Package 'Cepo'

March 31, 2025

Title Cepo for the identification of differentially stable genes

Version 1.12.0

Description
Defining the identity of a cell is fundamental to understand the heterogeneity of cells to various environmental signals and perturbations. We present Cepo, a new method to explore cell identities from single-cell RNA-sequencing data using differential stability as a new metric to define cell identity genes. Cepo computes cell-type specific gene statistics pertaining to differential stable gene expression.

License MIT + file LICENSE

Encoding UTF-8

LazyData false

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Imports DelayedMatrixStats, DelayedArray, HDF5Array, S4Vectors, methods, SingleCellExperiment, SummarizedExperiment, ggplot2, rlang, grDevices, patchwork, reshape2, BiocParallel, stats, dplyr, purrr

biocViews Classification, GeneExpression, SingleCell, Software, Sequencing, DifferentialExpression

Suggests knitr, rmarkdown, BiocStyle, testthat, covr, UpSetR, scater, scMerge, fgsea, escape, pheatmap

VignetteBuilder knitr

Depends GSEABase, R (>= 4.1)

git_url https://git.bioconductor.org/packages/Cepo

git_branch RELEASE_3_20

git_last_commit d36a16b

git_last_commit_date 2024-10-29

Repository Bioconductor 3.20

Date/Publication 2025-03-31

Author Hani Jieun Kim [aut, cre] (<https://orcid.org/0000-0003-1844-3275>), Kevin Wang [aut] (<https://orcid.org/0000-0003-2615-6102>)

Maintainer Hani Jieun Kim <hani.kim127@gmail.com>

Contents

`cellbench` *cellbench*

Description

A single-cell RNA-seq dataset adapted from sc_mixology

Usage

```
data(cellbench)
```

Format

An object of SingleCellExperiment class with 895 cells and 2001 genes.

Source

https://github.com/LuyiTian/sc_mixology

`Cepo` *Computing Cepo cell identity genes*

Description

ExprsMat accepts various matrix objects, including DelayedArray and HDF5Array for out-of-memory computations. See vignette.

Usage

```
Cepo(
  exprsMat,
  cellTypes,
  minCells = 20,
  minCelltype = 3,
  exprsPct = 0.05,
  prefilter_sd = NULL,
  prefilter_pzero = NULL,
  logfc = NULL,
  computePvalue = NULL,
  computeFastPvalue = TRUE,
```

```
  variability = "CV",
  method = "weightedMean",
  weight = c(0.5, 0.5),
  workers = 1L,
  block = NULL,
  ...
)
```

Arguments

exprsMat	Expression matrix where columns denote cells and rows denote genes
cellTypes	Vector of cell type labels
minCells	Integer indicating the minimum number of cells required within a cell type
minCelltype	Integer indicating the minimum number of cell types required in each batch
exprsPct	Percentage of lowly expressed genes to remove. Default to NULL to not remove any genes.
prefilter_sd	Numeric value indicating threshold relating to standard deviation of genes. Used with prefilter_zeros.
prefilter_pzero	Numeric value indicating threshold relating to the percentage of zero expression of genes. Used with prefilter_sd.
logfc	Numeric value indicating the threshold of log fold-change to use to filter genes.
computePvalue	Whether to compute p-values using bootstrap test. Default to NULL to not make computations. Set this to an integer to set the number of bootstraps needed (recommend to be at least 100).
computeFastPvalue	Logical vector indicating whether to perform a faster version of p-value calculation. Set to TRUE by default.
variability	A character indicating the stability measure (CV, IQR, MAD, SD). Default is set to CV.
method	Character indicating the method for integration the two stability measures. By default this is set to 'weightedMean' with equal weights.
weight	Vector of two values indicating the weights for each stability measure. By default this value is c(0.5, 0.5).
workers	Number of cores to use. Default to 1, which invokes `BiocParallel::SerialParam`. For workers greater than 1, see the `workers` argument in `BiocParallel::MulticoreParam` and `BiocParallel::SnowParam`.
block	Vector of batch labels
...	Additional arguments passed to `BiocParallel::MulticoreParam` and `BiocParallel::SnowParam`.

Value

Returns a list of key genes.

Examples

```
library(SingleCellExperiment)
data('cellbench', package = 'Cepo')
cellbench
```

```
cepoOutput <- Cepo(logcounts(cellbench), cellbench$celltype)
cepoOutput
```

`plotDensities` *Plot densities*

Description

Plot densities

Usage

```
plotDensities(
  x,
  cepoOutput,
  nGenes = 2,
  assay = "logcounts",
  celltypeColumn,
  celltype = NULL,
  genes = NULL,
  plotType = c("histogram", "density"),
  color = NULL
)
```

Arguments

`x`	a `SummarizedExperiment` or a `SingleCellExperiment` object.
`cepoOutput`	an output from Cepo or doLimma/doVoom/doTtest/doWilcoxon functions
`nGenes`	number of top genes from each celltype to plot. Default to 2.
`assay`	a character ('logcounts' by default), indicating the name of the assays(x) element which stores the expression data (i.e., assays(x)$name_assays_expression). We strongly encourage using normalized data, such as counts per million (CPM) or log-CPM.
`celltypeColumn`	a character, indicating the name of the name of the cell type column in the colData(x).
`celltype`	a character, indicating the name of the cell type to plot. Default is NULL which selects all celltypes in the cepoOutput.
`genes`	a character vector, indicating the name of the genes to plot. Default to NULL, so that 2 top genes from each celltype will be plotted.
`plotType`	Either 'histogram' or 'density'
`color`	a named color vector. The names should correspond to the `celltype` argument above

Value

A `ggplot` object with cell-type specific densities for a gene.

A `ggplot` object.

Examples

```
library(SingleCellExperiment)
data('cellbench', package = 'Cepo')
cellbench
cepoOutput <- Cepo(logcounts(cellbench), cellbench$celltype)

plotDensities(
  x = cellbench,
  cepoOutput = cepoOutput,
  assay = 'logcounts',
  plotType = 'histogram',
  celltypeColumn = 'celltype'
)

plotDensities(
  x = cellbench,
  cepoOutput = cepoOutput,
  genes = c('PLTP', 'CPT1C', 'MEG3', 'SYCE1', 'MICOS10P3', 'HOXB7'),
  assay = 'logcounts',
  plotType = 'histogram',
  celltypeColumn = 'celltype'
)
```

---

sce_pancreas *sce_pancreas*

Description

A subsampled single-cell RNA-seq dataset

Usage

```
data(sce_pancreas)
```

Format

An object of SingleCellExperiment class with 528 cells and 1358 genes.

---

setCepoBPPARAM *Setting parallel params based on operating platform*

Description

Setting parallel params based on operating platform

Usage

```
setCepoBPPARAM(workers = 1L, ...)
```

Arguments

workers	Number of cores to use. Default to 1, which invokes BiocParallel::SerialParam. For workers greater than 1, see the workers argument in BiocParallel::MulticoreParam and BiocParallel::SnowParam.
...	Additional arguments passed to BiocParallel::MulticoreParam and BiocParallel::SnowParam.

Value

Parameters for parallel computing depending on OS

Examples

```
# system.time(BiocParallel::bplapply(1:3, FUN = function(i){Sys.sleep(i)},
# BPPARAM = setCepoBPPARAM(workers = 1)))
# system.time(BiocParallel::bplapply(1:3, FUN = function(i){Sys.sleep(i)},
# BPPARAM = setCepoBPPARAM(workers = 3)))
```

---

topGenes *Extract the top genes from the Cepo output*

Description

Extract the top genes from the Cepo output

Usage

```
topGenes(object, n = 5, returnValues = FALSE)
```

Arguments

object	Output from the Cepo function
n	Number of top genes to extract
returnValues	Whether to return the numeric value associated with the top selected genes

Value

Returns a list of key genes.

Examples

```
set.seed(1234)
n <- 50 ## genes, rows
p <- 100 ## cells, cols
exprsMat <- matrix(rpois(n * p, lambda = 5), nrow = n)
rownames(exprsMat) <- paste0('gene', 1:n)
colnames(exprsMat) <- paste0('cell', 1:p)
cellTypes <- sample(letters[1:3], size = p, replace = TRUE)
cepo_output <- Cepo(exprsMat = exprsMat, cellTypes = cellTypes)
cepo_output
topGenes(cepo_output, n = 2)
topGenes(cepo_output, n = 2, returnValues = TRUE)
```

Index