Package 'scDesign3'

November 12, 2026

Type Package

Title A unified framework of realistic in silico data generation and statistical model inference for single-cell and spatial omics

Version 1.10.0

Description We present a statistical simulator, scDesign3, to generate realistic single-cell and spatial omics data, including various cell states, experimental designs, and feature modalities, by learning interpretable parameters from real data. Using a unified probabilistic model for single-cell and spatial omics data, scDesign3 infers biologically meaningful parameters; assesses the goodness-of-fit of inferred cell clusters, trajectories, and spatial locations; and generates in silico negative and positive controls for benchmarking computational tools.

License MIT + file LICENSE

Encoding UTF-8

LazyData false

Depends R (>= 4.3.0)

Imports dplyr, tibble, stats, methods, mgcv, gamlss, gamlss.dist, SummarizedExperiment, SingleCellExperiment, mclust, mvtnorm, parallel, pbmcapply, umap, ggplot2, irlba, viridis, BiocParallel, matrixStats, Matrix, sparseMVN, coop

Suggests mvnfast, igraph, rvinecopulib, knitr, rmarkdown, testthat (>= 3.0.0), RefManageR, sessioninfo, BiocStyle

biocViews Software, SingleCell, Sequencing, GeneExpression, Spatial

URL https://github.com/SONGDONGYUAN1994/scDesign3

BugReports https://github.com/SONGDONGYUAN1994/scDesign3/issues

RoxygenNote 7.3.3

Config/testthat/edition 3

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/scDesign3

git_branch RELEASE_3_23

git_last_commit c1f38d2

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Dongyuan Song [aut, cre] (ORCID:
<https://orcid.org/0000-0003-1114-1215>),
Qingyang Wang [aut] (ORCID: <https://orcid.org/0000-0002-1051-609X>),
Chenxin Jiang [aut] (ORCID: <https://orcid.org/0009-0005-7369-4116>)

Maintainer Dongyuan Song <dongyuansong@ucla.edu>

Contents

ba . . . 2
construct_data . . . 3
example_count . . . 5
example_sce . . . 5
extract_para . . . 6
fit_copula . . . 8
fit_marginal . . . 10
ga . . . 12
gamlss.ba . . . 13
gamlss.ga . . . 14
perform_lrt . . . 14
plot_reduceddim . . . 16
pseudotime . . . 17
scdesign3 . . . 17
simu_new . . . 21
sparse_cov . . . 24

Index **25**

`ba` *Functions from gamlss/gamlss.add with bugs fixed*

Description

An additive function to be used while fitting GAMLSS models. The interface for `bam()` in the **mgcv** package.

Usage

```
ba(formula, control = ba.control(...), ...)
```

Arguments

`formula`	A formula of the model.
`control`	The control of the model fitting.
`...`	Other arguments.

Value

A xvar list.

ba

NA

Examples

```
print("No example")
```

`construct_data` *Construct the input data (covariate matrix and expression matrix)*

Description

This function constructs the input data for `fit_marginal`.

Usage

```
construct_data(
  sce,
  assay_use = "counts",
  celltype,
  pseudotime,
  spatial,
  other_covariates,
  ncell = dim(sce)[2],
  corr_by,
  parallelization = "mcmapply",
  BPPARAM = NULL
)
```

Arguments

`sce`	A `SingleCellExperiment` object.
`assay_use`	A string which indicates the assay you will use in the sce. Default is 'counts'.
`celltype`	A string of the name of cell type variable in the `colData` of the sce. Default is 'cell_type'.
`pseudotime`	A string or a string vector of the name of pseudotime and (if exist) multiple lineages. Default is NULL.
`spatial`	A length two string vector of the names of spatial coordinates. Default is NULL.
`other_covariates`	A string or a string vector of the other covariates you want to include in the data.

ncell The number of cell you want to simulate. Default is dim(sce)[2] (the same number as the input data). If an arbitrary number is provided, the function will use Vine Copula to simulate a new covariate matrix.

corr_by A string or a string vector which indicates the groups for correlation structure. If '1', all cells have one estimated corr. If 'ind', no corr (features are independent). If others, this variable decides the corr structures.

parallelization A string indicating the specific parallelization function to use. Must be one of 'mcmapply', 'bpmapply', or 'pbmcmapply', which corresponds to the parallelization function in the package parallel,BiocParallel, and pbmcapply respectively. The default value is 'mcmapply'.

BPPARAM A MulticoreParam object or NULL. When the parameter parallelization = 'mcmapply' or 'pbmcmapply', this parameter must be NULL. When the parameter parallelization = 'bpmapply', this parameter must be one of the MulticoreParam object offered by the package 'BiocParallel. The default value is NULL.

Details

This function takes a SingleCellExperiment object as the input. Based on users' choice, it constructs the matrix of covariates (explanatory variables) and the expression matrix (e.g., count matrix for scRNA-seq).

Value

A list with the components:

count_mat The expression matrix

dat The original covariate matrix

newCovariate The simulated new covariate matrix, is NULL if the parameter ncell is default

filtered_gene The genes that are excluded in the marginal and copula fitting steps because these genes only express in less than two cells.

Examples

```
data(example_sce)
my_data <- construct_data(
sce = example_sce,
assay_use = "counts",
celltype = "cell_type",
pseudotime = "pseudotime",
spatial = NULL,
other_covariates = NULL,
corr_by = "1"
)
```

example_count *A sparse matrix with example data*

Description

A sparse matrix with example data

Usage

```
data("example_count")
```

Format

A dgCMatrix with 100 rows (genes) and 2087 cols (cells)

Value

The corresponding sparse matrix object 'example_count'

example_sce *A SingleCellExperiment object containing both cell type and pseudotime*

Description

A SingleCellExperiment object containing both cell type and pseudotime

Usage

```
data("example_sce")
```

Format

A dataset with 10 rows (genes) and 1289 cols (cells)

Value

The corresponding SingleCellExperiment object

extract_para *Extract the parameters of each cell's distribution*

Description

extract_para generates parameter matrices which determine each cell's distribution

Usage

```
extract_para(
  sce,
  assay_use = "counts",
  marginal_list,
  n_cores,
  family_use,
  new_covariate,
  parallelization = "mcmapply",
  BPPARAM = NULL,
  data
)
```

Arguments

sce	A SingleCellExperiment object.
assay_use	A string which indicates the assay you will use in the sce. Default is 'counts'.
marginal_list	A list of fitted regression models from fit_marginal for each gene in sce.
n_cores	An integer. The number of cores to use.
family_use	A string of the marginal distribution. Must be one of 'poisson', 'nb', 'zip', 'zinb' or 'gaussian', which represent 'poisson distribution', 'negative binomial distribution', 'zero-inflated poisson distribution', 'zero-inflated negative binomial distribution', and 'gaussian distribution' respectively.
new_covariate	A data.frame which contains covariates of targeted simulated data from construct_data and the correlation group assignment for each cell in the column 'corr_group'.
parallelization	A string indicating the specific parallelization function to use. Must be one of 'mcmapply', 'bpmapply', or 'pbmcmapply', which corresponds to the parallelization function in the package parallel,BiocParallel, and pbmcapply respectively. The default value is 'mcmapply'.
BPPARAM	A MulticoreParam object or NULL. When the parameter parallelization = 'mcmapply' or 'pbmcmapply', this parameter must be NULL. When the parameter parallelization = 'bpmapply', this parameter must be one of the MulticoreParam object offered by the package 'BiocParallel. The default value is NULL.
data	A dataframe which is used when fitting the gamlss model

Details

The function takes the new covariate (if use) from `construct_data` and marginal models from `fit_marginal`.

Value

A list with the components:

`mean_mat` A cell by feature matrix of the mean parameter.

`sigma_mat` A cell by feature matrix of the sigma parameter (for Gaussian, the variance; for NB, the dispersion.).

`zero_mat` A cell by feature matrix of the zero-inflation parameter (only non-zero for ZIP and ZINB).

Examples

```
data(example_sce)
my_data <- construct_data(
sce = example_sce,
assay_use = "counts",
celltype = "cell_type",
pseudotime = "pseudotime",
spatial = NULL,
other_covariates = NULL,
corr_by = "1"
)
my_marginal <- fit_marginal(
data = my_data,
mu_formula = "s(pseudotime, bs = 'cr', k = 10)",
sigma_formula = "1",
family_use = "nb",
n_cores = 1,
usebam = FALSE
)
my_copula <- fit_copula(
sce = example_sce,
assay_use = "counts",
marginal_list = my_marginal,
family_use = c(rep("nb", 5), rep("zip", 5)),
copula = "vine",
n_cores = 1,
input_data = my_data$dat
)
my_para <- extract_para(
  sce = example_sce,
  marginal_list = my_marginal,
  n_cores = 1,
  family_use = c(rep("nb", 5), rep("zip", 5)),
  new_covariate = my_data$new_covariate,
  data = my_data$dat
)
```

`fit_copula` *Fit the copula model*

Description

`fit_copula` fits the copula model.

Usage

```
fit_copula(
  sce,
  assay_use,
  input_data,
  empirical_quantile = FALSE,
  marginal_list,
  family_use,
  copula = "gaussian",
  DT = TRUE,
  pseudo_obs = FALSE,
  epsilon = 1e-06,
  family_set = c("gaussian", "indep"),
  important_feature = "all",
  if_sparse = FALSE,
  correlation_function = "default",
  n_cores,
  parallelization = "mcmapply",
  BPPARAM = NULL
)
```

Arguments

`sce`	A `SingleCellExperiment` object.
`assay_use`	A string which indicates the assay you will use in the sce. Default is 'counts'.
`input_data`	The input data, which is one of the output from `construct_data`.
`empirical_quantile`	Please only use it if you clearly know what will happen! A logic variable. If TRUE, DO NOT fit the copula and use the EMPIRICAL CDF values of the original data; it will make the simulated data fixed (no randomness). Default is FALSE. Only works if ncell is the same as your original data.
`marginal_list`	A list of fitted regression models from `fit_marginal`.
`family_use`	A string or a vector of strings of the marginal distribution. Must be one of 'poisson', 'nb', 'zip', 'zinb' or 'gaussian'.

`copula` A string of the copula choice. Must be one of 'gaussian' or 'vine'. Default is 'gaussian'. Note that vine copula may have better modeling of high-dimensions, but can be very slow when features are >1000.

`DT` A logic variable. If TRUE, perform the distributional transformation to make the discrete data 'continuous'. This is useful for discrete distributions (e.g., Poisson, NB). Default is TRUE. Note that for continuous data (e.g., Gaussian), DT does not make sense and should be set as FALSE.

`pseudo_obs` A logic variable. If TRUE, use the empirical quantiles instead of theoretical quantiles for fitting copula. Default is FALSE.

`epsilon` A numeric variable for preventing the transformed quantiles to collapse to 0 or 1.

`family_set` A string or a string vector of the bivariate copula families. Default is c("gaussian", "indep").

`important_feature`
A numeric value or vector which indicates whether a gene will be used in correlation estimation or not. If this is a numeric value, then gene with zero proportion greater than this value will be excluded form gene-gene correlation estimation. If this is a vector, then this should be a logical vector with length equal to the number of genes in `sce`. `TRUE` in the logical vector means the corresponding gene will be included in gene-gene correlation estimation and `FALSE` in the logical vector means the corresponding gene will be excluded from the gene-gene correlation estimation. The default value for is "all" (a special string which means no filtering).

`if_sparse` A logic variable. Only works for Gaussian copula (`family_set = "gaussian"`). If TRUE, a thresholding strategy will make the corr matrix sparse.

`correlation_function`
A string. If 'default', the function from `Rfast`; if 'coop', the function from `coop`, which calls BLAS.

`n_cores` An integer. The number of cores to use.

`parallelization`
A string indicating the specific parallelization function to use. Must be one of 'mcmapply', 'bpmapply', or 'pbmcmapply', which corresponds to the parallelization function in the package `parallel`,`BiocParallel`, and `pbmcapply` respectively. The default value is 'mcmapply'.

`BPPARAM` A `MulticoreParam` object or NULL. When the parameter parallelization = 'mcmapply' or 'pbmcmapply', this parameter must be NULL. When the parameter parallelization = 'bpmapply', this parameter must be one of the `MulticoreParam` object offered by the package 'BiocParallel. The default value is NULL.

Details

This function takes the result from `fit_marginal` as the input and and fit the copula model on the residuals.

Value

A list with the components:

`new_mvu` A matrix of the new multivariate uniform distribution from the copula.

`copula_list` A list of the fitted copula model. If using Gaussian copula, a list of correlation matrices; if vine, a list of vine objects.

`model_aic` A vector of the marginal AIC and the copula AIC.

`model_bic` A vector of the marginal BIC and the copula BIC.

Examples

```
data(example_sce)
my_data <- construct_data(
sce = example_sce,
assay_use = "counts",
celltype = "cell_type",
pseudotime = "pseudotime",
spatial = NULL,
other_covariates = NULL,
corr_by = "1"
)
my_marginal <- fit_marginal(
data = my_data,
mu_formula = "s(pseudotime, bs = 'cr', k = 10)",
sigma_formula = "1",
family_use = "nb",
n_cores = 1,
usebam = FALSE
)
my_copula <- fit_copula(
sce = example_sce,
assay_use = "counts",
marginal_list = my_marginal,
family_use = c(rep("nb", 5), rep("zip", 5)),
copula = "vine",
n_cores = 1,
input_data = my_data$dat
)
```

`fit_marginal` *Fit the marginal models*

Description

`fit_marginal` fits the per-feature regression models.

Usage

```
fit_marginal(
  data,
  predictor = "gene",
  mu_formula,
  sigma_formula,
  family_use,
  n_cores,
  usebam = FALSE,
  edf_flexible = FALSE,
  parallelization = "mcmapply",
  BPPARAM = NULL,
  trace = FALSE,
  simplify = FALSE,
  filter_cells = FALSE
)
```

Arguments

data	An object from construct_data.
predictor	A string of the predictor for the gam/gamlss model. Default is "gene". This is just a name.
mu_formula	A string of the mu parameter formula. It follows the format of formula in bam. Note: if the formula has multiple smoothers (s()) (we do not recommend this), please put the one with largest k (most complex one) as the first one.
sigma_formula	A string of the sigma parameter formula
family_use	A string or a vector of strings of the marginal distribution. Must be one of 'binomial', 'poisson', 'nb', 'zip', 'zinb' or 'gaussian', which represent 'poisson distribution', 'negative binomial distribution', 'zero-inflated poisson distribution', 'zero-inflated negative binomial distribution', and 'gaussian distribution' respectively.
n_cores	An integer. The number of cores to use.
usebam	A logic variable. If use bam for acceleration.
edf_flexible	A logic variable. It uses simpler model to accelerate the marginal fitting with a mild loss of accuracy. If TRUE, the fitted regression model will use the fitted relationship between Gini coefficient and the effective degrees of freedom on a random selected gene sets. Default is FALSE.
parallelization	A string indicating the specific parallelization function to use. Must be one of 'mcmapply', 'bpmapply', or 'pbmcmapply', which corresponds to the parallelization function in the package parallel,BiocParallel, and pbmcapply respectively. The default value is 'mcmapply'.
BPPARAM	A MulticoreParam object or NULL. When the parameter parallelization = 'mcmapply' or 'pbmcmapply', this parameter must be NULL. When the parameter parallelization = 'bpmapply', this parameter must be one of the MulticoreParam object offered by the package 'BiocParallel. The default value is NULL.

trace	A logic variable. If TRUE, the warning/error log and runtime for gam/gamlss will be returned. will be returned, FALSE otherwise. Default is FALSE.
simplify	A logic variable. If TRUE, the fitted regression model will only keep the essential contains for predict. Default is FALSE.
filter_cells	A logic variable. If TRUE, when all covariates used for fitting the GAM/GAMLSS model are categorical, the code will check each unique combination of categories and remove cells in that category if it has all zero gene expression for each fitted gene.

Details

The function takes the result from construct_data as the input, and fit the regression models for each feature based on users' specification.

Value

A list of fitted regression models. The length is equal to the total feature number.

Examples

```
data(example_sce)
my_data <- construct_data(
sce = example_sce,
assay_use = "counts",
celltype = "cell_type",
pseudotime = "pseudotime",
spatial = NULL,
other_covariates = NULL,
corr_by = "1"
)
my_marginal <- fit_marginal(
data = my_data,
mu_formula = "s(pseudotime, bs = 'cr', k = 10)",
sigma_formula = "1",
family_use = "nb",
n_cores = 1,
usebam = FALSE
)
```

---

ga *Functions from gamlss/gamlss.add with bugs fixed*

Description

An additive function to be used while fitting GAMLSS models. The interface for gam() in the **mgcv** package.

Usage

```
ga(formula, envir, control = ga.control(...), ...)
```

Arguments

formula	A formula of the model.
envir	The environment.
control	The control of the model fitting.
...	Other arguments.

Value

A xvar list.

ga

NA

Examples

```
print("No example")
```

---

gamlss.ba *Support for Function ba()*

Description

This is support for the smoother functions `ba()` interfaces for Simon Wood's `bam()` functions from package **mgcv**. It is not intended to be called directly by users. From `gamlss.add::gamlss.ba`.

Usage

```
gamlss.ba(x, y, w, xeval = NULL, ...)
```

Arguments

x	The explanatory variables
y	Iterative y variable
w	Iterative weights
xeval	If xeval=TRUE then prediction is used
...	Other arguments

Value

Not used

Examples

```
print("No example")
```

`gamlss.ga` *Support for Function ga()*

Description

This is support for the smoother functions `ga()` interfaces for Simon Wood's `gam()` functions from package **mgcv**. It is not intended to be called directly by users. From `gamlss.add::gamlss.ga`.

Usage

```
gamlss.ga(x, y, w, xeval = NULL, ...)
```

Arguments

`x`	The explanatory variables
`y`	Iterative y variable
`w`	Iterative weights
`xeval`	If xeval=TRUE then prediction is used
`...`	Other arguments

Value

Not used

Examples

```
print("No example")
```

`perform_lrt` *Perform the likelihood ratio test*

Description

`perform_lrt` performs the likelihood ratio test to compare two list of marginal models.

Usage

```
perform_lrt(alter_marginal, null_marginal)
```

Arguments

alter_marginal	A list of marginal models from the alternative hypothesis.
null_marginal	A list of marginal models from the null hypothesis. It must be strictly nested in the alternative model.

Details

The function takes two lists of marginal models (by default, the first list is the alternative and the second is the null) from fit_marginal. Note that LRT only makes sense for NESTED models. This can be quite tricky if you use penalized-splines (e.g., for trajectory data).

Value

A data.frame of the LRT result.

Examples

```
data(example_sce)
my_data <- construct_data(
sce = example_sce,
assay_use = "counts",
celltype = "cell_type",
pseudotime = "pseudotime",
spatial = NULL,
other_covariates = NULL,
corr_by = "cell_type"
)

my_data2 <- construct_data(
  sce = example_sce,
  assay_use = "counts",
  celltype = "cell_type",
  pseudotime = "pseudotime",
  spatial = NULL,
  other_covariates = NULL,
  corr_by = "pseudotime",
  ncell = 10000
)

my_marginal1 <- fit_marginal(
  data = my_data,
  mu_formula = "1",
  sigma_formula = "1",
  family_use = "nb",
  n_cores = 1,
  usebam = FALSE
)
my_marginal2 <- fit_marginal(
  data = my_data,
  mu_formula = "s(pseudotime, bs = 'cr', k = 10)",
  sigma_formula = "1",
```

```
  family_use = "nb",
  n_cores = 1,
  usebam = FALSE
)
my_fit1 <- lapply(my_marginal1, function(x)x$fit)
my_fit2 <- lapply(my_marginal2, function(x)x$fit)
my_pvalue <- perform_lrt(my_fit2, my_fit1)
```

`plot_reduceddim` *Dimensionality reduction and visualization*

Description

`plot_reduceddim` performs the dimensionality reduction

Usage

```
plot_reduceddim(
  ref_sce,
  sce_list,
  name_vec,
  assay_use = "logcounts",
  pc_umap = TRUE,
  n_pc = 50,
  center = TRUE,
  scale. = TRUE,
  if_plot = TRUE,
  shape_by = NULL,
  color_by,
  point_size = 1
)
```

Arguments

`ref_sce`	The reference sce.
`sce_list`	A list of synthetic sce.
`name_vec`	A vector of the names of each dataset. The length should be `length(sce_list) + 1`, where the first name is for `ref_sce`.
`assay_use`	A string which indicates the assay you will use in the sce. Default is 'logcounts'.
`pc_umap`	A logic value of whether using PCs as the input of UMAP. Default is TRUE.
`n_pc`	An integer of the number of PCs.
`center`	A logic value of whether centering the data before PCA. Default is TRUE.
`scale.`	A logic value of whether scaling the data before PCA. Default is TRUE.
`if_plot`	A logic value of whether returning the plot. If FALSE, return the reduced dimensions of each dataset.

shape_by	A string which indicates the column in colData used for shape.
color_by	A string which indicates the column in colData used for color.
point_size	A numeric value of the point size in the final plot. Default is 1.

Details

This function takes a reference sce and a list of new sces, performs the dimensionality reduction on the reference data, projects the synthetic datasets on the same low dimensional space, then visualize the results.

Value

The ggplot or the data.frame of reduced dimensions.

---

pseudotime *A numeric vector of pseudotime values*

Description

A numeric vector of pseudotime values

Usage

```
data("pseudotime")
```

Format

A numeric vector with 2087 values

Value

The corresponding pseudotime of the 'example_count'

---

scdesign3 *The wrapper for the whole scDesign3 pipeline*

Description

scdesign3 takes the input data, fits the model and

Usage

```
scdesign3(
  sce,
  assay_use = "counts",
  celltype,
  pseudotime = NULL,
  spatial = NULL,
  other_covariates,
  ncell = dim(sce)[2],
  mu_formula,
  sigma_formula = "1",
  family_use = "nb",
  n_cores = 2,
  correlation_function = "default",
  usebam = FALSE,
  edf_flexible = FALSE,
  corr_formula,
  empirical_quantile = FALSE,
  copula = "gaussian",
  if_sparse = FALSE,
  fastmvn = FALSE,
  DT = TRUE,
  pseudo_obs = FALSE,
  family_set = c("gauss", "indep"),
  important_feature = "all",
  nonnegative = TRUE,
  nonzerovar = FALSE,
  return_model = FALSE,
  simplify = FALSE,
  parallelization = "mcmapply",
  n_rep = 1,
  BPPARAM = NULL,
  trace = FALSE
)
```

Arguments

sce	A SingleCellExperiment object.
assay_use	A string which indicates the assay you will use in the sce. Default is 'counts'.
celltype	A string of the name of cell type variable in the colData of the sce. Default is 'cell_type'.
pseudotime	A string or a string vector of the name of pseudotime and (if exist) multiple lineages. Default is NULL.
spatial	A length two string vector of the names of spatial coordinates. Default is NULL.
other_covariates	A string or a string vector of the other covariates you want to include in the data.

`ncell` The number of cell you want to simulate. Default is `dim(sce)[2]` (the same number as the input data).

`mu_formula` A string of the mu parameter formula

`sigma_formula` A string of the sigma parameter formula

`family_use` A string of the marginal distribution. Must be one of 'poisson', 'nb', 'zip', 'zinb' or 'gaussian'.

`n_cores` An integer. The number of cores to use.

`correlation_function`
A string. If 'default', the function from `Rfast`; if 'coop', the function from `coop`, which calls BLAS.

`usebam` A logic variable. If use `bam` for acceleration in marginal fitting.

`edf_flexible` A logic variable. It is used for accelerating for spatial model if k is large in 'mu_formula'. Default is FALSE.

`corr_formula` A string of the correlation structure.

`empirical_quantile`
Please only use it if you clearly know what will happen! A logic variable. If TRUE, DO NOT fit the copula and use the EMPIRICAL CDF values of the original data; it will make the simulated data fixed (no randomness). Default is FALSE. Only works if ncell is the same as your original data.

`copula` A string of the copula choice. Must be one of 'gaussian' or 'vine'. Default is 'gaussian'. Note that vine copula may have better modeling of high-dimensions, but can be very slow when features are >1000.

`if_sparse` A logic variable. Only works for Gaussian copula (`family_set = "gaussian"`). If TRUE, a thresholding strategy will make the corr matrix sparse.

`fastmvn` An logical variable. If TRUE, the sampling of multivariate Gaussian is done by `mvnfast`, otherwise by `mvtnorm`. Default is FALSE. It only matters for Gaussian copula.

`DT` A logic variable. If TRUE, perform the distributional transformation to make the discrete data 'continuous'. This is useful for discrete distributions (e.g., Poisson, NB). Default is TRUE. Note that for continuous data (e.g., Gaussian), DT does not make sense and should be set as FALSE.

`pseudo_obs` A logic variable. If TRUE, use the empirical quantiles instead of theoretical quantiles for fitting copula. Default is FALSE.

`family_set` A string or a string vector of the bivariate copula families. Default is c("gauss", "indep"). For more information please check package `rvinecoplib`.

`important_feature`
A numeric value or vector which indicates whether a gene will be used in correlation estimation or not. If this is a numeric value, then gene with zero proportion greater than this value will be excluded form gene-gene correlation estimation. If this is a vector, then this should be a logical vector with length equal to the number of genes in `sce`. `TRUE` in the logical vector means the corresponding gene will be included in gene-gene correlation estimation and `FALSE` in the logical vector means the corresponding gene will be excluded from the gene-gene correlation estimation. The default value is "all" (a special string which means no filtering).

nonnegative A logical variable. If TRUE, values < 0 in the synthetic data will be converted to 0. Default is TRUE (since the expression matrix is nonnegative).

nonzerovar A logical variable. If TRUE, for any gene with zero variance, a cell will be replaced with 1. This is designed for avoiding potential errors, for example, PCA. Default is FALSE.

return_model A logic variable. If TRUE, the marginal models and copula models will be returned. Default is FALSE.

simplify A logic variable. If TRUE, the fitted regression model will only keep the essential contains for predict, otherwise the fitted models can be VERY large. Default is FALSE.

parallelization
A string indicating the specific parallelization function to use.

n_rep An integer number. The number of replicates of simulated new count matrix. Default is 1. Must be one of 'mcmapply', 'bpmapply', or 'pbmcmapply', which corresponds to the parallelization function in the package parallel,BiocParallel, and pbmcapply respectively. The default value is 'mcmapply'.

BPPARAM A MulticoreParam object or NULL. When the parameter parallelization = 'mcmapply' or 'pbmcmapply', this parameter must be NULL. When the parameter parallelization = 'bpmapply', this parameter must be one of the MulticoreParam object offered by the package 'BiocParallel. The default value is NULL.

trace A logic variable. If TRUE, the warning/error log and runtime for gam/gamlss will be returned, FALSE otherwise. Default is FALSE.

Value

A list with the components:

new_count A matrix of the new simulated count (expression) matrix.

new_covariate A data.frame of the new covariate matrix.

model_aic The model AIC.

marginal_list A list of marginal regression models if return_model = TRUE.

corr_list A list of correlation models (conditional copulas) if return_model = TRUE.

Examples

```
data(example_sce)
my_simu <- scdesign3(
sce = example_sce,
assay_use = "counts",
celltype = "cell_type",
pseudotime = "pseudotime",
spatial = NULL,
other_covariates = NULL,
mu_formula = "s(pseudotime, bs = 'cr', k = 10)",
sigma_formula = "1",
family_use = "nb",
n_cores = 2,
```

```
usebam = FALSE,
edf_flexible = FALSE,
corr_formula = "pseudotime",
copula = "gaussian",
if_sparse = TRUE,
DT = TRUE,
pseudo_obs = FALSE,
ncell = 1000,
return_model = FALSE
)
```

---

simu_new *Simulate new data*

Description

simu_new generates new simulated data based on fitted marginal and copula models.

Usage

```
simu_new(
  sce,
  assay_use = "counts",
  mean_mat,
  sigma_mat,
  zero_mat,
  quantile_mat = NULL,
  copula_list,
  n_cores,
  fastmvn = FALSE,
  family_use,
  nonnegative = TRUE,
  nonzerovar = FALSE,
  input_data,
  new_covariate,
  important_feature = "all",
  parallelization = "mcmapply",
  BPPARAM = NULL,
  filtered_gene
)
```

Arguments

sce	A SingleCellExperiment object.
assay_use	A string which indicates the assay you will use in the sce. Default is 'counts'.
mean_mat	A cell by feature matrix of the mean parameter.

sigma_mat	A cell by feature matrix of the sigma parameter.
zero_mat	A cell by feature matrix of the zero-inflation parameter.
quantile_mat	A cell by feature matrix of the multivariate quantile.
copula_list	A list of copulas for generating the multivariate quantile matrix. If provided, the quantile_mat must be NULL.
n_cores	An integer. The number of cores to use.
fastmvn	An logical variable. If TRUE, the sampling of multivariate Gaussian is done by mvnfast, otherwise by mvtnorm. Default is FALSE.
family_use	A string of the marginal distribution. Must be one of 'poisson', "binomial", 'nb', 'zip', 'zinb' or 'gaussian'.
nonnegative	A logical variable. If TRUE, values < 0 in the synthetic data will be converted to 0. Default is TRUE (since the expression matrix is nonnegative).
nonzerovar	A logical variable. If TRUE, for any gene with zero variance, a cell will be replaced with 1. This is designed for avoiding potential errors, for example, PCA.
input_data	A input count matrix.
new_covariate	A data.frame which contains covariates of targeted simulated data from construct_data.
important_feature	important_feature A string or vector which indicates whether a gene will be used in correlation estimation or not. If this is a string, then this string must be either "all" (using all genes) or "auto", which indicates that the genes will be automatically selected based on the proportion of zero expression across cells for each gene. Gene with zero proportion greater than 0.8 will be excluded form gene-gene correlation estimation. If this is a vector, then this should be a logical vector with length equal to the number of genes in sce. TRUE in the logical vector means the corresponding gene will be included in gene-gene correlation estimation and FALSE in the logical vector means the corresponding gene will be excluded from the gene-gene correlation estimation. The default value for is "all".
parallelization	A string indicating the specific parallelization function to use. Must be one of 'mcmapply', 'bpmapply', or 'pbmcmapply', which corresponds to the parallelization function in the package parallel,BiocParallel, and pbmcapply respectively. The default value is 'mcmapply'.
BPPARAM	A MulticoreParam object or NULL. When the parameter parallelization = 'mcmapply' or 'pbmcmapply', this parameter must be NULL. When the parameter parallelization = 'bpmapply', this parameter must be one of the MulticoreParam object offered by the package 'BiocParallel. The default value is NULL.
filtered_gene	A vector or NULL which contains genes that are excluded in the marginal and copula fitting steps because these genes only express in less than two cells. This can be obtain from construct_data

Details

The function takes the new covariate (if use) from construct_data, parameter matrices from extract_para and multivariate Unifs from fit_copula.

Value

A feature by cell matrix of the new simulated count (expression) matrix or sparse matrix.

Examples

```
data(example_sce)
my_data <- construct_data(
sce = example_sce,
assay_use = "counts",
celltype = "cell_type",
pseudotime = "pseudotime",
spatial = NULL,
other_covariates = NULL,
corr_by = "1"
)
my_marginal <- fit_marginal(
data = my_data,
mu_formula = "s(pseudotime, bs = 'cr', k = 10)",
sigma_formula = "1",
family_use = "nb",
n_cores = 1,
usebam = FALSE
)
my_copula <- fit_copula(
sce = example_sce,
assay_use = "counts",
marginal_list = my_marginal,
family_use = c(rep("nb", 5), rep("zip", 5)),
copula = "vine",
n_cores = 1,
input_data = my_data$dat
)
my_para <- extract_para(
  sce = example_sce,
  marginal_list = my_marginal,
  n_cores = 1,
  family_use = c(rep("nb", 5), rep("zip", 5)),
  new_covariate = my_data$new_covariate,
  data = my_data$dat
)
my_newcount <- simu_new(
sce = example_sce,
mean_mat = my_para$mean_mat,
sigma_mat = my_para$sigma_mat,
zero_mat = my_para$zero_mat,
quantile_mat = NULL,
copula_list = my_copula$copula_list,
n_cores = 1,
family_use = c(rep("nb", 5), rep("zip", 5)),
input_data = my_data$dat,
new_covariate = my_data$new_covariate,
important_feature = my_copula$important_feature,
```

```
  filtered_gene = my_data$filtered_gene
  )
```

`sparse_cov` *This function computes the thresholding sparse covariance/correlation estimator with the optimal threshold level.*

Description

Part from Chenxin Jiang

Usage

```
sparse_cov(
  data,
  method = c("cv", "qiu"),
  operator = c("hard", "soft", "scad", "al"),
  corr = TRUE
)
```

Arguments

`data`	The data matrix.
`method`	The choice of method to select the optimal threshold level.
`operator`	The choice of the thresholding operator.
`corr`	The indicator of computing correlation or covariance matrix.

Value

The thresholding sparse covariance/correlation estimator.

Examples

```
print("No example")
```

Index

∗ **datasets**
example_count, 5
example_sce, 5
pseudotime, 17

ba, 2
bam, *11*, *19*

construct_data, 3, *6–8*, *11, 12*, *22*

example_count, 5
example_sce, 5
extract_para, 6, *22*

fit_copula, 8, *22*
fit_marginal, *3*, *6–9*, 10, *15*

ga, 12
gamlss.ba, 13
gamlss.ga, 14

perform_lrt, 14
plot_reduceddim, 16
pseudotime, 17

scdesign3, 17
simu_new, 21
sparse_cov, 24