Package ‘MOSim’

May 20, 2024

Title  Multi-Omics Simulation (MOSim)
Version  2.0.0
Description  MOSim package simulates multi-omic experiments that mimic regulatory mechanisms within the cell, allowing flexible experimental design including time course and multiple groups.
Encoding  UTF-8
Depends  R (>= 4.2.0)
License  GPL-3
LazyData  false
biocViews  Software, TimeCourse, ExperimentalDesign, RNASeq
BugReports  https://github.com/ConesaLab/MOSim/issues
URL  https://github.com/ConesaLab/MOSim
Imports  HiddenMarkov, zoo, IRanges, S4Vectors, dplyr, ggplot2, lazyeval, matrixStats, methods, rlang, stringi, stringr, scran, Seurat, Signac, edgeR, Rcpp
Suggests  testthat, knitr, markdown, codetools, BiocStyle, stats, utils, purrr, scales, tibble, tidyr, Biobase, scater, SingleCellExperiment, decor, markdown, Rsamtools, igraph, leiden, bluster
RoxygenNote  7.3.1
VignetteBuilder  knitr
LinkingTo  cpp11, Rcpp
git_url  https://git.bioconductor.org/packages/MOSim
git_branch  RELEASE_3_19
git_last_commit  bee9b3a
Contents

MOSim-package ........................................... 3
associationList ........................................ 3
check_patterns .......................................... 4
discretize ............................................. 4
experimentalDesign ..................................... 5
is.declared ........................................... 6
make_association_dataframe .............................. 6
make_cluster_patterns ................................ 7
mosim ................................................... 7
MOSimulation-class ...................................... 9
MOSimulator-class ...................................... 10
MOSimulatorRegion-class ............................... 11
omicData .............................................. 12
omicResults ........................................... 13
omicSettings ......................................... 14
omicSim ............................................... 15
order_FC_forMatrix .................................... 16
plotProfile ........................................... 17
random_unif_interval .................................. 18
sampleData ........................................... 18
scatac ................................................ 19
scMOSim ............................................... 19
scOmicResults ....................................... 21
scOmicSettings ....................................... 21
scrna ................................................... 22
sc_omicData ......................................... 23
sc_param_estimation .................................... 23
shuffle_group_matrix .................................. 25
simulate_coexpression .................................. 25
simulate_hyper ........................................ 26
sparsim_create_simulation_parameter .................. 27
sparsim_estimate_intensity ............................. 28
sparsim_estimate_library_size ......................... 29
sparsim_estimate_parameter_from_data ................ 29
sparsim_estimate_variability ........................... 30
sparsim_simulation .................................... 30
**Description**

Multiomics simulation package.

**Author(s)**

Maintainer: Sonia Tarazona <sotacam@gmail.com>

Authors:

- Carolina Monzó <carolmonzoc@gmail.com>
- Carlos Martínez <cmarmir@gmail.com>

**See Also**

Useful links:

- [https://github.com/ConesaLab/MOSim](https://github.com/ConesaLab/MOSim)
- Report bugs at [https://github.com/ConesaLab/MOSim/issues](https://github.com/ConesaLab/MOSim/issues)

---

**associationList**

*Data to showcase scRNA and scATAC-seq association*

**Description**

Data to showcase scRNA and scATAC-seq association

**Usage**

data("associationList")

**Format**

A dataframe with two columns and rows according to gene/feature relationships

- **Peak_ID** ATAC chromosomal positions associated to genes
- **Gene_ID** RNA genes associated to peaks

@source Created in-house to serve as an example
**check_patterns**

**Description**

Function to check if the TRUE FALSE patterns have at least two rows that are opposite, we need this to be able to generate repressor regulators

**Usage**

```r
check_patterns(patterns_ret)
```

**Arguments**

- `patterns_ret` tibble of TRUE FALSE values

**Value**

list of indices where the rows are opposite

**Examples**

```r
patterns <- tibble::tibble(
  one = c(TRUE, FALSE, TRUE, FALSE),
  two = c(TRUE, TRUE, TRUE, TRUE),
  three = c(FALSE, TRUE, FALSE, TRUE),
  four = c(FALSE, TRUE, TRUE, TRUE))
opposite_indices <- check_patterns(patterns)
```

**discretize**

*Discretize ChIP-Seq counts to simulate a binary dataset*

**Description**

Discretize ChIP-Seq counts to simulate a binary dataset

**Usage**

```r
discretize(df, omic)
```

**Arguments**

- `df` A MOSimulated object
- `omic` Character string of the omic to transform into binary data
experimentalDesign

Value

A regulator dataframe of 0 and 1

Examples

omic_list <- c("RNA-seq", "ChIP-seq")
rnaseq_simulation <- mosim(omics = omic_list,
    omicsOptions = c(omicSim("ChIP-seq", totalFeatures = 2500)))
rnaseq_simulated <- omicResults(rnaseq_simulation, omic_list)
discrete_ChIP <- discretize(rnaseq_simulated, "ChIP-seq")

design_matrix <- experimentalDesign(rnaseq_simulation)

experimentalDesign

Retrieves the experimental design

Description

Retrieves the experimental design

Usage

experimentalDesign(simulation)

Arguments

simulation A MOSimulation object

Value

A data frame containing the experimental design used to simulate the data.

Examples

omic_list <- c("RNA-seq")
rnaseq_simulation <- mosim(omics = omic_list)
# This will be a data frame with RNA-seq counts

design_matrix <- experimentalDesign(rnaseq_simulation)
**is.declared**

*Check if a variable is declared.*

**Description**

Check if a variable is declared.

**Usage**

```r
is.declared(object, key = NULL)
```

**Arguments**

- `object`: Variable name to check
- `key`: Optional key to check inside object.

**Value**

TRUE or FALSE indicating if the variable is initialized & non-empty.

---

**make_association_dataframe**

`make_association_dataframe`

**Description**

This function generates a dataframe containing the information of the relationship between ATAC and RNA, based on the cluster groups, and then tells the order the genes and peaks should be in the simulated dataframe of the group.

**Usage**

```r
make_association_dataframe(group, genereggroup)
```

**Arguments**

- `group`: Group from which we are generating the association dataframe
- `genereggroup`: list of elements to generate the association dataframe such as clusters of each omic, indices of opposite clusters, which genes are activated, repressed, behavior of the features etc.

**Value**

a dataframe with all the information the user needs about each gene and the order of gene and peak names to rename them in the simulated datasets of the group.
### make_cluster_patterns

**Description**

Function to make the tibble with cluster combinations for the gene expression patterns along the cells

**Usage**

```r
make_cluster_patterns(numcells = 4, clusters = 8)
```

**Arguments**

- `numcells`: Number of different celltypes we are simulating
- `clusters`: OPTIONAL. Number of co-expression patterns the user wants to simulate

**Value**

A tibble with number of columns equal to number of celltypes, rows according to the number of TRUE/FALSE combinations corresponding to the gene expression patterns along the cells

**Examples**

```r
patterns <- make_cluster_patterns(numcells = 4, clusters = 8)
cell_types <- list('Treg' = c(1:10), 'cDC' = c(11:20), 'CD4_TEM' = c(21:30), 'Memory_B' = c(31:40))
patterns <- make_cluster_patterns(numcells = length(cell_types), clusters = 8)
```

### mosim

**Description**

Performs a multiomic simulation by chaining two actions: 1) Creating the "MOSimulation" class with the provided params. 2) Calling "simulate" method on the initialized object.
mosim(
  omics,
  omicsOptions,
  diffGenes,
  numberReps,
  numberGroups,
  times,
  depth,
  profileProbs,
  minMaxFC,
  TFtoGene
)

Arguments

**omics** Character vector containing the names of the omics to simulate, which can be "RNA-seq", "miRNA-seq", "DNase-seq", "ChIP-seq" or "Methyl-seq" (e.g. c("RNA-seq", "miRNA-seq")). It can also be a list with the omic names as names and their options as values, but we recommend to use the argument omicSim to provide the options to simulated each omic.

**omicsOptions** List containing the options to simulate each omic. We recommend to apply the helper method omicSim to create this list in a friendly way, and the function omicData to provide custom data (see the related sections for more information). Each omic may have different configuration parameters, but the common ones are:

  - **simuData/idToGene** Seed sample and association tables for regulatory omics. The helper function omicData should be used to provide this information (see the following section).
  - **regulatorEffect** For regulatory omics. List containing the percentage of effect types (repressor, activator or no effect) over the total number of regulators. See vignette for more information.
  - **totalFeatures** Number of features to simulate. By default, the total number of features in the seed dataset.
  - **depth** Sequencing depth in millions of reads. If not provided, it takes the global parameter passed to mosim function.
  - **replicateParams** List with parameters \(a\) and \(b\) for adjusting the variability in the generation of replicates using the negative binomial. See vignette for more information.

**diffGenes** Number of differentially expressed genes to simulate, given in percentage (0 - 1) or in absolute number (> 1). By default 0.15

**numberReps** Number of replicates per experimental condition (and time point, if time series are to be generated). By default 3.

**numberGroups** Number of experimental groups or conditions to simulate.

**times** Vector of time points to consider in the experimental design.

**depth** Sequencing depth in millions of reads.
MOSimulation-class

profileProbs Numeric vector with the probabilities to assign each of the patterns. Defaults to 0.2 for each.

minMaxFC Numeric vector of length 2 with minimum and maximum fold-change for differentially expressed features, respectively.

TFtoGene A logical value indicating if default transcription factors data should be used (TRUE) or not (FALSE), or a 3 column data frame containing custom associations. By default FALSE.

Value

Instance of class "MOSimulation" containing the multiomic simulation data.

Examples

```r
moSimulation <- mosim(
  omics = c("RNA-seq"),
  numberReps = 3,
  times = c(0, 2, 6, 12, 24)
)

# Retrieve simulated count matrix for RNA-seq
dataRNAseq <- omicResults(moSimulation, "RNA-seq")
```

Description

This class manages the global simulation process, like associating genes with gene classes, regulatory programs and other settings. Finally it will initialize the simulators with their options that will use the previously generated settings to simulate the data.

Slots

- simulators Vector containing either S4 initialized classes of simulators or a list with the class name as keys, and its options as value, see example.
- totalGenes A number with the total number of genes including not expressed. Overwritten if a genome reference is provided. Currently not used as we force to provide real data.
- diffGenes A number with the total number of differential genes (if value > 1) or % or total genes (if value < 1).
- numberReps Number of replicates of the experiment.
numberGroups  Number of samples considered on the experiment.
times  Numeric vector containing the measured times. If numberGroups < 2, the number of times 
must be at least 2.
geneNames  Read only. List containing the IDs of the genes. Overwritten by the genome reference 
if provided. Currently not used as we force to provide real data.
simSettings  List of settings that overrides initializing the configuration of the simulation by pass-
ing a previously generated list. This could be used to tweak by hand the assigned profiles, 
genes, regulatory programs, etc.
noiseFunction  Noise function to apply when simulating counts. Must accept the parameter ’n’ 
and return a vector of the same length. Defaults to ‘rnorm’
profiles  Named list containing the patterns with their coefficients.
profileProbs  Numeric vector with the probabilities to assign each of the patterns. Defaults to 0.2 
for each.
noiseParams  Default noise parameters to be used with noise function.
depth  Default depth to simulate.
TFtoGene  Boolean (for default data) or 3 column data frame containing Symbol-TFGene-LinkedGene
minMaxQuantile  Numeric vector of length 2 indicating the quantiles to use in order to retrieve the 
absolute minimum and maximum value that a differentially expressed feature can have.
minMaxFC  Numeric vector of length 2 indicating the minimum and maximum fold-change that a 
differentially expressed feature can have.

---

MOSimulator-class  Virtual class containing common methods and slots for child classes.

Description

Virtual class containing common methods and slots for child classes.

Slots

name  Name of the simulator to be used in messages.
data  Data frame containing the initial sample to be used, with the features IDs as rownames and 
only one column named "Counts".
regulator  Boolean flag to indicate if the omic is a regulator or not.
regulatorEffect  Possible regulation effects of the omic (enhancer, repressor or both).
idToGene  Data frame with the association table between genes and other features. The structure 
must be 2 columns, one named "ID" and the other "Gene".
min  Minimum value allowed in the omic.
max  Maximum value allowed in the omic.
depth  Sequencing depth to simulate.
depthRound  Number of decimal places to round when adjusting depth.
depthAdjust  Boolean indicating whether to adjust by sequencing depth or not.
totalFeatures  Number of features to simulate. This will replace the data with a subset.
noiseFunction  Noise function to apply when simulating counts. Must accept the parameter ‘n’
               and return a vector of the same length. Defaults to ‘rnorm’
increment  Read-only. Minimum value to increase when simulating counts.
simData  Contains the final simulated data.
pregenerated  Indicates if the child class will generate the simulated data instead of the general
              process.
randData  Auxiliary vector containing the original count data in random order with other adjust-
           ments.
noiseParams  Noise parameters to be used with noise function.
roundDigits  Number of digits to round the simulated count values.
minMaxQuantile  Numeric vector of length 2 indicating the quantiles to use in order to retrieve
                the absolute minimum and maximum value that a differentially expressed feature can have.
minMaxFC  Numeric vector of length 2 indicating the minimum and maximum fold-change that a
differentially expressed feature can have.
minMaxDist  Named list containing different minimum and maximum constraints values calculated
            at the beginning of the simulation process.
replicateParams  Named list containing the parameters a and b to be used in the replicates gener-
                 ation process, see the vignette for more info.

MOSimulatorRegion-class

Virtual class containing general methods for simulators based on regions of the chromosomes, like DNase-seq, ChiP-seq or Methyl-seq

Description

Virtual class containing general methods for simulators based on regions of the chromosomes, like DNase-seq, ChiP-seq or Methyl-seq
Class to simulate RNA-seq data
Class to simulate transcription factor data
Class to simulate miRNA-seq
Class to simulate ChiP-seq data
Class to simulate DNase-seq data
Class to simulate Methyl-seq data.
Slots

- `locs` Vector containing the list of locations of the sites.
- `locsName` Type of the site to simulate, only for debug.
- `splitChar` Character symbol used to split identifiers in chr/start/end.
- `nCpG` numeric. Number of CpG sites to simulate.
- `pSuccessDemethReg` numeric. Probability of success in non methylated region.
- `errorMethReg` numeric. Error rate in methylated region.
- `errorDemethReg` numeric. Error rate in methylated region.
- `nReadsMethReg` numeric. Mean number of reads in methylated region.
- `nReadsDemethReg` numeric. Mean number of reads in non methylated regions.
- `phaseDiff` numeric. Phase difference in the differentially methylated regions between two samples.
- `ratesHMMMatrix` numeric. Matrix of values that describes the exponential decay functions that define the distances between CpG values.
- `distType` character. Distribution used to generate replicates.
- `transitionSize` numeric.
- `PhiMeth` matrix. Transition matrix for CpG locations.
- `PhiDemeth` matrix. <Not used>
- `typesLocation` numeric. <Not used>
- `returnValue` character. Selected column.
- `betaThreshold` numeric. Beta threshold value used to calculate M values.

omicData

Set customized data for an omic.

Description

Set customized data for an omic.

Usage

omicData(omic, data = NULL, associationList = NULL)

Arguments

- `omic` The name of the omic to provide data.
- `data` Data frame with the omic identifiers as row names and just one column named Counts containing numeric values used as initial sample for the simulation.
- `associationList` Only for regulatory omics, a data frame with 2 columns, the first called containing the regulator ID and the second called Gene with the gene identifier.
Value

Initialized simulation object with the given data.

Examples

# Take a subset of the included dataset for illustration purposes. We could also load it from a csv file or RData, # as long as we transform it to have 1 column named "Counts" # and the identifiers as row names.

data(sampleData)

custom_rnaseq <- head(sampleData$SimRNAseq$data, 100)

# In this case, 'custom_rnaseq' is a data frame with # the structure:
head(custom_rnaseq)
## Counts
## ENSMUSG0000000001 6572
## ENSMUSG0000000003 0
## ENSMUSG0000000028 4644
## ENSMUSG0000000031 8
## ENSMUSG0000000037 0
## ENSMUSG0000000049 0

# The helper 'omicData' returns an object with our custom data.
rnaseq_customdata <- omicData("RNA-seq", data = custom_rnaseq)

omicResults  Retrieves the simulated data.

Description

Retrieves the simulated data.

Usage

omicResults(simulation, omics = NULL, format = "data.frame")

Arguments

simulation  A MOSimulation object.
omics  List of the omics to retrieve the simulated data.
format  Type of object to use for returning the results
omicSettings

Value

A list containing an element for every omic specific, with the simulation data in the format indicated, or a numeric matrix with simulated data if the omic name is directly provided.

Examples

```r
omic_list <- c("RNA-seq")
rnaseq_simulation <- mosim(omics = omic_list)
  # This will be a data frame with RNA-seq counts
rnaseq_simulated <- omicResults(rnaseq_simulation, "RNA-seq")
```

```
# Group1.Time0.Rep1 Group1.Time0.Rep2 Group1.Time0.Rep3 ...
# ENSMUSG00000073155 4539 5374 5808 ...
# ENSMUSG00000026251 0 0 0 ...
# ENSMUSG00000040472 2742 2714 2912 ...
# ENSMUSG00000021598 5256 4640 5130 ...
# ENSMUSG00000032348 421 348 492 ...
# ENSMUSG00000097226 16 14 9 ...
# ENSMUSG00000027857 0 0 0 ...
# ENSMUSG00000032081 1 0 0 ...
# ENSMUSG00000097164 794 822 965 ...
# ENSMUSG00000097871 0 0 0 ...
```

omicSettings

Retrieves the settings used in a simulation

Description

Retrieves the settings used in a simulation

Usage

```r
omicSettings(
  simulation,
  omics = NULL,
  association = FALSE,
  reverse = FALSE,
  only.linked = FALSE,
  prefix = FALSE,
  include.lagged = TRUE
)
```

Arguments

- **simulation**: A MOSimulation object.
- **omics**: List of omics to retrieve the settings.
- **association**: A boolean indicating if the association must also be returned for the regulators.
omicSim

Set the simulation settings for an omic.

Description

Set the simulation settings for an omic.

Usage

omicSim(omic, depth = NULL, totalFeatures = NULL, regulatorEffect = NULL)

Arguments

omic Name of the omic to set the settings.
depth Sequencing depth in millions of counts. If not provided will take the global parameter passed to mosim function.
totalFeatures Limit the number of features to simulate. By default include all present in the dataset.
regulatorEffect

only for regulatory omics. Associative list containing the percentage of effects over the total number of regulator, including repressor, association and no effect (NE).

Value

A list with the appropriate structure to be given as options in mosim function.

Examples

```r
omic_list <- c("RNA-seq", "miRNA-seq")
rnaseq_options <- c(omicSim("miRNA-seq", totalFeatures = 2500))
# The return value is an associative list compatible with
# 'omicsOptions'
rnaseq_simulation <- mosim(omics = omic_list, omicsOptions = rnaseq_options)
```

order_FC_forMatrix

Function to sort the FC values according to the genes that must be up or downregulated

Usage

```r
order_FC_forMatrix(A, B, C, D)
```

Arguments

A Vector of c("Up", "Down", "NE") from the Gene or Peak_DE extracted from the association matrix
B Calculated vector of Up FC values
C Calculated vector of Down FC values
D Calculated vector of NE FC values

Examples

```r
Up_FCvec <- c(1, 1, 1)
Down_FCvec <- c(2, 2)
notDE_FCvec <- c(2, 2, 2, 2)
FC_vec <- order_FC_forMatrix(DE, Up_FCvec, Down_FCvec, notDE_FCvec)
```
**plotProfile**

*Generate a plot of a feature’s profile for one or two omics.*

### Description

Generate a plot of a feature’s profile for one or two omics.

### Usage

```r
plotProfile(simulation, omics, featureIDS, drawReps = FALSE, groups = NULL)
```

### Arguments

- **simulation**: A MOSimulation object
- **omics**: Character vector of the omics to simulate.
- **featureIDS**: List containing the feature to show per omic. Must have the omics as the list names and the features as values.
- **drawReps**: Logical to enable/disable the representation of the replicates inside the plot.
- **groups**: Character vector indicating the groups to plot in the form “GroupX” (i.e. Group1)

### Value

A ggplot2 object.

### Examples

```r
omic_list <- c("RNA-seq", "miRNA-seq")

rnaseq_options <- c(omicSim("miRNA-seq", totalFeatures = 2500))
rnaseq_simulation <- mosim(omics = omic_list,
                           omicsOptions = rnaseq_options)

#plotProfile(rnaseq_simulation,
#            omics = c("RNA-seq", "miRNA-seq"),
#            featureIDS = list("RNA-seq"="ENSMUSG00000007682", "miRNA-seq"="mmu-miR-320-3p")
#)
```
random_unif_interval  

**random_unif_interval Function to call the C code**

**Description**
random_unif_interval Function to call the C code

**Usage**
random_unif_interval(size, max_val)

**Arguments**
- size from sparsim
- max_val from sparsim

---

**sampleData**  

**Default data**

**Description**
Dataset with base counts and id-gene tables.

**Usage**
data("sampleData")

**Format**
An object of class list of length 6.

**Details**
List with 6 elements:
- **SimRNAseq data** Dataframe with base counts with gene id as rownames.
  - geneLength Length of every gene.
- **SimChIPseq data** Dataframe with base counts with regions as rownames.
  - idToGene Dataframe with region as "ID" column and gene name on "Gene" column.
- **SimDNaseseq data** Dataframe with base counts with regions as rownames.
  - idToGene Dataframe with region as "ID" column and gene name on "Gene" column.
- **SimMiRNAseq data** Dataframe with base counts with miRNA id as rownames.
  - idToGene Dataframe with miRNA as "ID" column and gene name on "Gene" column.
- **SimMethylseq idToGene** Dataframe with region as "ID" column and gene name on "Gene" column.
- **CpGisland** Dataframe of CpG to be used as initialization data, located on "Region" column.
Data to test scMOSim

Usage

data("scatac")

Format

A seurat Object, subset from seuratData with ATAC

assays  ATAC expression values
meta.data annotations of celltypes

@source https://github.com/satijalab/seurat-data, we took 11 cells from each of 4 celltypes

scMOSim

Performs multiomic simulation of single cell datasets

Usage

scMOSim(
  omics,
  cellTypes,
  numberReps = 1,
  numberGroups = 1,
  diffGenes = NULL,
  minFC = 0.25,
  maxFC = 4,
  numberCells = NULL,
  mean = NULL,
  sd = NULL,
  noiseRep = 0.1,
  noiseGroup = 0.5,
  regulatorEffect = NULL,
  associationList = NULL,
  feature_no = 8000,
  clusters = 3,
  cluster_size = NULL
)
**Arguments**

- **omics** named list containing the omic to simulate as names, which can be "scRNA-seq" or "scATAC-seq".
- **cellTypes** list where the i-th element of the list contains the column indices for i-th experimental conditions. List must be a named list.
- **numberReps** OPTIONAL. Number of replicates per group
- **numberGroups** OPTIONAL. Number of different groups
- **diffGenes** OPTIONAL. If number groups > 1, Percentage DE genes to simulate. List of vectors (one per group to compare to group 1) where the vector contains absolute number of genes for Up and Down ex: c(250, 500) or a percentage for up, down ex: c(0.2, 0.2). The rest will be NE
- **minFC** OPTIONAL. Threshold of FC below which are downregulated, by default 0.25
- **maxFC** OPTIONAL. Threshold of FC above which are upregulated, by default 4
- **numberCells** OPTIONAL. Vector of numbers. The numbers correspond to the number of cells the user wants to simulate per each cell type. The length of the vector must be the same as length of cellTypes.
- **mean** OPTIONAL. Vector of numbers of mean depth per each cell type. Must be specified just if numberCells is specified. The length of the vector must be the same as length of cellTypes.
- **sd** OPTIONAL. Vector of numbers of standard deviation per each cell type. Must be specified just if numberCells is specified. The length of the vector must be the same as length of cellTypes.
- **noiseRep** OPTIONAL. Number indicating the desired standard deviation between biological replicates.
- **noiseGroup** OPTIONAL. Number indicating the desired standard deviation between treatment groups
- **regulatorEffect** OPTIONAL. To simulate relationship scRNA-scATAC, list of vectors (one per group) where the vector contains absolute number of regulators for Activator and repressor ex: c(150, 200) or a percentage for Activator and repressor ex: c(0.2, 0.1). The rest will be NE. If not provided, no table of association between scRNA and scATAC is outputted.
- **associationList** REQUIRED A 2 columns dataframe reporting peak ids related to gene names. If user doesn't have one, load from package data("associationList")
- **feature_no** OPTIONAL. If only scRNA-seq to simulate or scRNA and scATAC but no regulatory constraints, total number of features to be distributed between the coexpression clusters.
- **clusters** OPTIONAL. Number of co-expression patterns the user wants to simulate
- **cluster_size** OPTIONAL. It may be inputted by the user. Recommended: by default, its the number of features divided by the number of patterns to generate.

**Value**

a list of Seurat object, one per each omic.
Examples

```r
omic_list <- sc_omicData(list("scRNA-seq"))
cell_types <- list('Treg' = c(1:10), 'cDC' = c(11:20), 'CD4_TEM' = c(21:30),
                    'Memory_B' = c(31:40))
sim <- scMOSim(omic_list, cell_types)
```

Description

scOmicResults

Usage

```r
scOmicResults(sim)
```

Arguments

- `sim`: a simulated object from scMOSim function

Value

list of seurat objects with simulated data

Examples

```r
cell_types <- list('Treg' = c(1:10), 'cDC' = c(11:20), 'CD4_TEM' = c(21:30),
                    'Memory_B' = c(31:40))
omic_list <- sc_omicData(list("scRNA-seq"))
sim <- scMOSim(omic_list, cell_types)
res <- scOmicResults(sim)
```

Description

scOmicSettings

Usage

```r
scOmicSettings(sim)
```
**Arguments**

sim | a simulated object from scMOSim function

**Value**

list of Association matrices explaining the effects of each regulator to each gene

**Examples**

cell_types <- list('Treg' = c(1:10), 'cDC' = c(11:20), 'CD4_TEM' = c(21:30), 'Memory_B' = c(31:40))

omicsList <- sc_omicData(list("scRNA-seq"))

sim <- scMOSim(omicsList, cell_types)

res <- scOmicSettings(sim)

---

**scrna** | *Data to test scMOSim*

---

**Description**

Data to test scMOSim

**Usage**

data("scrna")

**Format**

A seurat Object, subset from seuratData with RNA

assays  RNA expression values

meta.data annotations of celltypes

@source https://github.com/satijalab/seurat-data, we took 11 cells from each of 4 celltypes This is how: dat <- pbmcMultiome.SeuratData::pbmc.rna dat <- subset(x = dat, subset = seurat_annotations "cDC", "Memory B", "Treg") unique_cell_types <- unique(datATmeta.data$seurat_annotations)

extracted_cells <- list() cellnames <- c() for (cell_type in unique_cell_types) type_cells <- subset(dat, subset = seurat_annotations counts <- as.matrix(type_cellsATassays["RNA"]ATcounts)

extracted_cells[[cell_type]] <- counts[, 1:10] cellnames <- append(cellnames, replicate(11, cell_type))

scrna <- Reduce(cbind, extracted_cells)
**sc_omicData**

**Description**

Checks if the user defined data is in the correct format, or loads the default multiomics pbmc dataset, a subset from SeuratData package.

**Usage**

```r
sc_omicData(omics_types, data = NULL)
```

**Arguments**

- `omics_types`: A list of strings which can be either "scRNA-seq" or "scATAC-seq".
- `data`: A user input matrix with genes (peaks in case of scATAC-seq) as rows and cells as columns. By default, it loads the example data. If a user input matrix is included, cell columns must be sorted by cell type.

**Value**

A named list with omics type as name and the count matrix as value.

**Examples**

```r
# Simulate from PBMC
omicsList <- sc_omicData(list("scRNA-seq", "scATAC-seq"))
```

**sc_param_estimation**

**Description**

Evaluate the users parameters for single cell simulation and use SPARSim to simulate the main dataset. Internal function.

**Usage**

```r
sc_param_estimation(
  omics,
  cellTypes,
  diffGenes = list(c(0.2, 0.2)),
  minFC = 0.25,
  maxFC = 4,
  numberCells = NULL,
)```

**Arguments**

- `omics`:
- `cellTypes`:
- `diffGenes`: a list of length 2 containing the minimum and maximum fold change.
- `minFC`:
- `maxFC`:
- `numberCells`:

**Examples**

```r
# Example usage
params <- sc_param_estimation(omis, cellTypes, diffGenes = list(0.2, 0.2), minFC = 0.25, maxFC = 4, numberCells = NULL)
```
mean = NULL,
sd = NULL,
noiseGroup = 0.5,
group = 1,
genereggroup
)

Arguments

- **omics**
  named list containing the omics to simulate as names, which can be "scRNA-seq" or "scATAC-seq".

- **cellTypes**
  list where the i-th element of the list contains the column indices for i-th cell type. List must be a named list.

- **diffGenes**
  If number groups > 1, Percentage DE genes to simulate. List of vectors (one per group to compare to group 1) where the vector contains absolute number of genes for Up and Down ex: c(250, 500) or a percentage for up, down ex: c(0.2, 0.2). The rest will be NE

- **minFC**
  Threshold of FC below which are downregulated, by default 0.25

- **maxFC**
  Threshold of FC above which are upregulated, by default 4

- **numberCells**
  vector of numbers. The numbers correspond to the number of cells the user wants to simulate per each cell type. The length of the vector must be the same as length of cellTypes.

- **mean**
  vector of numbers of mean depth per each cell type. Must be specified just if numberCells is specified.

- **sd**
  vector of numbers of standard deviation per each cell type. Must be specified just if numberCells is specified.

- **noiseGroup**
  OPTIONAL. Number indicating the desired standard deviation between treatment groups

- **group**
  Group for which to estimate parameters

- **genereggroup**
  List with information of genes, clusters and regulators that must be related to each other

Value

- a list of Seurat object, one per each omic.

- a named list with simulation parameters for each omics as values.

Examples

```r
omicsList <- sc_omicData(list("scRNA-seq"))
cell_types <- list("Treg" = c(1:10), "cDC" = c(11:20), "CD4_TEM" = c(21:30), "Memory_B" = c(31:40))
#estimated_params <- sc_param_estimation(omicsList, cell_types)
```
**shuffle_group_matrix**

*shuffle_group_matrix*, Reorder cell type-specific expression matrix during co-expression simulation. Copied from ACORDE (https://github.com/ConesaLab/acorde) to facilitate stability and running within our scripts

**Description**

This function is used internally by acorde to perform the shuffling of simulated features for an individual cell type, as part of the co-expression simulation process. The function is called recursively by `simulate_coexpression()` to perform the simulation on a full scRNA-seq matrix.

**Usage**

```r
shuffle_group_matrix(sim_data, feature_ids, group_pattern, ngroups)
```

**Arguments**

- **sim_data**: A count matrix with features as rows and cells as columns. Feature IDs must be included in an additional column named `feature`.
- **feature_ids**: A two-column tibble containing `top` and `bottom` columns, each including the feature IDs of features to be used as highly or lowly expressed when shuffling by the indicated expression pattern.
- **group_pattern**: A logical vector, containing `TRUE` to indicate that high expression in that cell type is desired and `FALSE` if the opposite. The vector must be ordered as the cell types in `sim_data`.
- **ngroups**: An integer indicating the number of groups that top and bottom features should be divided into. It is computed by dividing the number of features selected as highly/lowly expressed by the size of the clusters that are to be generated.

**Value**

An expression matrix, with the same characteristics as `sim_data`, and a number of features defined as the total amount of top/bottom features selected divided by the number of clusters for which co-expression patterns where supplied.

**simulate_coexpression**

*simulate coexpression*

**Description**

Adapted from ACORDE (https://github.com/ConesaLab/acorde) to adapt to our data input type. Simulates coexpression of genes along celltypes
Usage

simulate_coexpression(
  sim_matrix,
  feature_no,
  cellTypes,
  patterns,
  cluster_size = NULL
)

Arguments

sim_matrix       Matrix with rows as features and columns as cells
feature_no       Total number of features to be distributed between the coexpression clusters
cellTypes        list where the i-th element of the list contains the column indices for i-th experimental conditions. List must be a named list.
patterns         Tibble with TRUE FALSE depicting the cluster patterns to simulate. Generated by the user or by make_cluster_patterns.
cluster_size     OPTIONAL. It may be inputted by the user. By default, its the number of features divided by the number of patterns to generate.

Value

the simulated coexpression

simulate_hyper            Simulate technical variability

Description

Function to simulate the technical variability (i.e. a multivariate hypergeometric on a gamma expression value array)

Usage

simulate_hyper(avgAbund, seqdepth = NULL, digits, max_val)

Arguments

avgAbund           array containing the intensity values for each feature. It describes the intensity of a single sample
seqdepth           sequencing depth (i.e. sample size of the MH)
digits             number of digits for random number generation
max_val            max value for random number generation

Value

An array of length(avgAbund) elements representing the count values for the current sample
sparsim_create_simulation_parameter

Create SPARSim simulation parameter

Description

Function to create a SPARSim simulation parameter.

Usage

sparsim_create_simulation_parameter(
  intensity,
  variability,
  library_size,
  feature_names = NA,
  sample_names = NA,
  condition_name = NA,
  intensity_2 = NULL,
  variability_2 = NULL,
  p_bimod = NULL
)

Arguments

intensity Array of gene expression intensity values
variability Array of gene expression variability values
library_size Array of library size values
feature_names Array of feature names. It must be of the same length of intensity array. If NA (default), feature will be automatically named "gene_1", "gene_2", ... "gene_<N>", where N = length(intensity)
sample_names Array of sample names. It must be of the same length of library_size array. If NA (defaul), sample will be automatically named "<condition_name>_cell1", "<condition_name>_cell2", ..., "<condition_name>_cell<M>", where M = length(library_size)
condition_name Name associated to the current experimental condition. If NA (default), it will be set to "cond<l1><l2>", where l1 and l2 are two random letters.
intensity_2 Array of gene expression intensity values for the second expression mode, if simulating genes with bimodal gene expression. Entries containing NAs will be ignored. If NULL (default), no bimodal gene expression is simulated.
variability_2 Array of gene expression variability values for the second expression mode, if simulating genes with bimodal gene expression. If NULL (default), no bimodal gene expression is simulated.
p_bimod Array of bimodal gene expression probabilities; the i-th value indicates the probability p of the i-th gene to be expressed in the first mode (i.e. the one specified in the i-th entries of parameters intensity and variability); with probability 1-p the i-th gene will be expressed in the second mode (i.e. the one specified in the i-th entries of parameters intensity_2 and variability_2)
Details

To simulate N feature (e.g. genes), user must specify N values of gene expression level and gene expression variability in the function input parameters intensity and variability, respectively. To simulate M samples (i.e. cells), user must specify M values of sample library size in the function input parameter library_size.

User can optionally specify the names to assign at the single feature and sample to simulate (function input parameters feature_names and sample_names, respectively, as well as the name of the experimental condition (function input parameter condition_name). If the user does not specify such information, the function will set some default values.

To simulate T different experimental conditions in a single count table, then T different simulation parameters must be created.

Value

SPARSim simulation parameter describing one experimental condition

---

sparsim_estimate_intensity

*Estimate SPARSim “intensity” parameter*

Description

Function to estimate the intensity values from the genes in data. The intensity is computed as mean of normalized counts for each gene.

Usage

sparsim_estimate_intensity(data)

Arguments

data normalized count data matrix (gene on rows, samples on columns). rownames(data) must contain gene names.

Details

This function is used in sparsim_estimate_parameter_from_data to compute SPARSim “intensity” parameter, given a real count table as input. If the count table contains more than one experimental condition, then the function is applied to each experimental conditions.

Value

An array of intensity values having N_genes elements (N_genes = nrow(data)). Array entries are named with gene names.
sparsim_estimate_library_size

Estimate SPARSim "library size" parameter

Description

Function to estimate the library sizes from the samples in data.

Usage

sparsim_estimate_library_size(data)

Arguments

data raw count data matrix (gene on rows, samples on columns)

Details

This function is used in sparsim_estimate_parameter_from_data to compute SPARSim "library size" parameter, given a real count table as input. If the count table contains more than one experimental condition, then the function is applied to each experimental conditions.

Value

An array of library size values having N_samples elements (N_samples = ncol(data))

sparsim_estimate_parameter_from_data

Estimate SPARSim simulation parameter from a given count table

Description

Function to estimate SPARSim simulation parameters (intensity, variability and library sizes) from a real count table. If the real count table contains more than one experimental condition, it is possible to estimate the parameters for each experimental condition.

Usage

sparsim_estimate_parameter_from_data(raw_data, norm_data, conditions)

Arguments

raw_data count matrix (gene on rows, samples on columns) containing raw count data
norm_data count matrix (gene on rows, samples on columns) containing normalized count data
conditions list where the i-th element of the list contains the column indices for i-th experimental conditions. List must be a named list.
sparsim_simulation

**Value**
A SPARSim simulation parameters

---

sparsim_estimate_variability

*Estimate SPARSim "variability" parameter*

**Description**
Function to estimate the variability values from the genes in data.

**Usage**
sparsim_estimate_variability(data)

**Arguments**
- data: raw count data matrix (gene on rows, samples on columns)

**Details**
This function is used in sparsim_estimate_parameter_from_data to compute SPARSim "variability" parameter, given a real count table as input. If the count table contains more than one experimental condition, then the function is applied to each experimental conditions.

**Value**
An array of variability values having N_genes elements (N_genes = nrow(data))

---

sparsim_simulation

*Function to simulate a raw count table*

**Description**
Function to simulate a raw count table

**Usage**
sparsim_simulation(
  dataset_parameter,
  output_sim_param_matrices = FALSE,
  output_batch_matrix = FALSE,
  count_data_simulation_seed = NULL
)
**sparsim_simulation**

**Arguments**

- **dataset_parameter**
  - list containing the intensity, variability and lib sizes of each experimental condition. It is the return value of "estimate_parameter_from_data" or could be created by the users.

- **output_sim_param_matrices**
  - boolean flag. If TRUE, the function will output two additional matrices, called abundance_matrix and variability_matrix, containing the gene intensities and gene variabilities used as simulation input. (Default: FALSE)

- **output_batch_matrix**
  - boolean flag. If TRUE, the function will output an additional matrix, called batch_factors_matrix, containing the multiplicative factors used in batch effect simulation. (Default: FALSE)

- **count_data_simulation_seed**
  - inherited from sparsim

**Value**

A list of 5 elements:

- **count_matrix**: the simulated count matrix (genes on rows, samples on columns)
- **gene_matrix**: the simulated gene expression levels (genes on rows, samples on columns)
- **abundance_matrix**: the input gene intensity values provided as input (genes on rows, samples on columns), if output_sim_param_matrices = TRUE. NULL otherwise.
- **variability_matrix**: the input gene variability values provided as input (genes on rows, samples on columns), if output_sim_param_matrices = TRUE. NULL otherwise.
- **batch_factors_matrix**: the multiplicative factor used in batch generation (genes on rows, samples on columns), if output_batch_matrix = TRUE. NULL otherwise.
Index

* datasets
  sampleData, 18
* internal
  MOSim-package, 3
  MOSimulation-class, 9
  MOSimulator-class, 10
  MOSimulatorRegion-class, 11

  associationList, 3
  check_patterns, 4
  discretize, 4
  experimentalDesign, 5
  is.declared, 6
  make_association_dataframe, 6
  make_cluster_patterns, 7
  MOSim (MOSim-package), 3
  mosim, 7, 8
  MOSim-package, 3
  MOSimulation-class, 9
  MOSimulator-class, 10
  MOSimulatorRegion-class, 11

  omicData, 8, 12
  omicResults, 13
  omicSettings, 14
  omicSim, 8, 15
  order_FC_forMatrix, 16

  plotProfile, 17

  random_unif_interval, 18

  sampleData, 18
  sc_omicData, 23
  sc_param_estimation, 23
  scatac, 19
  scMOSim, 19
  scOmicResults, 21
  scOmicSettings, 21
  scrna, 22
  shuffle_group_matrix, 25
  SimChIPseq-class
    (MOSimulatorRegion-class), 11
  SimDNaseseq-class
    (MOSimulatorRegion-class), 11
  SimMethylseq-class
    (MOSimulatorRegion-class), 11
  SimmiRNASEq-class
    (MOSimulatorRegion-class), 11
  SimRNAseq-class
    (MOSimulatorRegion-class), 11
  SimTF-class (MOSimulatorRegion-class), 11
  simulate_coexpression, 25
  simulate_coexpression(), 25
  simulate_hyper, 26
  sparsim_create_simulation_parameter, 27
  sparsim_estimate_intensity, 28
  sparsim_estimate_library_size, 29
  sparsim_estimate_parameter_from_data, 29
  sparsim_estimate_variability, 30
  sparsim_simulation, 30