
[34–37](#), [39](#), [40](#), [42–44](#), [46](#), [47](#)

base::rank, [32](#)

colAlls, [3](#)  
colAlls, xgCMatrix-method, [2](#)  
colAnyNAs, [4](#)  
colAnyNAs, xgCMatrix-method, [4](#)  
colAnys, [6](#)  
colAnys, xgCMatrix-method, [5](#)  
colAvgPerRowSet, [7](#)  
colAvgPerRowSet  
(colAvgPerRowSet, xgCMatrix-method),  
[6](#)  
colAvgPerRowSet, xgCMatrix-method, [6](#)  
colCollapse, [8](#)  
colCollapse, xgCMatrix-method, [8](#)  
colCounts, [10](#)  
colCounts, xgCMatrix-method, [9](#)  
colCummaxs, [11](#)  
colCummaxs, dgCMatrix-method, [10](#)  
colCummins, [12](#)  
colCummins, dgCMatrix-method, [11](#)  
colCumprods, [13](#)  
colCumprods, xgCMatrix-method, [12](#)  
colCumsums, [14](#)  
colCumsums, xgCMatrix-method, [13](#)  
colDiffs, [15](#)  
colDiffs, dgCMatrix-method, [14](#)  
colIQRDiffs, [16](#), [17](#)  
colIQRDiffs, dgCMatrix-method, [16](#)  
colIQRs, [17](#), [18](#)  
colIQRs, xgCMatrix-method, [17](#)  
colLogSumExps, [19](#)  
colLogSumExps, xgCMatrix-method, [18](#)  
colMadDiffs, [20](#)  
colMadDiffs, dgCMatrix-method, [19](#)  
colMads, [21](#), [22](#)  
colMads, dgCMatrix-method, [21](#)  
colMaxs, [22](#), [23](#)  
colMaxs, dgCMatrix-method, [22](#)

colMeans2, [23](#), [24](#)  
colMeans2, xgCMatrix-method, [23](#)  
colMedians, [25](#)  
colMedians, dgCMatrix-method, [24](#)  
colMins, [26](#)  
colMins, dgCMatrix-method, [25](#)  
colOrderStats, [27](#)  
colOrderStats, dgCMatrix-method, [26](#)  
colProds, [28](#)  
colProds, xgCMatrix-method, [27](#)  
colQuantiles, [29](#), [30](#)  
colQuantiles, xgCMatrix-method, [29](#)  
colRanges, [30](#), [31](#)  
colRanges, dgCMatrix-method, [30](#)  
colRanks, [32](#)  
colRanks, dgCMatrix-method, [31](#)  
colSdDiffs, [34](#)  
colSdDiffs, dgCMatrix-method, [33](#)  
colSds, [35](#)  
colSds, xgCMatrix-method, [34](#)  
colSums2, [36](#)  
colSums2, xgCMatrix-method, [36](#)  
colTabulates, [37](#)  
colTabulates, xgCMatrix-method, [37](#)  
colVarDiffs, [39](#)  
colVarDiffs, dgCMatrix-method, [38](#)  
colVars, [40](#)  
colVars, xgCMatrix-method, [39](#)  
colWeightedMads, [42](#)  
colWeightedMads, dgCMatrix-method, [41](#)  
colWeightedMeans, [43](#)  
colWeightedMeans, xgCMatrix-method, [42](#)  
colWeightedMedians, [44](#), [45](#)  
colWeightedMedians, dgCMatrix-method,  
[44](#)  
colWeightedSds, [46](#)  
colWeightedSds, xgCMatrix-method, [45](#)  
colWeightedVars, [47](#)  
colWeightedVars, xgCMatrix-method, [46](#)  
cummax, [11](#)  
cummin, [12](#)  
cumprod, [13](#)  
cumsum, [14](#)

- diff, [15](#)
- FALSE, [3–5](#), [8](#), [10–18](#), [20–25](#), [27–30](#), [32](#),  
[34–38](#), [40](#), [41](#), [43](#), [44](#), [46](#), [47](#)
- integer, [7](#), [10](#), [32](#)
- IQR, [18](#)
- is.na, [4](#)
- logical, [3](#), [4](#), [6](#)
- mad, [20](#), [21](#), [41](#)
- matrix, [3](#), [4](#), [6](#), [8](#), [10–17](#), [19–23](#), [25–27](#),  
[29–32](#), [34–37](#), [39](#), [40](#), [42–44](#), [46](#), [47](#)
- NA, [3](#), [5](#), [9](#), [16–18](#), [20–25](#), [28–30](#), [33](#), [35](#), [36](#),  
[38](#), [40](#), [41](#), [43](#), [44](#), [46](#), [47](#)
- NaN, [3](#), [5](#), [9](#), [16–18](#), [20–25](#), [28–30](#), [33](#), [35](#), [36](#),  
[38](#), [40](#), [41](#), [43](#), [44](#), [46](#), [47](#)
- NULL, [3–5](#), [7–9](#), [11–18](#), [20–25](#), [27–30](#), [32](#), [33](#),  
[35–38](#), [40](#), [41](#), [43](#), [44](#), [46](#), [47](#)
- numeric, [3](#), [4](#), [6](#), [8](#), [10–31](#), [34–37](#), [39–44](#), [46](#),  
[47](#)
- prod, [28](#)
- quantile, [29](#)
- range, [31](#)
- RangedSummarizedExperiment, [31](#)
- rowAlls, [3](#), [6](#), [10](#)
- rowAlls, xgCMatrix-method  
(colAlls, xgCMatrix-method), [2](#)
- rowAnyNAs, [4](#)
- rowAnyNAs, xgCMatrix-method  
(colAnyNAs, xgCMatrix-method), [4](#)
- rowAnys, [3](#), [4](#), [6](#), [10](#)
- rowAnys, xgCMatrix-method  
(colAnys, xgCMatrix-method), [5](#)
- rowAvgsPerColSet, [7](#)
- rowAvgsPerColSet, xgCMatrix-method  
(colAvgsPerRowSet, xgCMatrix-method),  
[6](#)
- rowCollapse, [8](#)
- rowCollapse, xgCMatrix-method  
(colCollapse, xgCMatrix-method),  
[8](#)
- rowCounts, [10](#)
- rowCounts, xgCMatrix-method  
(colCounts, xgCMatrix-method), [9](#)
- rowCummaxs, [11](#)
- rowCummaxs, dgCMatrix-method  
(colCummaxs, dgCMatrix-method),  
[10](#)
- rowCummins, [12](#)
- rowCummins, dgCMatrix-method  
(colCummins, dgCMatrix-method),  
[11](#)
- rowCumprods, [13](#)
- rowCumprods, xgCMatrix-method  
(colCumprods, xgCMatrix-method),  
[12](#)
- rowCumsums, [14](#)
- rowCumsums, xgCMatrix-method  
(colCumsums, xgCMatrix-method),  
[13](#)
- rowDiffs, [15](#)
- rowDiffs, dgCMatrix-method  
(colDiffs, dgCMatrix-method), [14](#)
- rowIQRDiffs, [16](#), [17](#)
- rowIQRDiffs, dgCMatrix-method  
(colIQRDiffs, dgCMatrix-method),  
[16](#)
- rowIQRs, [17](#), [18](#)
- rowIQRs, xgCMatrix-method  
(colIQRs, xgCMatrix-method), [17](#)
- rowLogSumExps, [19](#)
- rowLogSumExps, xgCMatrix-method  
(colLogSumExps, xgCMatrix-method),  
[18](#)
- rowMadDiffs, [20](#)
- rowMadDiffs, dgCMatrix-method  
(colMadDiffs, dgCMatrix-method),  
[19](#)
- rowMads, [21](#), [22](#), [42](#)
- rowMads(), [18](#)
- rowMads, dgCMatrix-method  
(colMads, dgCMatrix-method), [21](#)
- rowMaxs, [11](#), [22](#), [23](#), [26](#), [31](#)
- rowMaxs, dgCMatrix-method  
(colMaxs, dgCMatrix-method), [22](#)
- rowMeans, [22](#), [24](#), [25](#), [35](#), [36](#), [40](#)
- rowMeans2, [22–25](#), [35](#), [36](#), [40](#), [43](#)
- rowMeans2, xgCMatrix-method  
(colMeans2, xgCMatrix-method),  
[23](#)
- rowMedians, [25](#), [45](#)
- rowMedians, dgCMatrix-method  
(colMedians, dgCMatrix-method),  
[24](#)
- rowMins, [12](#), [23](#), [26](#), [31](#)
- rowMins, dgCMatrix-method  
(colMins, dgCMatrix-method), [25](#)
- rowOrderStats, [27](#)
- rowOrderStats, dgCMatrix-method  
(colOrderStats, dgCMatrix-method),

- 26
- rowProds, 28
- rowProds, xgCMatrix-method
  - (colProds, xgCMatrix-method), 27
- rowQuantiles, 29, 30
- rowQuantiles, xgCMatrix-method
  - (colQuantiles, xgCMatrix-method), 29
- rowRanges, 30, 31
- rowRanges, dgCMatrix-method
  - (colRanges, dgCMatrix-method), 30
- rowRanks, 32
- rowRanks, dgCMatrix-method
  - (colRanks, dgCMatrix-method), 31
- rowSdDiffs, 34
- rowSdDiffs, dgCMatrix-method
  - (colSdDiffs, dgCMatrix-method), 33
- rowSds, 18, 22, 35, 40, 46
- rowSds(), 34
- rowSds, xgCMatrix-method
  - (colSds, xgCMatrix-method), 34
- rowSums2, 36
- rowSums2(), 19
- rowSums2, xgCMatrix-method
  - (colSums2, xgCMatrix-method), 36
- rowTabulates, 37
- rowTabulates, xgCMatrix-method
  - (colTabulates, xgCMatrix-method), 37
- rowVarDiffs, 39
- rowVarDiffs, dgCMatrix-method
  - (colVarDiffs, dgCMatrix-method), 38
- rowVars, 24, 35, 40, 47
- rowVars(), 39
- rowVars, xgCMatrix-method
  - (colVars, xgCMatrix-method), 39
- rowWeightedMads, 42
- rowWeightedMads, dgCMatrix-method
  - (colWeightedMads, dgCMatrix-method), 41
- rowWeightedMeans, 43
- rowWeightedMeans, xgCMatrix-method
  - (colWeightedMeans, xgCMatrix-method), 42
- rowWeightedMedians, 44, 45
- rowWeightedMedians, dgCMatrix-method
  - (colWeightedMedians, dgCMatrix-method), 44
- rowWeightedSds, 46
- rowWeightedSds, xgCMatrix-method
  - (colWeightedSds, xgCMatrix-method), 45
- rowWeightedVars, 47
- rowWeightedVars, xgCMatrix-method
  - (colWeightedVars, xgCMatrix-method), 46
- stats::quantile, 30
- sum, 36
- table, 3, 4, 6, 8, 10–17, 19–23, 25–27, 29, 30, 34–37, 39, 40, 42–44, 46, 47
- TRUE, 3–5, 8–18, 20–25, 27–30, 32–38, 40, 41, 43, 44, 46, 47
- var, 40
- vector, 3–30, 32–44, 46, 47
- xgCMatrix-class, 48