

# Package ‘BASiCS’

April 11, 2018

**Type** Package

**Title** Bayesian Analysis of Single-Cell Sequencing data

**Version** 1.0.1

**Date** : 2018-03-18

**Maintainer** Catalina A. Vallejos <cnvallej@uc.cl>

**Description** Single-cell mRNA sequencing can uncover novel cell-to-cell heterogeneity in gene expression levels in seemingly homogeneous populations of cells. However, these experiments are prone to high levels of technical noise, creating new challenges for identifying genes that show genuine heterogeneous expression within the population of cells under study. BASiCS (Bayesian Analysis of Single-Cell Sequencing data) is an integrated Bayesian hierarchical model to perform statistical analyses of single-cell RNA sequencing datasets in the context of supervised experiments (where the groups of cells of interest are known a priori, e.g. experimental conditions or cell types). BASiCS performs built-in data normalisation (global scaling) and technical noise quantification (based on spike-in genes). BASiCS provides an intuitive detection criterion for highly (or lowly) variable genes within a single group of cells. Additionally, BASiCS can compare gene expression patterns between two or more pre-specified groups of cells. Unlike traditional differential expression tools, BASiCS quantifies changes in expression that lie beyond comparisons of means, also allowing the study of changes in cell-to-cell heterogeneity. The latter are quantified via a biological over-dispersion parameter that measures residual over-dispersion (with respect to Poisson sampling) after normalisation and technical noise removal.

**License** GPL (>= 2)

**Depends** R (>= 3.4), SingleCellExperiment

**Imports** SummarizedExperiment, S4Vectors, BiocGenerics, Rcpp (>= 0.11.3), methods, coda, scran, testthat, data.table, matrixStats, graphics, KernSmooth, grDevices, stats, utils

**Suggests** knitr, BiocStyle

**LinkingTo** Rcpp, RcppArmadillo

**VignetteBuilder** knitr

**biocViews** Normalization, Sequencing, RNASeq, Software, GeneExpression, Transcriptomics, SingleCell, DifferentialExpression, Bayesian, CellBiology

**URL** <https://github.com/catavallejos/BASiCS>

**BugReports** <https://github.com/catavallejos/BASiCS/issues>

**RoxygenNote** 6.0.1

**NeedsCompilation** yes

**Author** Catalina Vallejos [aut, cre],  
Nils Eling [aut],  
Sylvia Richardson [ctb],  
John Marioni [ctb]

## R topics documented:

|                                            |           |
|--------------------------------------------|-----------|
| BASiCS_Chain                               | 2         |
| BASiCS_Chain-methods                       | 3         |
| BASiCS_DenoisedCounts                      | 4         |
| BASiCS_DenoisedRates                       | 5         |
| BASiCS_DetectHVG                           | 6         |
| BASiCS_D_TestDE                            | 8         |
| BASiCS_Filter                              | 8         |
| BASiCS_LoadChain                           | 10        |
| BASiCS_MCMC                                | 11        |
| BASiCS_Sim                                 | 14        |
| BASiCS_Summary                             | 15        |
| BASiCS_Summary-methods                     | 16        |
| BASiCS_TestDE                              | 17        |
| BASiCS_VarianceDecomp                      | 19        |
| BASiCS_VarThresholdSearchHVG               | 21        |
| ChainRNA                                   | 22        |
| ChainSC                                    | 22        |
| displayChainBASiCS-BASiCS_Chain-method     | 23        |
| displaySummaryBASiCS-BASiCS_Summary-method | 24        |
| makeExampleBASiCS_Data                     | 25        |
| newBASiCS_Chain                            | 26        |
| newBASiCS_Data                             | 27        |
| plot-BASiCS_Chain-method                   | 28        |
| plot-BASiCS_Summary-method                 | 29        |
| Summary                                    | 30        |
| <b>Index</b>                               | <b>32</b> |

---

BASiCS\_Chain

*The BASiCS\_Chain class*

---

### Description

Container of an MCMC sample of the BASiCS' model parameters (see Vallejos et al, 2015) as generated by the function `BASiCS_MCMC`.

**Slots**

parameters List of matrices containing MCMC chains for each model parameter.

- mu** MCMC chain for gene-specific mean expression parameters  $\mu_i$ , biological genes only (matrix with q.bio columns, all elements must be positive numbers)
- delta** MCMC chain for gene-specific biological over-dispersion parameters  $\delta_i$ , biological genes only (matrix with q.bio columns, all elements must be positive numbers)
- phi** MCMC chain for cell-specific mRNA content normalisation parameters  $\phi_j$  (matrix with n columns, all elements must be positive numbers and the sum of its elements must be equal to n)
- s** MCMC chain for cell-specific technical normalisation parameters  $s_j$  (matrix with n columns, all elements must be positive numbers)
- nu** MCMC chain for cell-specific random effects  $\nu_j$  (matrix with n columns, all elements must be positive numbers)
- theta** MCMC chain for technical over-dispersion parameter(s)  $\theta$  (matrix, all elements must be positive, each column represents 1 batch)

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**Examples**

```
# A BASiCS_Chain object created by the BASiCS_MCMC function.
Data <- makeExampleBASiCS_Data()
Chain <- BASiCS_MCMC(Data, N = 100, Thin = 2, Burn = 2)
```

---

BASiCS\_Chain-methods *'show' method for BASiCS\_Chain objects*

---

**Description**

'show' method for [BASiCS\\_Chain](#) objects.

'updateObject' method for [BASiCS\\_Chain](#) objects. It is used to convert outdated [BASiCS\\_Chain](#) objects into a version that is compatible with the Bioconductor release of BASiCS. Do not use this method if [BASiCS\\_Chain](#) already contains a parameters slot.

**Usage**

```
## S4 method for signature 'BASiCS_Chain'
show(object)

## S4 method for signature 'BASiCS_Chain'
updateObject(object, ..., verbose = FALSE)
```

**Arguments**

|         |                                                                                              |
|---------|----------------------------------------------------------------------------------------------|
| object  | A <a href="#">BASiCS_Chain</a> object.                                                       |
| ...     | Additional arguments of <a href="#">updateObject</a> generic method. Not used within BASiCS. |
| verbose | Additional argument of <a href="#">updateObject</a> generic method. Not used within BASiCS.  |

**Value**

Prints a summary of the properties of object.

Returns an updated [BASiCS\\_Chain](#) object that contains all model parameters in a single slot object (list).

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**Examples**

```
Data <- makeExampleBASiCS_Data()
Chain <- BASiCS_MCMC(Data, N = 50, Thin = 2, Burn = 2)

# Not run
# New_Chain <- updateObject(Old_Chain)
```

---

`BASiCS_DenoisedCounts` *Calculates denoised expression counts*

---

**Description**

Calculates denoised expression counts by removing cell-specific technical variation.

**Usage**

```
BASiCS_DenoisedCounts(Data, Chain)
```

**Arguments**

|       |                                                         |
|-------|---------------------------------------------------------|
| Data  | an object of class <a href="#">SingleCellExperiment</a> |
| Chain | an object of class <a href="#">BASiCS_Chain</a>         |

**Details**

See vignette

**Value**

A matrix of denoised expression counts. In line with global scaling normalisation strategies, these are defined as  $X_{ij}/(\phi_j\nu_j)$  for biological genes and  $X_{ij}/(\nu_j)$  for spike-in genes. For this calculation  $\phi_j$   $\nu_j$  are estimated by their corresponding posterior medians.

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**See Also**

[BASiCS\\_Chain](#)

**Examples**

```
Data <- makeExampleBASiCS_Data(WithSpikes = TRUE)
Chain <- BASiCS_MCMC(Data, N = 10000, Thin = 10, Burn = 5000,
  PrintProgress = FALSE)

DC <- BASiCS_DenoisedCounts(Data, Chain)
```

---

BASiCS\_DenoisedRates *Calculates denoised expression rates*

---

**Description**

Calculates normalised and denoised expression rates, by removing the effect of technical variation.

**Usage**

```
BASiCS_DenoisedRates(Data, Chain, Propensities = FALSE)
```

**Arguments**

|              |                                                                                                                                                           |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Data         | an object of class <a href="#">SingleCellExperiment</a>                                                                                                   |
| Chain        | an object of class <a href="#">BASiCS_Chain</a>                                                                                                           |
| Propensities | If TRUE, returns underlying expression propensities $\rho_{ij}$ . Otherwise, denoised rates $\mu_i\rho_{ij}$ are returned. Default: Propensities = FALSE. |

**Details**

See vignette

**Value**

A matrix of denoised expression rates (biological genes only)

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**See Also**

[BASiCS\\_Chain](#)

**Examples**

```
Data <- makeExampleBASiCS_Data(WithSpikes = TRUE)
Chain <- BASiCS_MCMC(Data, N = 10000, Thin = 10, Burn = 5000,
  PrintProgress = FALSE)

DR <- BASiCS_DenoisedRates(Data, Chain)
```

---

BASiCS\_DetectHVG

*Detection method for highly and lowly variable genes*

---

**Description**

Functions to detect highly and lowly variable genes

**Usage**

```
BASiCS_DetectHVG(Chain, VarThreshold, ProbThreshold = NULL, EFDR = 0.1,
  OrderVariable = "Prob", Plot = FALSE, ...)
```

```
BASiCS_DetectLVG(Chain, VarThreshold, ProbThreshold = NULL, EFDR = 0.1,
  OrderVariable = "Prob", Plot = FALSE, ...)
```

**Arguments**

|               |                                                                                                 |
|---------------|-------------------------------------------------------------------------------------------------|
| Chain         | an object of class <a href="#">BASiCS_Chain</a>                                                 |
| VarThreshold  | Variance contribution threshold (must be a positive value, between 0 and 1)                     |
| ProbThreshold | Optional parameter. Posterior probability threshold (must be a positive value, between 0 and 1) |
| EFDR          | Target for expected false discovery rate related to HVG/LVG detection (default = 0.10)          |
| OrderVariable | Ordering variable for output. Possible values: 'GeneIndex', 'Mu', 'Delta', 'Sigma' and 'Prob'.  |
| Plot          | If Plot = TRUE error control and expression versus HVG/LVG probability plots are generated      |
| ...           | Graphical parameters (see <a href="#">par</a> ).                                                |

**Details**

See vignette

**Value**

BASiCS\_DetectHVG returns a list of 4 elements:

Table Matrix whose columns contain

GeneIndex Vector of length  $q.bio$ . Gene index as in the order present in the analysed [SingleCellExperiment](#)

GeneName Vector of length  $q.bio$ . Gene name as in the order present in the analysed [SingleCellExperiment](#)

Mu Vector of length  $q.bio$ . For each biological gene, posterior median of gene-specific mean expression parameters  $\mu_i$

Delta Vector of length  $q.bio$ . For each biological gene, posterior median of gene-specific biological over-dispersion parameter  $\delta_i$

Sigma Vector of length  $q.bio$ . For each biological gene, proportion of the total variability that is due to a biological heterogeneity component.

Prob Vector of length  $q.bio$ . For each biological gene, probability of being highly variable according to the given thresholds.

HVG Vector of length  $q.bio$ . For each biological gene, indicator of being detected as highly variable according to the given thresholds.

ProbThreshold Posterior probability threshold.

EFDR Expected false discovery rate for the given thresholds.

EFNR Expected false negative rate for the given thresholds.

BASiCS\_DetectLVG produces a similar output, replacing the column HVG by LVG, an indicator of a gene being detected as lowly variable according to the given thresholds.

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**See Also**

[BASiCS\\_Chain](#)

**Examples**

```
# See  
help(BASiCS_MCMC)
```

---

BASiCS\_D\_TestDE      *Detection of genes with changes in expression.*

---

### Description

This function is no longer in use and will be removed from future releases of BASiCS. Please use [BASiCS\\_TestDE](#) instead.

### Usage

```
BASiCS_D_TestDE(...)
```

### Arguments

...      Optional parameters.

### Details

This function is no longer in use and will be removed from future releases of BASiCS.

### Value

This function is no longer in use and will be removed from future releases of BASiCS. Please use [BASiCS\\_TestDE](#) instead.

### Author(s)

Catalina A. Vallejos <cnvallej@uc.cl>

### Examples

```
# Use 'BASiCS_TestDE' instead
help(BASiCS_TestDE)
```

---

BASiCS\_Filter      *Filter for input datasets*

---

### Description

BASiCS\_Filter indicates which transcripts and cells pass a pre-defined inclusion criteria. The output of this function can be combined with newBASiCS\_Data to generate a the [SingleCellExperiment](#) object required to run BASiCS.

### Usage

```
BASiCS_Filter(Counts, Tech, SpikeInput, BatchInfo = NULL,
  MinTotalCountsPerCell = 2, MinTotalCountsPerGene = 2,
  MinCellsWithExpression = 2, MinAvCountsPerCellsWithExpression = 2)
```



**Arguments**

|                                   |                                                                                                                                                                                                                        |
|-----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Counts                            | Matrix of dimensions $q$ times $n$ whose elements corresponds to the simulated expression counts. First $q$ .bio rows correspond to biological genes. Last $q$ - $q$ .bio rows correspond to technical spike-in genes. |
| Tech                              | Logical vector of length $q$ . If <code>Tech = FALSE</code> the gene is biological; otherwise the gene is spike-in.                                                                                                    |
| SpikeInput                        | Vector of length $q$ - $q$ .bio whose elements indicate the simulated input concentrations for the spike-in genes.                                                                                                     |
| BatchInfo                         | Vector of length $n$ whose elements indicate batch information. Not required if a single batch is present on the data. Default: <code>BatchInfo = NULL</code> .                                                        |
| MinTotalCountsPerCell             | Minimum value of total expression counts required per cell (biological and technical). Default: <code>MinTotalCountsPerCell = 2</code> .                                                                               |
| MinTotalCountsPerGene             | Minimum value of total expression counts required per transcript (biological and technical). Default: <code>MinTotalCountsPerGene = 2</code> .                                                                         |
| MinCellsWithExpression            | Minimum number of cells where expression must be detected (positive count). Criteria applied to each transcript. Default: <code>MinCellsWithExpression = 2</code> .                                                    |
| MinAvCountsPerCellsWithExpression | Minimum average number of counts per cells where expression is detected. Criteria applied to each transcript. Default value: <code>MinAvCountsPerCellsWithExpression = 2</code> .                                      |

**Value**

A list of 2 elements

Counts Filtered matrix of expression counts

Tech Filtered vector of spike-in indicators

SpikeInput Filtered vector of spike-in genes input molecules

BatchInfo Filtered vector of the 'BatchInfo' argument

IncludeGenes Inclusion indicators for transcripts

IncludeCells Inclusion indicators for cells

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

Vallejos, Richardson and Marioni (2016). Genome Biology.

**Examples**

```
set.seed(1)
Counts <- matrix(rpois(50*10, 2), ncol = 10)
rownames(Counts) <- c(paste0('Gene', 1:40), paste0('Spike', 1:10))
Tech <- c(rep(FALSE, 40), rep(TRUE, 10))
```

```

set.seed(2)
SpikeInput <- rgamma(10,1,1)
SpikeInfo <- data.frame('SpikeID' = paste0('Spike', 1:10),
                       'SpikeInput' = SpikeInput)

Filter = BASiCS_Filter(Counts, Tech, SpikeInput,
                      MinTotalCountsPerCell = 2,
                      MinTotalCountsPerGene = 2,
                      MinCellsWithExpression = 2,
                      MinAvCountsPerCellsWithExpression = 2)
SpikeInfoFilter <- SpikeInfo[SpikeInfo$SpikeID %in% rownames(Filter$Counts),]
FilterData <- newBASiCS_Data(Filter$Counts, Filter$Tech, SpikeInfoFilter)

```

---

|                  |                                                                                             |
|------------------|---------------------------------------------------------------------------------------------|
| BASiCS_LoadChain | <i>Loads pre-computed MCMC chains generated by the <a href="#">BASiCS_MCMC</a> function</i> |
|------------------|---------------------------------------------------------------------------------------------|

---

## Description

Loads pre-computed MCMC chains generated by the [BASiCS\\_MCMC](#) function, creating a [BASiCS\\_Chain](#) object

## Usage

```
BASiCS_LoadChain(RunName, StoreDir = getwd(), StoreUpdatedChain = FALSE)
```

## Arguments

|                   |                                                                                                                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RunName           | String used to index ‘.Rds’ file containing the MCMC chain (produced by the <a href="#">BASiCS_MCMC</a> function, with <code>StoreChains = TRUE</code> )                                                                                        |
| StoreDir          | Directory where ‘.Rds’ file is stored. Default: <code>StoreDir = getwd()</code>                                                                                                                                                                 |
| StoreUpdatedChain | Only required when the input files contain an outdated version of a <a href="#">BASiCS_Chain</a> object. If <code>StoreUpdatedChain = TRUE</code> , an updated object is saved (this overwrites original input file, if it was an ‘.Rds’ file). |

## Value

An object of class [BASiCS\\_Chain](#).

## Author(s)

Catalina A. Vallejos <[cvallej@uc.cl](mailto:cvallej@uc.cl)>, Nils Eling

## References

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

## See Also

[BASiCS\\_Chain](#)

**Examples**

```
Data <- makeExampleBASiCS_Data()
Chain <- BASiCS_MCMC(Data, N = 50, Thin = 5, Burn = 5,
                    StoreChains = TRUE, StoreDir = tempdir(),
                    RunName = 'Test')
ChainLoad <- BASiCS_LoadChain(RunName = 'Test', StoreDir = tempdir())
```

BASiCS\_MCMC

*BASiCS MCMC sampler***Description**

MCMC sampler to perform Bayesian inference for single-cell mRNA sequencing datasets using the model described in Vallejos et al (2015).

**Usage**

```
BASiCS_MCMC(Data, N, Thin, Burn, ...)
```

**Arguments**

|      |                                                                                                                                                                                                                                                                                      |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Data | A <code>SingleCellExperiment</code> object. This MUST be formatted to include the spike-ins information (see vignette).                                                                                                                                                              |
| N    | Total number of iterations for the MCMC sampler. Use $N \geq \max(4, \text{Thin})$ , N being a multiple of Thin.                                                                                                                                                                     |
| Thin | Thinning period for the MCMC sampler. Use $\text{Thin} \geq 2$ .                                                                                                                                                                                                                     |
| Burn | Burn-in period for the MCMC sampler. Use $\text{Burn} \geq 1$ , $\text{Burn} < N$ , Burn being a multiple of Thin.                                                                                                                                                                   |
| ...  | Optional parameters.                                                                                                                                                                                                                                                                 |
|      | <code>PriorDelta</code> Specifies the prior used for delta. Possible values are 'gamma' ( <code>Gamma(a.theta, b.theta)</code> prior) and 'log-normal' ( <code>log-Normal(0, s2.delta)</code> prior).. Default value: <code>PriorDelta = 'log-normal'</code> .                       |
|      | <code>PriorParam</code> List of 7 elements, containing the hyper-parameter values required for the adopted prior (see Vallejos et al, 2015, 2016). All elements must be positive real numbers.                                                                                       |
|      | <code>s2.mu</code> Scale hyper-parameter for the <code>log-Normal(0, s2.mu)</code> prior that is shared by all gene-specific expression rate parameters $\mu_i$ . Default: <code>s2.mu = 0.5</code> .                                                                                |
|      | <code>s2.delta</code> Only used when ' <code>PriorDelta == 'log-normal'</code> '. Scale hyper-parameter for the <code>log-Normal(0, s2.delta)</code> prior that is shared by all gene-specific over-dispersion parameters $\delta_i$ . Default: <code>s2.delta = 0.5</code> .        |
|      | <code>a.delta</code> Only used when ' <code>PriorDelta == 'gamma'</code> '. Shape hyper-parameter for the <code>Gamma(a.delta, b.delta)</code> prior that is shared by all gene-specific biological over-dispersion parameters $\delta_i$ . Default: <code>a.delta = 1</code> .      |
|      | <code>b.delta</code> Only used when ' <code>PriorDelta == 'gamma'</code> '. Rate hyper-parameter for the <code>Gamma(a.delta, b.delta)</code> prior that is shared by all gene-specific biological over-dispersion hyper-parameters $\delta_i$ . Default: <code>b.delta = 1</code> . |

- `p.phi` Dirichlet hyper-parameter for the joint of all (scaled by  $n$ ) cell-specific mRNA content normalising constants  $\phi_j/n$ . Default: `p.phi = rep(1, n)`.
- `a.s` Shape hyper-parameter for the  $\text{Gamma}(a.s, b.s)$  prior that is shared by all cell-specific capture efficiency normalising constants  $s_j$ . Default: `a.s = 1`.
- `b.s` Rate hyper-parameter for the  $\text{Gamma}(a.s, b.s)$  prior that is shared by all cell-specific capture efficiency normalising constants  $s_j$ . Default: `b.s = 1`.
- `a.theta` Shape hyper-parameter for the  $\text{Gamma}(a.theta, b.theta)$  prior for technical noise parameter  $\theta$ . Default: `a.theta = 1`.
- `b.theta` Rate hyper-parameter for the  $\text{Gamma}(a.theta, b.theta)$  prior for technical noise parameter  $\theta$ . Default: `b.theta = 1`.
- `AR` Optimal acceptance rate for adaptive Metropolis Hastings updates. It must be a positive number between 0 and 1. Default (and recommended): `AR = 0.44`.
- `StopAdapt` Iteration at which adaptive proposals are not longer adapted. Use `StopAdapt >= 1`. Default: `StopAdapt = Burn`.
- `StoreChains` If `StoreChains = TRUE`, the generated `BASiCS_Chain` object is stored as a `‘.Rds’` file (`RunName` argument used to index the file name). Default: `StoreChains = FALSE`.
- `StoreAdapt` If `StoreAdapt = TRUE`, trajectory of adaptive proposal variances (in log-scale) for all parameters is stored as a list in a `‘.Rds’` file (`RunName` argument used to index file name). Default: `StoreAdapt = FALSE`.
- `StoreDir` Directory where output files are stored. Only required if `StoreChains = TRUE` and/or `StoreAdapt = TRUE`). Default: `StoreDir = getwd()`.
- `RunName` String used to index `‘.Rds’` files storing chains and/or adaptive proposal variances.
- `PrintProgress` If `PrintProgress = FALSE`, console-based progress report is suppressed.
- `ls.phi0` Starting value for the adaptive concentration parameter of the Metropolis proposals for  $\phi = (\phi_1, \dots, \phi_n)'$ .
- `Start` Starting values for the MCMC sampler. We do not advise to specify this argument. Default options have been tuned to facilitate convergence. If changed, it must be a list containing the following elements: `mu0`, `delta0`, `phi0`, `s0`, `nu0`, `theta0`, `ls.mu0`, `ls.delta0`, `ls.phi0`, `ls.nu0` and `ls.theta0`

**Value**

An object of class `BASiCS_Chain`.

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl> and Nils Eling

**References**

- Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.
- Vallejos, Richardson and Marioni (2016). Genome Biology.

**Examples**

```

# Built-in simulated dataset
Data = makeExampleBASiCS_Data()
# To analyse real data, please refer to the instructions in:
# https://github.com/catavallejos/BASiCS/wiki/2.-Input-preparation

# Only a short run of the MCMC algorithm for illustration purposes
# Longer runs might be required to reach convergence
Chain <- BASiCS_MCMC(Data, N = 50, Thin = 2, Burn = 10,
                    PrintProgress = FALSE)

# For illustration purposes we load a built-in 'BASiCS_Chain' object
# (obtained using the 'BASiCS_MCMC' function)
data(ChainSC)

# `displayChainBASiCS` can be used to extract information from this output.
# For example:
head(displayChainBASiCS(ChainSC, Param = 'mu'))

# Traceplot (examples only)
plot(ChainSC, Param = 'mu', Gene = 1)
plot(ChainSC, Param = 'phi', Cell = 1)
plot(ChainSC, Param = 'theta', Batch = 1)

# Calculating posterior medians and 95% HPD intervals
ChainSummary <- Summary(ChainSC)

# `displaySummaryBASiCS` can be used to extract information from this output.
# For example:
head(displaySummaryBASiCS(ChainSummary, Param = 'mu'))

# Graphical display of posterior medians and 95% HPD intervals
# For example:
plot(ChainSummary, Param = 'mu', main = 'All genes')
plot(ChainSummary, Param = 'mu', Genes = 1:10, main = 'First 10 genes')
plot(ChainSummary, Param = 'phi', main = 'All cells')
plot(ChainSummary, Param = 'phi', Cells = 1:5, main = 'First 5 cells')
plot(ChainSummary, Param = 'theta')

# To contrast posterior medians of cell-specific parameters
# For example:
par(mfrow = c(1,2))
plot(ChainSummary, Param = 'phi', Param2 = 's', SmoothPlot = FALSE)
# Recommended for large numbers of cells
plot(ChainSummary, Param = 'phi', Param2 = 's', SmoothPlot = TRUE)

# To contrast posterior medians of gene-specific parameters
par(mfrow = c(1,2))
plot(ChainSummary, Param = 'mu', Param2 = 'delta', log = 'x',
     SmoothPlot = FALSE)
# Recommended
plot(ChainSummary, Param = 'mu', Param2 = 'delta', log = 'x',
     SmoothPlot = TRUE)

# Highly and lowly variable genes detection (within a single group of cells)

```

```

DetectHVG <- BASiCS_DetectHVG(ChainSC, VarThreshold = 0.60,
                             EFDR = 0.10, Plot = TRUE)
DetectLVG <- BASiCS_DetectLVG(ChainSC, VarThreshold = 0.40,
                              EFDR = 0.10, Plot = TRUE)

plot(ChainSummary, Param = 'mu', Param2 = 'delta', log = 'x', col = 8)
with(DetectHVG$Table, points(Mu[HVG == TRUE], Delta[HVG == TRUE],
                             pch = 16, col = 'red', cex = 1))
with(DetectLVG$Table, points(Mu[LVG == TRUE], Delta[LVG == TRUE],
                             pch = 16, col = 'blue', cex = 1))

# If variance thresholds are not fixed
BASiCS_VarThresholdSearchHVG(ChainSC,
                             VarThresholdsGrid = seq(0.55,0.65,by=0.01),
                             EFDR = 0.10)
BASiCS_VarThresholdSearchLVG(ChainSC,
                              VarThresholdsGrid = seq(0.35,0.45,by=0.01),
                              EFDR = 0.10)

# To obtain denoised rates / counts, see:
help(BASiCS_DenoisedRates)
help(BASiCS_DenoisedCounts)

# For examples of differential analyses between 2 populations of cells see:
help(BASiCS_TestDE)

```

---

BASiCS\_Sim

*Generates synthetic data according to the model implemented in BASiCS*

---

## Description

BASiCS\_Sim creates a simulated dataset from the model implemented in BASiCS.

## Usage

```
BASiCS_Sim(Mu, Mu_spikes, Delta, Phi, S, Theta)
```

## Arguments

|           |                                                                                                                                                                                                   |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Mu        | Gene-specific mean expression parameters $\mu_i$ for all biological genes (vector of length <code>q.bio</code> , all elements must be positive numbers)                                           |
| Mu_spikes | $\mu_i$ for all technical genes defined as true input molecules (vector of length <code>q-q.bio</code> , all elements must be positive numbers)                                                   |
| Delta     | Gene-specific biological over-dispersion parameters $\delta_i$ , biological genes only (vector of length <code>q.bio</code> , all elements must be positive numbers)                              |
| Phi       | Cell-specific mRNA content normalising parameters $\phi_j$ (vector of length <code>n</code> , all elements must be positive numbers and the sum of its elements must be equal to <code>n</code> ) |
| S         | Cell-specific technical normalising parameters $s_j$ (vector of length <code>n</code> , all elements must be positive numbers)                                                                    |
| Theta     | Technical variability parameter $\theta$ (must be positive)                                                                                                                                       |

**Value**

An object of class `SingleCellExperiment`, including synthetic data generated by the model implemented in BASiCS.

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>, Nils Eling

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**Examples**

```
# Simulated parameter values for 10 genes
# (7 biological and 3 spike-in) measured in 5 cells
Mu <- c(8.36, 10.65, 4.88, 6.29, 21.72, 12.93, 30.19)
Mu_spike <- c(1010.72, 7.90, 31.59)
Delta <- c(1.29, 0.88, 1.51, 1.49, 0.54, 0.40, 0.85)
Phi <- c(1.00, 1.06, 1.09, 1.05, 0.80)
S <- c(0.38, 0.40, 0.38, 0.39, 0.34)
Theta <- 0.39

Data <- BASiCS_Sim(Mu, Mu_spike, Delta, Phi, S, Theta)
head(assay(Data))
dim(assay(Data))
metadata(Data)$SpikeInput
isSpike(Data)
```

---

BASiCS\_Summary

*The BASiCS\_Summary class*

---

**Description**

Container of a summary of a `BASiCS_Chain` object. In each slot, first column contains posterior medians; second and third columns respectively contain the lower and upper limits of an high posterior density interval (for a given probability).

**Slots**

**parameters** List of parameters in which each entry contains a matrix: first column contains posterior medians, second column contains the lower limits of an high posterior density interval and third column contains the upper limits of high posterior density intervals.

**mu** Posterior medians (1st column), lower (2nd column) and upper (3rd column) limits of gene-specific mean expression parameters  $\mu_i$ .

**delta** Posterior medians (1st column), lower (2nd column) and upper (3rd column) limits of gene-specific biological over-dispersion parameters  $\delta_i$ , biological genes only

**phi** Posterior medians (1st column), lower (2nd column) and upper (3rd column) limits of cell-specific mRNA content normalisation parameters  $\phi_j$

- s** Posterior medians (1st column), lower (2nd column) and upper (3rd column) limits of cell-specific technical normalisation parameters  $s[j]$
- nu** Posterior medians (1st column), lower (2nd column) and upper (3rd column) limits of cell-specific random effects  $\nu_j$
- theta** Posterior median (1st column), lower (2nd column) and upper (3rd column) limits of technical over-dispersion parameter(s)  $\theta$  (each row represents one batch)

### Examples

```
# A BASiCS_Summary object created by the Summary method.
Data = makeExampleBASiCS_Data()
Chain <- BASiCS_MCMC(Data, N = 100, Thin = 2, Burn = 2)
ChainSummary <- Summary(Chain)
```

---

BASiCS\_Summary-methods

*'show' method for BASiCS\_Summary objects*

---

### Description

'show' method for [BASiCS\\_Summary](#) objects.

### Usage

```
## S4 method for signature 'BASiCS_Summary'
show(object)
```

### Arguments

object            A [BASiCS\\_Summary](#) object.

### Value

Prints a summary of the properties of object.

### Author(s)

Catalina A. Vallejos <cnvallej@uc.cl>

### References

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

### Examples

```
# See
help(BASiCS_MCMC)
```



BASiCS\_TestDE

*Detection of genes with changes in expression***Description**

Function to assess changes in expression between two groups of cells (mean and over-dispersion)

**Usage**

```
BASiCS_TestDE(Chain1, Chain2, EpsilonM = log2(1.5), EpsilonD = log2(1.5),
  ProbThresholdM = NULL, ProbThresholdD = NULL, OrderVariable = "Prob",
  GroupLabel1 = "Group1", GroupLabel2 = "Group2", Plot = TRUE,
  PlotOffset = TRUE, Offset = TRUE, EFDR_M = 0.1, EFDR_D = 0.1,
  GenesSelect = NULL, ...)
```

**Arguments**

|                |                                                                                                                                                                                                                                                                                                                                                           |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Chain1         | an object of class <code>BASiCS_Chain</code> containing parameter estimates for the first group of cells                                                                                                                                                                                                                                                  |
| Chain2         | an object of class <code>BASiCS_Chain</code> containing parameter estimates for the second group of cells                                                                                                                                                                                                                                                 |
| EpsilonM       | Minimum fold change tolerance threshold for detecting changes in overall expression (must be a positive real number). Default value: $\text{EpsilonM} = \log_2(1.5)$ (i.e. 50% increase).                                                                                                                                                                 |
| EpsilonD       | Minimum fold change tolerance threshold for detecting changes in biological over-dispersion (must be a positive real number). Default value: $\text{EpsilonM} = \log_2(1.5)$ (i.e. 50% increase).                                                                                                                                                         |
| ProbThresholdM | Optional parameter. Probidence threshold for detecting changes in overall expression (must be a positive value, between 0 and 1)                                                                                                                                                                                                                          |
| ProbThresholdD | Optional parameter. Probidence threshold for detecting changes in cell-to-cell biological over-dispersion (must be a positive value, between 0 and 1)                                                                                                                                                                                                     |
| OrderVariable  | Ordering variable for output. Possible values: 'GeneIndex', 'GeneName' and 'Prob'.                                                                                                                                                                                                                                                                        |
| GroupLabel1    | Label assigned to reference group. Default: <code>GroupLabel1 = 'Group1'</code>                                                                                                                                                                                                                                                                           |
| GroupLabel2    | Label assigned to reference group. Default: <code>GroupLabel2 = 'Group2'</code>                                                                                                                                                                                                                                                                           |
| Plot           | If <code>Plot = TRUE</code> , MA and volcano plots are generated.                                                                                                                                                                                                                                                                                         |
| PlotOffset     | If <code>Plot = TRUE</code> , the offset effect is visualised.                                                                                                                                                                                                                                                                                            |
| Offset         | Optional argument to remove a fix offset effect (if not previously removed from the MCMC chains). Default: <code>Offset = TRUE</code> .                                                                                                                                                                                                                   |
| EFDR_M         | Target for expected false discovery rate related to the comparison of means. Default <code>EFDR_M = 0.10</code> .                                                                                                                                                                                                                                         |
| EFDR_D         | Target for expected false discovery rate related to the comparison of dispersions. Default <code>EFDR_D = 0.10</code> .                                                                                                                                                                                                                                   |
| GenesSelect    | Optional argument to provide a user-defined list of genes to be considered for the comparison. Default: <code>GenesSelect = NULL</code> . When used, this argument must be a vector of <code>TRUE</code> (include gene) / <code>FALSE</code> (exclude gene) indicator, with the same length as the number of intrinsic genes and following the same order |

as how genes are displayed in the table of counts. This argument is necessary in order to have a meaningful EFDR calibration when the user decides to exclude some genes from the comparison.

... Graphical parameters (see [par](#)).

## Value

BASiCS\_TestDE returns a list of 4 elements:

**TableMean** A [data.frame](#) containing the results of the differential mean test

**GeneName** Gene name

**MeanOverall** For each gene, the estimated mean expression parameter  $\mu_i$  is averaged across both groups of cells (weighted by sample size).

**Mean1** Estimated mean expression parameter  $\mu_i$  for each biological gene in the first group of cells.

**Mean2** Estimated mean expression parameter  $\mu_i$  for each biological gene in the second group of cells.

**MeanFC** Fold change in mean expression parameters between the first and second groups of cells.

**MeanLog2FC** Log2-transformed fold change in mean expression between the first and second groups of cells.

**ProbDiffMean** Posterior probability for mean expression difference between the first and second groups of cells.

**ResultDiffExp** Indicator if a gene has a higher mean expression in the first or second groups of cells.

**TableDisp** A [data.frame](#) containing the results of the differential dispersion test (excludes genes for which the mean does not changes).

**GeneName** Gene name

**MeanOverall** For each gene, the estimated mean expression parameter  $\mu_i$  is averaged across both groups of cells (weighted by sample size).

**DispOverall** For each gene, the estimated over-dispersion parameter  $\delta_i$  is averaged across both groups of cells (weighted by sample size).

**Disp1** Estimated over-dispersion parameter  $\delta_i$  for each biological gene in the first group of cells.

**Disp2** Estimated over-dispersion parameter  $\delta_i$  for each biological gene in the second group of cells.

**DispFC** Fold change in over-dispersion parameters between the between the first and second groups of cells.

**DispLog2FC** Log-transformed fold change in over-dispersion between the first and second groups of cells.

**ProbDiffDisp** Posterior probability for over-dispersion difference between the first and second groups of cells.

**ResultDiffDisp** Indicator if a gene has a higher over-dispersion in the first or second groups of cells.

**DiffExpSummary** A list containing the following information for the differential mean expression test:

**ProbThreshold** Posterior probability threshold.

**EFDR** Expected false discovery rate for the given thresholds.

**EFNR** Expected false negative rate for the given thresholds.

**DiffOverDispSummary** A list containing the following information for the differential over-dispersion test:

- ProbThreshold** Posterior probability threshold.
- EFDR** Expected false discovery rate for the given thresholds.
- EFNR** Expected false negative rate for the given thresholds.
- Chain1\_offset** an **BASiCS\_Chain** object: Chain1 after offset removal.
- Chain2\_offset** an **BASiCS\_Chain** object: Chain2 after offset removal (this is only provided for completeness; Chain2 is not affected by the offset).
- OffsetChain** MCMC chain calculated for the offset effect.
- Offset** Estimated offset (posterior median of **OffsetChain**). Default value set equal to 1 when offset correction is not performed.

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl> and Nils Eling

**References**

Vallejos, Richardson and Marioni (2016). *Genome Biology*.

**Examples**

```
# Loading two 'BASiCS_Chain' objects (obtained using 'BASiCS_MCMC')
data(ChainSC)
data(ChainRNA)

Test <- BASiCS_TestDE(Chain1 = ChainSC, Chain2 = ChainRNA,
                     GroupLabel1 = 'SC', GroupLabel2 = 'P&S',
                     EpsilonM = log2(1.5), EpsilonD = log2(1.5),
                     OffSet = TRUE)

# Results for the differential mean test
head(Test$TableMean)

# Results for the differential over-dispersion test
# This only includes genes marked as 'NoDiff' in Test$TableMean
head(Test$TableDisp)
```

---

BASiCS\_VarianceDecomp *Decomposition of gene expression variability according to BASiCS*

---

**Description**

Function to decompose total variability of gene expression into biological and technical components.

**Usage**

```
BASiCS_VarianceDecomp(Chain, OrderVariable = "BioVarGlobal", Plot = TRUE,
  ...)
```

**Arguments**

|               |                                                                                                                                                            |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Chain         | an object of class <a href="#">BASiCS_Chain</a>                                                                                                            |
| OrderVariable | Ordering variable for output. Possible values: 'GeneName', 'BioVarGlobal', 'TechVarGlobal' and 'ShotNoiseGlobal'. Default: OrderVariable = "BioVarGlobal". |
| Plot          | If TRUE, a barplot of the variance decomposition (global and by batches, if any) is generated. Default: Plot = TRUE.                                       |
| ...           | Other arguments to be passed to <a href="#">barplot</a>                                                                                                    |

**Details**

See vignette

**Value**

A [data.frame](#) whose first 4 columns correspond to

GeneName Gene name (as indicated by user)

BioVarGlobal Percentage of variance explained by a biological component (overall across all cells)

TechVarGlobal Percentage of variance explained by the technical component (overall across all cells)

ShotNoiseGlobal Percentage of variance explained by the shot noise component (baseline Poisson noise, overall across all cells)

If more than 1 batch of cells are being analysed, the remaining columns contain the corresponding variance decomposition calculated within each batch.

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**See Also**

[BASiCS\\_Chain](#)

**Examples**

```
# See  
help(BASiCS_MCMC)
```

---

BASiCS\_VarThresholdSearchHVG

*Detection method for highly and lowly variable genes using a grid of variance contribution thresholds*

---

## Description

Detection method for highly and lowly variable genes using a grid of variance contribution thresholds

## Usage

```
BASiCS_VarThresholdSearchHVG(Chain, VarThresholdsGrid, EFDR = 0.1,  
  Progress = TRUE)
```

```
BASiCS_VarThresholdSearchLVG(Chain, VarThresholdsGrid, EFDR = 0.1,  
  Progress = TRUE)
```

## Arguments

|                   |                                                                                              |
|-------------------|----------------------------------------------------------------------------------------------|
| Chain             | an object of class <a href="#">BASiCS_Chain</a>                                              |
| VarThresholdsGrid | Grid of values for the variance contribution threshold (they must be contained in (0,1))     |
| EFDR              | Target for expected false discovery rate related to HVG/LVG detection. Default: EFDR = 0.10. |
| Progress          | If Progress = TRUE, partial output is printed in the console. Default: Progress = TRUE.      |

## Details

See vignette

## Value

BASiCS\_VarThresholdSearchHVG A table displaying the results of highly variable genes detection for different variance contribution thresholds.

BASiCS\_VarThresholdSearchLVG A table displaying the results of lowly variable genes detection for different variance contribution thresholds.

## Author(s)

Catalina A. Vallejos <cnvallej@uc.cl>

## References

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

## See Also

[BASiCS\\_Chain](#)

**Examples**

```
# See
help(BASiCS_MCMC)
```

---

|          |                                                                                         |
|----------|-----------------------------------------------------------------------------------------|
| ChainRNA | <i>Extract from the chain obtained in Vallejos et al (2016): pool-and-split samples</i> |
|----------|-----------------------------------------------------------------------------------------|

---

**Description**

Small extract (100 MCMC iterations, 500 randomly selected genes) from the chain obtained in Vallejos et al (2016), related to pool-and-split samples (this corresponds to the RNA 2i samples in Grun et al, 2014).

**Usage**

```
ChainRNA
```

**Format**

An object of class [BASiCS\\_Chain](#) containing 10 MCMC iterations.

**References**

Vallejos, Richardson and Marioni (2016). Genome Biology.  
 Grun, Kester and van Oudenaarden (2014). Nature Methods.

---

|         |                                                                                      |
|---------|--------------------------------------------------------------------------------------|
| ChainSC | <i>Extract from the chain obtained in Vallejos et al (2016): single-cell samples</i> |
|---------|--------------------------------------------------------------------------------------|

---

**Description**

Small extract (100 MCMC iterations, 500 randomly selected genes) from the chain obtained in Vallejos et al (2016), related to single-cell samples (this corresponds to the SC 2i samples in Grun et al, 2014).

**Usage**

```
ChainSC
```

**Format**

An object of class [BASiCS\\_Chain](#) containing 10 MCMC iterations.

**References**

Vallejos, Richardson and Marioni (2016). Genome Biology.  
 Grun, Kester and van Oudenaarden (2014). Nature Methods.

---

displayChainBASiCS-BASiCS\_Chain-method

*Accessors for the slots of a BASiCS\_Chain object*

---

## Description

Accessors for the slots of a [BASiCS\\_Chain](#)

## Usage

```
## S4 method for signature 'BASiCS_Chain'  
displayChainBASiCS(object, Param = "mu")
```

## Arguments

|        |                                                                                                             |
|--------|-------------------------------------------------------------------------------------------------------------|
| object | an object of class <a href="#">BASiCS_Chain</a>                                                             |
| Param  | Name of the slot to be used for the accessed. Possible values: 'mu', 'delta', 'phi', 's', 'nu' and 'theta'. |

## Value

The requested slot of a [BASiCS\\_Chain](#) object

## Author(s)

Catalina A. Vallejos <cnvallej@uc.cl>

## References

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

## See Also

[BASiCS\\_Chain](#)

## Examples

```
# See  
help(BASiCS_MCMC)
```

---

displaySummaryBASiCS-BASiCS\_Summary-method

*Accessors for the slots of a [BASiCS\\_Summary](#) object*

---

## Description

Accessors for the slots of a [BASiCS\\_Summary](#) object

## Usage

```
## S4 method for signature 'BASiCS_Summary'  
displaySummaryBASiCS(object, Param = "mu")
```

## Arguments

|        |                                                                                                            |
|--------|------------------------------------------------------------------------------------------------------------|
| object | an object of class <a href="#">BASiCS_Summary</a>                                                          |
| Param  | Name of the slot to be used for the accessed. Possible values: 'mu', 'delta', 'phi', 's', 'nu' and 'theta' |

## Value

The requested slot of a [BASiCS\\_Summary](#) object

## Author(s)

Catalina A. Vallejos <cnvallej@uc.cl>

## References

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

## See Also

[BASiCS\\_Summary](#)

## Examples

```
# See  
help(BASiCS_MCMC)
```



---

`makeExampleBASiCS_Data`

*Create a synthetic SingleCellExperiment example object with the format required for BASiCS*

---

### Description

A synthetic `SingleCellExperiment` object is generated by simulating a dataset from the model underlying BASiCS. This is used to illustrate BASiCS in some of the package and vignette examples.

### Usage

```
makeExampleBASiCS_Data(WithBatch = FALSE, WithSpikes = TRUE)
```

### Arguments

|                         |                                                                                                                |
|-------------------------|----------------------------------------------------------------------------------------------------------------|
| <code>WithBatch</code>  | If TRUE, 2 batches are generated (each of them containing 10 cells). Default: <code>WithBatch = FALSE</code> . |
| <code>WithSpikes</code> | If TRUE, the simulated dataset contains 20 spike-in genes. Default: <code>WithSpikes = TRUE</code> .           |

### Value

An object of class `SingleCellExperiment`, with synthetic data simulated from the model implemented in BASiCS. If `WithSpikes = TRUE`, it contains 70 genes (50 biological and 20 spike-in) and 20 cells. Alternatively, it contains 50 biological genes and 20 cells.

### Author(s)

Catalina A. Vallejos <cnvallej@uc.cl> and Nils Eling

### References

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.  
Vallejos, Marioni and Richardson (2016). Genome Biology.

### Examples

```
Data <- makeExampleBASiCS_Data()  
is(Data, 'SingleCellExperiment')
```

---

|                 |                                                                    |
|-----------------|--------------------------------------------------------------------|
| newBASiCS_Chain | <i>Creates a BASiCS_Chain object from pre-computed MCMC chains</i> |
|-----------------|--------------------------------------------------------------------|

---

### Description

BASiCS\_Chain creates a [BASiCS\\_Chain](#) object from pre-computed MCMC chains.

### Usage

```
newBASiCS_Chain(parameters)
```

### Arguments

**parameters** List of matrices containing MCMC chains for each model parameter.

**mu** MCMC chain for gene-specific mean expression parameters  $\mu_i$ , biological genes only (matrix with q.bio columns, all elements must be positive numbers)

**delta** MCMC chain for gene-specific biological over-dispersion parameters  $\delta_i$ , biological genes only (matrix with q.bio columns, all elements must be positive numbers)

**phi** MCMC chain for cell-specific mRNA content normalisation parameters  $\phi_j$  (matrix with n columns, all elements must be positive numbers and the sum of its elements must be equal to n)

**s** MCMC chain for cell-specific technical normalisation parameters  $s_j$  (matrix with n columns, all elements must be positive numbers)

**nu** MCMC chain for cell-specific random effects  $\nu_j$  (matrix with n columns, all elements must be positive numbers)

**theta** MCMC chain for technical over-dispersion parameter(s)  $\theta$  (matrix, all elements must be positive, each column represents 1 batch)

### Value

An object of class [BASiCS\\_Chain](#).

### Author(s)

Catalina A. Vallejos <cnvallej@uc.cl>

### References

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

### See Also

[BASiCS\\_Chain](#)

**Examples**

```
Data <- makeExampleBASiCS_Data()
Chain <- BASiCS_MCMC(Data, N = 50, Thin = 5, Burn = 5)

ChainMu <- displayChainBASiCS(Chain, 'mu')
ChainDelta <- displayChainBASiCS(Chain, 'delta')
ChainPhi <- displayChainBASiCS(Chain, 'phi')
ChainS <- displayChainBASiCS(Chain, 's')
ChainNu <- displayChainBASiCS(Chain, 'nu')
ChainTheta <- displayChainBASiCS(Chain, 'theta')

ChainNew <- newBASiCS_Chain(parameters = list(mu = ChainMu, delta = ChainDelta,
      phi = ChainPhi, s = ChainS,
      nu = ChainNu, theta = ChainTheta))
```

---

|                |                                                                                                                                   |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------|
| newBASiCS_Data | <i>Creates a SingleCellExperiment object from a matrix of expression counts and experimental information about spike-in genes</i> |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------|

---

**Description**

newBASiCS\_Data creates a [SingleCellExperiment](#) object from a matrix of expression counts and experimental information about spike-in genes.

**Usage**

```
newBASiCS_Data(Counts, Tech, SpikeInfo, BatchInfo = NULL)
```

**Arguments**

|           |                                                                                                                                                                                                                                       |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Counts    | Matrix of dimensions $q$ times $n$ whose elements contain the expression counts to be analysed (including biological and technical spike-in genes). Gene names must be stored as <code>rownames(Counts)</code> .                      |
| Tech      | Logical vector of length $q$ . If <code>Tech = FALSE</code> the gene is biological; otherwise the gene is spike-in.                                                                                                                   |
| SpikeInfo | <code>data.frame</code> whose first and second columns contain the gene names assigned to the spike-in genes (they must match the ones in <code>rownames(Counts)</code> ) and the associated input number of molecules, respectively. |
| BatchInfo | Vector of length $n$ whose elements indicate batch information. Not required if a single batch is present on the data. Default value: <code>BatchInfo = NULL</code> .                                                                 |

**Value**

An object of class [SingleCellExperiment](#).

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl> and Nils Eling

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**See Also**

[SingleCellExperiment](#)

**Examples**

```
# Expression counts
set.seed(1)
Counts <- matrix(rpois(50*10, 2), ncol = 10)
rownames(Counts) <- c(paste0('Gene', 1:40), paste0('Spike', 1:10))

# Technical information
Tech <- c(rep(FALSE,40),rep(TRUE,10))

# Spikes input number of molecules
set.seed(2)
SpikeInfo <- data.frame(gene=rownames(Counts)[Tech],amount=rgamma(10,1,1))

# Creating a BASiCS_Data object (no batch effect)
DataExample <- newBASiCS_Data(Counts, Tech, SpikeInfo)

# Creating a BASiCS_Data object (with batch effect)
BatchInfo <- c(rep(1, 5), rep(2, 5))
DataExample <- newBASiCS_Data(Counts, Tech, SpikeInfo, BatchInfo)
```

---

plot-BASiCS\_Chain-method

*'plot' method for BASiCS\_Chain objects*

---

**Description**

'plot' method for [BASiCS\\_Chain](#) objects

**Usage**

```
## S4 method for signature 'BASiCS_Chain,ANY'
plot(x, Param = "mu", Gene = NULL,
     Cell = NULL, Batch = 1, ylab = "", xlab = "", ...)
```

**Arguments**

|       |                                                                                                        |
|-------|--------------------------------------------------------------------------------------------------------|
| x     | A <a href="#">BASiCS_Chain</a> object.                                                                 |
| Param | Name of the slot to be used for the plot. Possible values: 'mu', 'delta', 'phi', 's', 'nu' and 'theta' |
| Gene  | Specifies which gene is requested. Required only if Param = 'mu' or 'delta'                            |
| Cell  | Specifies which cell is requested. Required only if Param = 'phi', 's' or 'nu'                         |

|       |                                                                      |
|-------|----------------------------------------------------------------------|
| Batch | Specifies which batch is requested. Required only if Param = 'theta' |
| ylab  | As in <a href="#">par</a> .                                          |
| xlab  | As in <a href="#">par</a> .                                          |
| ...   | Other graphical parameters (see <a href="#">par</a> ).               |

**Value**

A plot object

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**Examples**

```
# See
help(BASiCS_MCMC)
```

---

plot-BASiCS\_Summary-method

*'plot' method for BASiCS\_Summary objects*

---

**Description**

'plot' method for [BASiCS\\_Summary](#) objects

**Usage**

```
## S4 method for signature 'BASiCS_Summary,ANY'
plot(x, Param = "mu", Param2 = NULL,
     Genes = NULL, Cells = NULL, Batches = NULL, xlab = "", ylab = "",
     xlim = "", ylim = "", pch = 16, col = "blue", bty = "n",
     SmoothPlot = TRUE, ...)
```

**Arguments**

|        |                                                                                                                                                                                         |
|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| x      | A <a href="#">BASiCS_Summary</a> object.                                                                                                                                                |
| Param  | Name of the slot to be used for the plot. Possible values: 'mu', 'delta', 'phi', 's', 'nu' and 'theta'                                                                                  |
| Param2 | Name of the second slot to be used for the plot. Possible values: 'mu', 'delta', 'phi', 's' and 'nu' (combinations between gene-specific and cell-specific parameters are not admitted) |
| Genes  | Specifies which genes are requested. Required only if Param = 'mu' or 'delta'                                                                                                           |

|            |                                                                                  |
|------------|----------------------------------------------------------------------------------|
| Cells      | Specifies which cells are requested. Required only if Param = 'phi', 's' or 'nu' |
| Batches    | Specifies which batches are requested. Required only if Param = 'theta'          |
| xlab       | As in <a href="#">par</a> .                                                      |
| ylab       | As in <a href="#">par</a> .                                                      |
| xlim       | As in <a href="#">par</a> .                                                      |
| ylim       | As in <a href="#">par</a> .                                                      |
| pch        | As in <a href="#">par</a> .                                                      |
| col        | As in <a href="#">par</a> .                                                      |
| btty       | As in <a href="#">par</a> .                                                      |
| SmoothPlot | Logical parameter. If TRUE, transparency will be added to the color of the dots. |
| ...        | Other graphical parameters (see <a href="#">par</a> ).                           |

**Value**

A plot object

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**Examples**

```
# See
help(BASiCS_MCMC)
```

---

Summary

*'Summary' method for BASiCS\_Chain objects*

---

**Description**

For each of the BASiCS parameters (see Vallejos et al 2015), Summary returns the corresponding posterior medians and limits of the high posterior density interval (probabilty equal to prob)

**Usage**

```
## S4 method for signature 'BASiCS_Chain'
Summary(x, prob = 0.95)
```

**Arguments**

x                    A [BASiCS\\_Chain](#) object.  
 prob                prob argument for [HPDinterval](#) function.

**Value**

An object of class [BASiCS\\_Summary](#).

**Author(s)**

Catalina A. Vallejos <cnvallej@uc.cl>

**References**

Vallejos, Marioni and Richardson (2015). PLoS Computational Biology.

**Examples**

```
help(BASiCS_MCMC)
```

# Index

## \*Topic **datasets**

- ChainRNA, [22](#)
- ChainSC, [22](#)
- barplot, [20](#)
- BASiCS\_Chain, [2](#), [3–7](#), [10](#), [12](#), [15](#), [17](#), [19–23](#), [26](#), [28](#), [30](#)
- BASiCS\_Chain-class (BASiCS\_Chain), [2](#)
- BASiCS\_Chain-methods, [3](#)
- BASiCS\_D\_TestDE, [8](#)
- BASiCS\_DenoisedCounts, [4](#)
- BASiCS\_DenoisedRates, [5](#)
- BASiCS\_DetectHVG, [6](#)
- BASiCS\_DetectHVG\_LVG (BASiCS\_DetectHVG), [6](#)
- BASiCS\_DetectLVG (BASiCS\_DetectHVG), [6](#)
- BASiCS\_Filter, [8](#)
- BASiCS\_LoadChain, [10](#)
- BASiCS\_MCMC, [2](#), [10](#), [11](#)
- BASiCS\_Sim, [14](#)
- BASiCS\_Summary, [15](#), [16](#), [24](#), [29](#), [31](#)
- BASiCS\_Summary-class (BASiCS\_Summary), [15](#)
- BASiCS\_Summary-methods, [16](#)
- BASiCS\_TestDE, [8](#), [17](#)
- BASiCS\_VarianceDecomp, [19](#)
- BASiCS\_VarThresholdSearchHVG, [21](#)
- BASiCS\_VarThresholdSearchHVG\_LVG (BASiCS\_VarThresholdSearchHVG), [21](#)
- BASiCS\_VarThresholdSearchLVG (BASiCS\_VarThresholdSearchHVG), [21](#)
- ChainRNA, [22](#)
- ChainSC, [22](#)
- data.frame, [18](#), [20](#)
- displayChainBASiCS (displayChainBASiCS-BASiCS\_Chain-method), [23](#)
- displayChainBASiCS, BASiCS\_Chain-method (displayChainBASiCS-BASiCS\_Chain-method), [23](#)
- displayChainBASiCS-BASiCS\_Chain-method, [23](#)
- displaySummaryBASiCS (displaySummaryBASiCS-BASiCS\_Summary-method), [24](#)
- displaySummaryBASiCS, BASiCS\_Summary-method (displaySummaryBASiCS-BASiCS\_Summary-method), [24](#)
- displaySummaryBASiCS-BASiCS\_Summary-method, [24](#)
- HPDinterval, [30](#)
- makeExampleBASiCS\_Data, [25](#)
- newBASiCS\_Chain, [26](#)
- newBASiCS\_Data, [27](#)
- par, [6](#), [18](#), [29](#), [30](#)
- plot (plot-BASiCS\_Chain-method), [28](#)
- plot, BASiCS\_Chain, ANY-method (plot-BASiCS\_Chain-method), [28](#)
- plot, BASiCS\_Chain-method (plot-BASiCS\_Chain-method), [28](#)
- plot, BASiCS\_Summary, ANY-method (plot-BASiCS\_Summary-method), [29](#)
- plot, BASiCS\_Summary-method, (plot-BASiCS\_Summary-method), [29](#)
- plot-BASiCS\_Chain-method, [28](#)
- plot-BASiCS\_Summary-method, [29](#)
- show, BASiCS\_Chain-method (BASiCS\_Chain-methods), [3](#)
- show, BASiCS\_Summary-method (BASiCS\_Summary-methods), [16](#)
- SingleCellExperiment, [4](#), [5](#), [7](#), [8](#), [11](#), [15](#), [25](#), [27](#), [28](#)
- Summary, [30](#)
- Summary, BASiCS\_Chain-method (Summary), [30](#)
- updateObject, [4](#)
- updateObject, BASiCS\_Chain-method (BASiCS\_Chain-methods), [3](#)