Package 'scUtils'

October 14, 2022

Type Package

Title Utility Functions for Single-Cell RNA Sequencing Data

Version 0.1.0

Description Analysis of single-cell RNA sequencing data can be simple and clear with the right utility functions. This package collects such functions, aiming to fulfill the following criteria:
code clarity over performance (i.e. plain R code instead of C code),
most important analysis steps over completeness
(analysis 'by hand', not automated integration etc.),
emphasis on quantitative visualization (intensity-coded color scale, etc.).

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Imports ggplot2, Matrix, scales, assertthat, dplyr, viridis, viridisLite, methods

Suggests testthat, tibble

NeedsCompilation no

Author Felix Frauhammer [aut, cre], Simon Anders [ctb] (Simon Anders wrote the colVars_spm function.)

Maintainer Felix Frauhammer <felixwertek@gmail.com>

Repository CRAN

Date/Publication 2020-06-25 16:20:02 UTC

R topics documented:

`closed_breaks_log2` *Closed breaks for log scale*

Description

Finds breaks that are powers of 2, and forces inclusion of upper and lower limits (displaying the closed interval). Including limits specifically is particularly useful for ggplot2's color/fill, as it emphasizes the meaning of maximal/minimal color intensities (see examples).

Usage

```
closed_breaks_log2(lims)
```

Arguments

`lims`	Vector with lower and upper limits (in that order) of the data that you want breaks for.

Details

The `feat` function uses `closed_breaks_log2` to color by gene expression, where the maximal expression gives valuable intuition for a gene's overall expression strength. For x- or y-axis (`scale_*_log10`), I still recommend `breaks_log` from the scales package.

Value

Numeric vector with breaks.

See Also

`closed_labels`

Examples

```
# closed breaks include maximum, breaks_log do not:
closed_breaks_log2(lims = c(.01, 977.1))
scales::breaks_log()(c(.01, 977.1))
```

`closed_labels` *Human-readable labels for closed breaks*

Description

Complements the closed_breaks_log2 function.

Usage

```
closed_labels(x, min_is_zero = FALSE)
```

Arguments

`x`	Vector of breaks for which to produce labels. Typically, this is the output of `closed_breaks_log2`.
`min_is_zero`	Should the smallest break be displayed as zero (TRUE) or as the actual value (FALSE). Default: FALSE

Details

This is a helper for the `feat` function. `feat` replaces numeric zeros with the next-smallest expression value to avoid taking the logarithm of zero. `min_is_zero` can be used to display the lowest break of the color scale as zero in these cases.

Value

Character vector with labels, used by `feat` function.

See Also

`label_scientific` `label_number_auto`

Examples

```
# human readable output:
closed_labels(c(.001111,.122, 0.5, 10, 100, 1800))
```

colVars_spm *Variance computation for sparse matrices*

Description

Compute variance for each column / each row of a dgCMatrix (from Matrix package).

Usage

```
colVars_spm(spm)

rowVars_spm(spm)
```

Arguments

spm	A sparse matrix of class dgCMatrix from the Matrix package.

Details

The only supported format currently is dgCMatrix. While the Matrix package has other formats, this one is used for scRNAseq raw count data. Function code written by Simon Anders.

Value

Vector with variances.

See Also

`vignette("Intro2Matrix", package="Matrix")` CsparseMatrix-class

Examples

```
library(Matrix)
 mat <- as(matrix(rpois(900,1), ncol=3), "dgCMatrix")
 colVars_spm(mat)
```

feat *Feature Plot*

Description

Highlight gene expression data in a 2D-embedding (UMAP, tSNE, etc.).

Usage

```
feat(embedding, expression, legend_name = "Expression")
```

Arguments

`embedding`	A matrix/data.frame/tibble/... with exactly two columns. If colnames are missing, the axis will be named "Dim1" and "Dim2". Other classes than matrix/data.frame/tibble are possible, as long as `data.frame(embedding)` produces a numeric data.frame.
`expression`	Numeric vector with expression values of the gene of interest. Order has to correspond to the row order in `embedding`. Typically, `expression` is normalized gene expression and we recommend `k/s/mean(1/s)`, where `k` are UMI counts for the gene of interest and `s` are totalUMI of the cell (aka library size).
`legend_name`	Text displayed above the legend. Most commonly the name of the displayed gene.

Details

This function discourages customization on purpose, because it bundles geoms, themes and settings that I found important for visualizing gene expression in scRNAseq data:

- coord_fixed, to avoid distortion of embeddings
- geom_point with size=.4, to ameliorate overplotting
- No background grid, because distances and axis units in embeddings do not carry meaning for most dimensionality reduction techniques.
- Intensity-coded color scales (viridis) displayed with log2-transformation. Makes visualization independent of colorblindness and appropriate for gene expression data (which is usually Log Normal distributed).
- Color scale breaks are displayed as 'closed interval', i.e. `max(expression)` and `min(expression)` are the most extreme breaks. Rounding makes them human-readable. This functionality is provided by closed_breaks_log2 and closed_labels.

If you insist on customizing, think of this function as a great starting point, you can simply copy-paste the code after typing `feat` into your console.

Value

A `ggplot2` object storing a colored scatter plot.

See Also

`ggplot`, `closed_labels`, `closed_breaks_log2`

Examples

```
# expression goes from 0 to 22:
set.seed(100)
feat(matrix(rnorm(2000, c(.1, 3)), ncol=2), rpois(1000, c(.1, 11)))
# expression goes from 2 to 52:
set.seed(100)
feat(matrix(rnorm(2000, c(.1, 3)), ncol=2), rpois(1000, c(10, 31)))
```

`is_wholenumber`	*Check if number(s) is/are integers. In contrast to is.integer, is_wholenumber does not check the class but accepts all numbers that are integers with reasonable precision.*

Description

Check if number(s) is/are integers. In contrast to is.integer, is_wholenumber does not check the class but accepts all numbers that are integers with reasonable precision.

Usage

```
is_wholenumber(x, tol = .Machine$double.eps^0.5)
```

Arguments

`x`	Number to test
`tol`	tolerance for testing

Index