

# Package ‘Cepo’

June 4, 2026

**Title** Cepo for the identification of differentially stable genes

**Version** 1.19.0

## 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** devel

**git\_last\_commit** 5f662d1

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

**Repository** Bioconductor 3.24

**Date/Publication** 2026-06-04

**Author** Hani Jieun Kim [aut, cre] (ORCID:

<<https://orcid.org/0000-0003-1844-3275>>),

Kevin Wang [aut] (ORCID: <<https://orcid.org/0000-0003-2615-6102>>)

**Maintainer** Hani Jieun Kim <[hani.kim127@gmail.com](mailto:hani.kim127@gmail.com)>

## Contents

cellbench . . . . .	2
Cepo . . . . .	2
plotDensities . . . . .	4
sce_pancreas . . . . .	5
setCepoBPPARAM . . . . .	5
topGenes . . . . .	6
<b>Index</b>	<b>7</b>

---

cellbench	<i>cellbench</i>
-----------	------------------

---

### 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](https://github.com/LuyiTian/sc_mixology)

---

Cepo	<i>Computing Cepo cell identity genes</i>
------	---

---

### 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

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

## 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 <a href="#">SummarizedExperiment</a> or a <a href="#">SingleCellExperiment</a> 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 col-Data(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)))
```

---

<code>topGenes</code>	<i>Extract the top genes from the Cepo output</i>
-----------------------	---

---

**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

## \* datasets

- cellbench, [2](#)
- sce\_pancreas, [5](#)

cellbench, [2](#)

Cepo, [2](#)

ggplot, [4](#)

plotDensities, [4](#)

sce\_pancreas, [5](#)

setCepoBPPARAM, [5](#)

SingleCellExperiment, [4](#)

SummarizedExperiment, [4](#)

topGenes, [6](#)