Package 'SCdeconR'

March 22, 2024

Type Package

Title Deconvolution of Bulk RNA-Seq Data using Single-Cell RNA-Seq Data as Reference

Version 1.0.0

Description Streamlined workflow from deconvolution of bulk RNA-seq data to downstream differential expression and gene-set enrichment analysis. Provide various visualization functions.

License GPL (>= 3)

URL https://github.com/Liuy12/SCdeconR/, https://liuy12.github.io/SCdeconR/

Encoding UTF-8

RoxygenNote 7.3.1

Imports data.table, gtools, harmony, MASS, Matrix, NMF, reshape2, Biobase, mgcv, grDevices, glmGamPoi

Suggests FARDEEP, reticulate, plotly, DESeq2, knitr, limma, nnls, preprocessCore, rmarkdown, scater, scran, SingleCellExperiment, doFuture, future, sctransform, Linnorm

Depends dplyr, edgeR, foreach, ggplot2, R (>= 2.10), Seurat

NeedsCompilation no

Author Yuanhang Liu [aut, cre]

Maintainer Yuanhang Liu <liu.yuanhang@mayo.edu>

Repository CRAN

Date/Publication 2024-03-22 19:20:02 UTC

R topics documented:

bulk_generator 2
celltype_expression 4
comparedeg_scatter 5
comparegsea_scatter 6

compute_metrics 7
construct_ref 8
gsea_heatmap 10
gsea_rwplot 11
gsea_sumplot 12
load_scdata 13
prepare_rnk 14
prop_barplot 15
reformat_gmt 15
run_de 16
scaling 19
scdecon 20
transformation 25

Index **26**

`bulk_generator` *Generate artificial bulk RNA-seq samples based on simulation*

Description

Generate artificial bulk RNA-seq samples with random or pre-defined cell-type proportions for benchmarking deconvolution algorithms

Usage

```
bulk_generator(
  ref,
  phenodata,
  num_mixtures = 500,
  num_mixtures_sprop = 10,
  pool_size = 100,
  seed = 1234,
  prop = NULL,
  replace = FALSE
)
```

Arguments

`ref`	a matrix-like object of gene expression values with rows representing genes, columns representing cells.
`phenodata`	a data.frame with rows representing cells, columns representing cell attributes. It should at least contain the first two columns as: 1. cell barcodes 2. cell types
`num_mixtures`	total number of simulated bulk samples. Have to be multiple of `num_mixtures_sprop`. Default to 500.

num_mixtures_sprop
: number of simulated bulk samples with the same simulated cell type proportions. Only applicable when prop is not specified. Those samples will be used to estimate bias & variance. Default to 10.

pool_size
: number of cells to use to construct each artificial bulk sample. Default to 100.

seed
: seed to use for simulation. Default to 1234.

prop
: a data.frame with two columns. The first column includes unique cell types in phenodata; the second column includes cell type proportions. If specified, bulk samples will be simulated based on the specified cell proportions.

replace
: logical value indicating whether to sample cells with replacement. Default to FALSE, to sample cells without replacement.

Details

If prop is not specified, cell type proportions will be firstly randomly generated with at least two cell types present. Then, for each cell proportion vector, num_mixtures_sprop number of samples is simulated. Eventually, a total of num_mixtures number of samples is simulated. If prop is specified, then a total of num_mixtures number of samples will be simulated based on the same cell proportion vector specified.

Value

a list of two objects:

1. simulated bulk RNA-seq data, with rows representing genes, columns representing samples
2. cell type proportions used to simulate the bulk RNA-seq data, with rows representing cell types, columns representing samples

Examples

```
ref_list <- c(paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample1"),
              paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample2"))
phenopath1 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample1.txt")
phenopath2 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample2.txt")
phenodata_list <- c(phenopath1,phenopath2)

# construct integrated reference using harmony algorithm
refdata <- construct_ref(ref_list = ref_list,
                      phenodata_list = phenodata_list,
                      data_type = "cellranger",
                      method = "harmony",
                      group_var = "subjectid",
                      nfeature_rna = 50,
                      vars_to_regress = "percent_mt", verbose = FALSE)
phenodata <- data.frame(cellid = colnames(refdata),
                        celltypes = refdata$celltype,
                        subjectid = refdata$subjectid)
```

```
prop <- data.frame(celltypes = unique(refdata$celltype),
proportion = rep(1/length(unique(refdata$celltype)), length(unique(refdata$celltype))))
bulk_sim <- bulk_generator(ref = GetAssayData(refdata, slot = "data", assay = "SCT"),
                           phenodata = phenodata,
                           num_mixtures = 20,
                           prop = prop,
                           num_mixtures_sprop = 1)
```

`celltype_expression` *Compute cell type specific gene expression*

Description

Compute cell type specific gene expression based on predicted cell proportions and reference data.

Usage

```
celltype_expression(bulk, ref, phenodata, prop, ...)
```

Arguments

`bulk`	a matrix-like object of bulk RNA-seq data with rows representing genes, columns representing samples
`ref`	a matrix-like object of scRNA-seq data with rows representing genes, columns representing cells.
`phenodata`	a data.frame with rows representing cells, columns representing cell attributes. It should at least contain the first two columns as: 1. cell barcodes 2. cell types
`prop`	a matrix-like object of cell proportion values with rows representing cell types, columns representing samples.
`...`	additional parameters passed to `create.RCTD` from `spacexr`.

Details

this function is inspired by cell-type specific gene expression estimation for doublet mode in `spacexr`. See examples from `run_de`.

Value

a list with length equal to number of unique cell types in phenodata. Each element in the list represents gene expression matrix for each unique cell type.

`comparedeg_scatter` *Generate a scatter plot comparing two differential expression results*

Description

Generate a scatter plot of fold changes comparing two differential expression results, e.g. w/wo adjusting for cell proportion differences.

Usage

```
comparedeg_scatter(
  results1,
  results2,
  result_names = NULL,
  fc_cutoff,
  pval_cutoff,
  pvalflag = TRUE,
  interactive = FALSE
)
```

Arguments

`results1`	a data.frame containing differential expression results with five columns: "Gene name", "log2 fold change", "log2 average expression", "p value", "adjusted p value". The second element of the output from function `run_de`.
`results2`	similar to `results1`.
`result_names`	a vector of length 2 indicating the names of the two differential results. If NULL, names will be set to c("results1", "results2")
`fc_cutoff`	fold change cutoff to identify differential expressed genes.
`pval_cutoff`	p value cutoff to identify differential expressed genes.
`pvalflag`	a logical value indicating whether to use adjusted p value in selecting differential expressed genes.
`interactive`	a logical value indicating whether to generate an interactive plot.

Details

See examples from `run_de`.

Value

a `ggplot` object or `plotly` object if interactive is set to TRUE

`comparegsea_scatter` *Generate a scatter plot comparing two gene set enrichment analysis results*

Description

Generate a scatter plot of normalized enrichment scores comparing two results for gene set enrichment analysis, e.g. w/wo adjusting for cell proportion differences.

Usage

```
comparegsea_scatter(
  gseares_path1,
  gseares_path2,
  result_names = NULL,
  nes_cutoff = 2,
  pval_cutoff = 0.1,
  pvalflag = TRUE,
  interactive = FALSE
)
```

Arguments

`gseares_path1`	path to GSEA output.
`gseares_path2`	path to a second GSEA output.
`result_names`	a vector of length 2 indicating the names of the two GSEA results. If NULL, names will be set to c("results1", "results2")
`nes_cutoff`	normalized enrichment score cutoff to identify enriched gene-sets.
`pval_cutoff`	p value cutoff to identify enriched gene-sets.
`pvalflag`	a logical value indicating whether to use adjusted p value in selecting enriched gene-sets. Default to TRUE.
`interactive`	a logical value indicating whether to generate an interactive plot. Default to FALSE.

Details

this function does not support output from GSEA R implementation

Value

a `ggplot` object or `plotly` object if interactive is set to TRUE

compute_metrics *Statistical evaluations of predicted cell proportions*

Description

Compute RMSE, bias & variance metrics for predicted cell proportions by comparing with expected cell proportions.

Usage

```
compute_metrics(prop_pred, prop_sim)
```

Arguments

prop_pred	a matrix-like object of predicted cell proportion values with rows representing cell types, columns representing samples.
prop_sim	a matrix-like object of simulated/expected cell proportion values with rows representing cell types, columns representing samples.

Value

a list of two objects:

1. a data.fame of summary metrics containing RMSE, bias & variance grouped by cell types and mixture ids (simulated samples with the same expected cell proportions).
2. a data.frame of aggregated RMSE values across all cell types within each sample.

Examples

```
## generate artificial bulk samples
ref_list <- c(paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample1"),
              paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample2"))
phenopath1 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample1.txt")
phenopath2 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample2.txt")
phenodata_list <- c(phenopath1,phenopath2)

# construct integrated reference using harmony algorithm
refdata <- construct_ref(ref_list = ref_list,
                      phenodata_list = phenodata_list,
                      data_type = "cellranger",
                      method = "harmony",
                      group_var = "subjectid",
                      nfeature_rna = 50,
                      vars_to_regress = "percent_mt", verbose = FALSE)
phenodata <- data.frame(cellid = colnames(refdata),
                        celltypes = refdata$celltype,
```

```
                        subjectid = refdata$subjectid)
prop <- data.frame(celltypes = unique(refdata$celltype),
proportion = rep(1/length(unique(refdata$celltype)), length(unique(refdata$celltype))))
bulk_sim <- bulk_generator(ref = GetAssayData(refdata, slot = "data", assay = "SCT"),
                           phenodata = phenodata,
                           num_mixtures = 20,
                           prop = prop,
                           num_mixtures_sprop = 1)

## perform deconvolution based on "OLS" algorithm
decon_res <- scdecon(bulk = bulk_sim[[1]],
                     ref = GetAssayData(refdata, slot = "data", assay = "SCT"),
                     phenodata = phenodata,
                     filter_ref = TRUE,
                     decon_method = "OLS",
                     norm_method_sc = "LogNormalize",
                     norm_method_bulk = "TMM",
                     trans_method_sc = "none",
                     trans_method_bulk = "log2",
                     marker_strategy = "all")

## compute metrics
metrics_res <- compute_metrics(decon_res[[1]], bulk_sim[[2]])
```

---

construct_ref *Integration of single-cell/nuclei RNA-seq data as reference*

Description

Integration of scRNA-seq or snRNA-seq data using either harmony or seurat.

Usage

```
construct_ref(
  ref_list,
  phenodata_list,
  data_type = c("cellranger", "h5", "matrix"),
  method = c("harmony", "seurat"),
  group_var,
  nfeature_rna = 200,
  percent_mt = 40,
  vars_to_regress = c("percent_mt", "phase"),
  ex_features = NULL,
  cluster = TRUE,
  resolution = 0.8,
  verbose = TRUE,
  ...
)
```

Arguments

ref_list	a character vector of data paths to scRNA-seq/snRNA-seq. See data_type for accepted data types.
phenodata_list	a character vector of data paths to metadata for elements in ref_list. All metadata within phenodata_list should have consistent column names. Columns represent cell attributes, such as cell type, rows represent cells. Each element in phenodata_list should at least contain the first two columns as: 1. cell barcodes 2. cell types
data_type	data type of the input scRNA-seq/snRNA-seq data. Could be either a single character value from "cellranger", "h5", "matrix", or a vector/list of values with the same length as ref_list indicating the data type for each element.
method	character value specifying the method to use. Has to be one of "harmony" or "seurat". See details for more information.
group_var	a vector of character values indicating which variables within phenodata_list metadata to use for integration. Only applicable when method is set to "harmony".
nfeature_rna	minimum # of features with non-zero UMIs. Cells with # of features lower than nfeature_rna will be removed. Default to 200.
percent_mt	maximum percentage of mitochondria (MT) mapped UMIs. Cells with MT percentage higher than percent_mt will be removed. Default to 40.
vars_to_regress	a list of character values indicating the variables to regress for SCTransform normalization step. Default is to regress out MT percentage ("percent_mt") & cell cycle effects ("phase")
ex_features	a vector of character values indicating genes to exclude from anchor features. Those genes will not be considered as anchor features for integration, but will still be present in the integrated data.
cluster	logical value indicating whether to perform clustering on the integrated data. If TRUE, unsupervised clustering will be performed, and the results will be saved in "seurat_clusters" metadata in the output Seurat object.
resolution	numeric value specifying resolution to use when cluster is set to TRUE.
verbose	logical value indicating whether to print messages.
...	additional parameters passed to SCTransform.

Details

data_type can be chosen from:

cellranger path to a directory containing the matrix.mtx, genes.tsv (or features.tsv), and barcodes.tsv files outputted by 10x's cell-ranger

h5 path to .h5 file outputted by 10x's cell-ranger

matrix path to a matrix-like file, with rows representing genes, columns representing cells.

SCTransform with vst.flavor = "v2" is used for normalization of individual data. Integration methods can be chosen from either "harmony" or "seurat". Harmony typically is more memory efficient and, recommended if you have large # of cells for integration.

Value

a Seurat-class object.

Examples

```
## random subset of two scRNA-seq datasets for breast tissue
ref_list <- c(paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample1"),
              paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample2"))
phenopath1 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample1.txt")
phenopath2 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample2.txt")
phenodata_list <- c(phenopath1,phenopath2)

## Register backend for parallel processing
#doFuture::registerDoFuture()
#future::plan("multisession", workers = 4)

## construct integrated reference data
refdata <- construct_ref(ref_list = ref_list,
                         phenodata_list = phenodata_list,
                         data_type = "cellranger",
                         method = "harmony",
                         group_var = "subjectid",
                         nfeature_rna = 50,
                         vars_to_regress = "percent_mt")
```

---

gsea_heatmap *Heatmap to demonstrate enrichment of selected gene-sets*

Description

Heatmap to demonstrate enrichment of selected gene-sets.

Usage

```
gsea_heatmap(
  normdata,
  teststats,
  gmtfile,
  numgenes,
  gsname_up,
  gsname_down,
  anncol,
  color,
  anncolors = NULL,
  rankcol = TRUE,
```

```
  zscore_range = c(-3, 3)
)
```

Arguments

normdata	a matrix-like object of normalized & untransformed bulk RNA-seq data, with rows representing genes and columns representing samples. The first element of the output from function run_de.
teststats	a data.frame containing differential expression results with five columns: "Gene name", "log2 fold change", "log2 average expression", "p value", "adjusted p value". The second element of the output from function run_de.
gmtfile	path to gmt file used for GSEA analysis.
numgenes	Number of genes to include in the heatmap. Will choose numgenes # of top up-regulated genes, as well as numgenes # of down-regulated genes
gsname_up	a character value indicating selected up-regulated gene-set.
gsname_down	a character value indicating selected down-regulated gene-set.
anncol	a data.frame of sample meta information to include as column annotation bars. See option annCol from aheatmap for more details.
color	color used for heatmap. See option color option from aheatmap for details.
anncolors	optional data.frame to define colors for column annotations in anncol.
rankcol	a logical value indicating whether to sort samples based on correlation between fold change & gene expression for better visualization. Default to TRUE.
zscore_range	a vector of length two indicating the desired range of z-score transformed data. Default to c(-3, 3).

Details

this function does not support output from GSEA R implementation

Value

a heatmap outputted from aheatmap function from NMF package

---

gsea_rwplot *GSEA random-walk plot*

Description

Generate high-quality GSEA random-walk figures.

Usage

```
gsea_rwplot(gseares_path, gsname, class_name, metric_range = NULL)
```

Arguments

gseares_path	path to GSEA output.
gsname	a character value indicating the gene-set name to generate random-walk plot.
class_name	a character value indicating the class of the gene-set, e.g. "GO".
metric_range	optional range of the ranking metric.

Details

this function does not support output from GSEA R implementation. Scripts initially implemented by Thomas Kuilman.

Value

a random-walk plot for selected gene-set

---

gsea_sumplot *Summary plot of gene set enrichment analysis*

Description

Summary plot of selected up/down regulated gene-sets for gene set enrichment analysis.

Usage

```
gsea_sumplot(
  gseares_path,
  pos_sel,
  neg_sel,
  pvalflag = TRUE,
  interactive = FALSE
)
```

Arguments

gseares_path	path to GSEA output.
pos_sel	a character vector of upregulated gene-set names.
neg_sel	a character vector of downregulated gene-set names.
pvalflag	a logical value indicating whether to use adjusted p value in the plot. Default to TRUE.
interactive	a logical value indicating whether to generate an interactive plot. Default to FALSE.

Details

this function does not support output from GSEA R implementation

Value

a ggplot object or plotly object if interactive is set to TRUE

---

load_scdata *Load, filter and normalize scRNA-seq/snRNA-seq data*

Description

Load and preprocess scRNA-seq/snRNA-seq data using seurat SCTransform workflow.

Usage

```
load_scdata(
  ref,
  data_type = c("cellranger", "h5", "matrix"),
  meta_info,
  nfeature_rna = 200,
  percent_mt = 40,
  cc.genes = NULL,
  vars_to_regress = c("percent_mt", "phase"),
  id,
  verbose,
  ...
)
```

Arguments

ref	path to scRNA-seq/snRNA-seq data.
data_type	a character value specifying data type of the input scRNA-seq/snRNA-seq data, should be one of "cellranger", "h5", "matrix".
meta_info	a data.frame with rows representing cells, columns representing cell attributes.
nfeature_rna	minimum # of features with non-zero UMIs. Cells with # of features lower than nfeature_rna will be removed. Default to 200.
percent_mt	maximum percentage of mitochondria (MT) mapped UMIs. Cells with MT percentage higher than percent_mt will be removed. Default to 40.
cc.genes	cell-cycle genes curated by Seurat. Can be loaded via data(cc.genes)
vars_to_regress	a list of character values indicating the variables to regress for SCTransform normalization step. Default is to regress out MT percentage ("percent_mt") & cell cycle effects ("phase")
id	a character value specifying project or sample id. Only used for printing purposes.
verbose	logical value indicating whether to print messages.
...	additional parameters passed to SCTransform.

Details

For more details, refer to construct_ref

Value

a Seurat-class object.

Examples

```
samplepath1 <- paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample1")
samplepath2 <- paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample2")
ref_list <- c(samplepath1, samplepath2)
phenopath1 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample1.txt")
phenopath2 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample2.txt")
phenodata_list <- c(phenopath1,phenopath2)
tmp <- load_scdata(
  ref = ref_list[[1]],
  data_type = c("cellranger"),
 meta_info = data.table::fread(file = phenodata_list[[1]], check.names = FALSE, header = TRUE),
  nfeature_rna = 50,
  vars_to_regress = c("percent_mt"),
  id = 1,
  verbose = TRUE)
```

---

prepare_rnk *Prepare .rnk file for GSEA preranked analysis*

Description

Prepare .rnk file for GSEA preranked analysis

Usage

```
prepare_rnk(teststats, outputfile, replace = FALSE)
```

Arguments

teststats	a data.frame containing differential expression results with five columns: "Gene name", "log2 fold change", "log2 average expression", "p value", "adjusted p value". The second element of the output from function run_de.
outputfile	full path including file name to .rnk file.
replace	a logical value indicating whether to replace the output file if it already exists. Default to FALSE.

Value

No return value

`prop_barplot` *Bar plot of cell type proportions across samples*

Description

Bar plot of cell type proportions across samples

Usage

```
prop_barplot(prop, sort = TRUE, interactive = FALSE)
```

Arguments

`prop`	a matrix or data.frame of cell proportion values with rows representing cell types, columns representing samples.
`sort`	a logical value indicating whether to sort the samples based on cell type with highest median cell proportion across samples. Default to TRUE.
`interactive`	a logical value indicating whether to generate interactive plot. Default to FALSE.

Value

a `ggplot` object or `plotly` object if interactive is set to TRUE

`reformat_gmt` *Methods to manipulate .gmt files*

Description

Reformat, read & write .gmt file.

Usage

```
reformat_gmt(gmtfile, outputfile, replace = FALSE)

read_gmt(gmtfile)

write_gmt(gmt, outputfile, replace = FALSE)
```

Arguments

gmtfile	path to a gene set definition file in .gmt format.
outputfile	full path including file name to export reformatted .gmt file.
replace	a logical value indicating whether to replace the output file if it already exists. Default to FALSE.
gmt	a gmt object returned by read_gmt.

Details

reformat_gmt replaces blank spaces within the gene-set names to help string-matching methods in downstream plot functions gsea_sumplot ,gsea_rwplot, gsea_heatmap.

Value

for read_gmt, returns a list object with length equal to the total number of gene sets within the .gmt file. Each list contains three elements: "id", "name", "genes". read_gmt & write_gmt are reimplemented based on functions from ActivePathways package.

---

run_de *Differential expression analysis*

Description

Performing differential expression analysis adjusting for cell proportion differences, and other covariates using a additive model.

Usage

```
run_de(
  bulk,
  prop = NULL,
  sampleinfo,
  control = NULL,
  case = NULL,
  de_method = c("edgeR", "DESeq2", "limma_voom", "limma"),
  padj_method = "BH",
  ...
)
```

Arguments

bulk	a matrix-like object of gene expression values with rows representing genes, columns representing samples
prop	a matrix-like object of cell proportion values with rows representing cell types, columns representing samples. Default to NULL, not adjust for cell proportion.

sampleinfo	a data.frame of metadata for the samples. Rows represents samples; columns represents covariates to adjust for. The first column of sampleinfo should contains group information for differential analysis.
control	a character value indicating the control group in sampleinfo. Set to NULL to perform ANOVA-like analysis.
case	a character value indicating the case group in sampleinfo. Set to NULL to perform ANOVA-like analysis.
de_method	a character value indicating the method to use for testing differential expression. Should be one of "edgeR", "DESeq2", "limma_voom", "limma"
padj_method	method for adjusting multiple hypothesis testing. Default to "BH". See p.adjust for more details.
...	parameters pass to DE methods.

Details

To perform ANOVA like analysis (differences between any groups), set control & case options to NULL and choose one of the following methods: edgeR, limma_voom or limma. DESeq2 does not provide direct support for this type of comparison.

Value

a list of two elements:

1. a data.frame containing normalized gene expression data.
2. a data.frame containing detailed differential expression statistics. Columns represent "Gene name", "log2 fold change", "log2 average expression", "p value", "adjusted p value" respectively

Examples

```
ref_list <- c(paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample1"),
              paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample2"))
phenopath1 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample1.txt")
phenopath2 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample2.txt")
phenodata_list <- c(phenopath1,phenopath2)

# construct integrated reference using harmony algorithm
refdata <- construct_ref(ref_list = ref_list,
                      phenodata_list = phenodata_list,
                      data_type = "cellranger",
                      method = "harmony",
                      group_var = "subjectid",
                      nfeature_rna = 50,
                      vars_to_regress = "percent_mt", verbose = FALSE)
phenodata <- data.frame(cellid = colnames(refdata),
                        celltypes = refdata$celltype,
```

```
                        subjectid = refdata$subjectid)
## construct a vector with same proportions across cell types
prop1 <- data.frame(celltypes = unique(refdata$celltype),
                    proportion = rep(0.125, 8))
## simulate 20 bulk samples based on specified cell type proportion
bulk_sim1 <- bulk_generator(ref = GetAssayData(refdata, slot = "data", assay = "SCT"),
                            phenodata = phenodata,
                            num_mixtures = 20,
                            prop = prop1, replace = TRUE)
## generate another vector with high proportion for a certian cell type
prop2 <- data.frame(celltypes = unique(refdata$celltype),
                    proportion = c(0.8, 0.1, 0.1, rep(0, 5)))
bulk_sim2 <- bulk_generator(ref = GetAssayData(refdata, slot = "data", assay = "SCT"),
                            phenodata = phenodata,
                            num_mixtures = 20,
                            prop = prop2, replace = TRUE)
## compare data for differential analysis
bulk_sim <- list(cbind(bulk_sim1[[1]], bulk_sim2[[1]]),
                 cbind(bulk_sim1[[2]], bulk_sim2[[2]]))
## force to be integer for DE purposes
bulk <- round(bulk_sim[[1]], digits=0)
colnames(bulk) <- paste0("sample", 1:ncol(bulk))
## predict cell type proportions using "OLS" algorithm
decon_res <- scdecon(bulk = bulk,
                     ref = GetAssayData(refdata, slot = "data", assay = "SCT"),
                     phenodata = phenodata,
                     filter_ref = TRUE,
                     decon_method = "OLS",
                     norm_method_sc = "LogNormalize",
                     norm_method_bulk = "TMM",
                     trans_method_sc = "none",
                     trans_method_bulk = "log2",
                     marker_strategy = "all")
## create sampleinfo
sampleinfo <- data.frame(condition = rep(c("group1", "group2"), each =20))
row.names(sampleinfo) <- colnames(bulk)
deres <- run_de(bulk = bulk,
                prop = decon_res[[1]],
                sampleinfo = sampleinfo,
                control = "group1",
                case = "group2",
                de_method = "edgeR")

 ## run differential analysis without adjusting for cell proportion differences
 deres_notadjust <- run_de(bulk = bulk,
                           prop = NULL,
                           sampleinfo = sampleinfo,
                           control = "group1",
                           case = "group2",
                           de_method = "edgeR")

## scatter plot to compare the effect of adjusting cell proportion differences
comparedeg_scatter(results1 = deres[[2]],
```

```
                results2 = deres_notadjust[[2]],
              result_names = c("adjust for cell proportion", "not adjust for cell proportion"),
                fc_cutoff = 1.5,
                pval_cutoff = 0.05,
                pvalflag = TRUE,
                interactive = FALSE)

## generate cell-type specific gene expression
ct_exprs_list <- celltype_expression(bulk = bulk,
                                ref = GetAssayData(refdata, slot = "data", assay = "SCT"),
                                      phenodata = phenodata,
                                      prop = decon_res[[1]],
                                      UMI_min = 0,
                                      CELL_MIN_INSTANCE = 1)
```

`scaling` *Normalization of gene expression data*

Description

Methods to use for data normalization.

Usage

```
scaling(
  matrix,
  option,
  gene_length = NULL,
  seed = 1234,
  ffpe_artifacts = FALSE,
  model = NULL
)
```

Arguments

`matrix`	a matrix-like objector of gene expression values with rows representing genes, columns representing samples or cells
`option`	character value specifying the normalization method to use. Has to be one of "none", "LogNormalize", "TMM", "median_ratios", "TPM", "SCTransform", "scran", "scater", "Linnorm".
`gene_length`	a data.frame with two columns. The first column represents gene names that match with provided bulk data. The second column represents length of each gene. Only applicable when norm_method is selected as "TPM"
`seed`	random seed used for simulating FFPE artifacts. Only applicable when ffpe_artifacts is set to TRUE.

ffpe_artifacts	logical value indicating whether to add simulated ffpe artifacts in the bulk data. Only applicable to simulation experiments in evaluating the effect of FFPE artifacts.
model	pre-constructed ffpe model data. Can be downloaded from github: https://github.com/Liuy12/SCdeconR/t

Details

refer to scdecon for more details.

Value

a matrix-like object with the same dimension of input object after data normalization.

---

scdecon *Deconvolution of bulk RNA-seq data*

Description

Deconvolution of bulk RNA-seq data based on single-cell reference data. Eight bulk deconvolution methods, along with eight normalization methods and four transformation methods are available.

Usage

```
scdecon(
  bulk,
  ref,
  phenodata,
  filter_ref = TRUE,
  marker_genes = NULL,
  genes_to_remove = NULL,
  min_pct_ct = 0.05,
 decon_method = c("scaden", "CIBERSORT", "OLS", "nnls", "FARDEEP", "RLR", "MuSiC",
    "SCDC", "scTAPE"),
 norm_method_sc = c("none", "LogNormalize", "SCTransform", "scran", "scater", "Linnorm"),
  norm_method_bulk = c("none", "TMM", "median_ratios", "TPM"),
  trans_method_sc = c("none", "log2", "sqrt", "vst"),
  trans_method_bulk = c("none", "log2", "sqrt", "vst"),
  gene_length = NULL,
  lfc_markers = log2(1.5),
  marker_strategy = c("all", "pos_fc", "top_50p_logFC", "top_50p_AveExpr"),
  to_remove = NULL,
  ffpe_artifacts = FALSE,
  model = NULL,
  prop = NULL,
  cibersortpath = NULL,
  pythonpath = NULL,
  tmpdir = NULL,
```

```
  remove_tmpdir = TRUE,
  seed = 1234,
  nsamples = 1000,
  return_value_only = FALSE,
  verbose = FALSE
)
```

Arguments

bulk	a matrix or data.frame of unnormalizaed & untransformed bulk RNA-seq gene expression values with rows representing genes, columns representing samples
ref	a matrix or data.frame of untransformed scRNA-seq gene expression counts with rows representing genes, columns representing cells. This data will be used to deconvolute provided bulk RNA-seq data.
phenodata	a data.frame with rows representing cells, columns representing cell attributes. It should at least contain the first three columns as: 1. cell barcodes 2. cell types 3. subject ids
filter_ref	logical value indicating whether outlier genes & cells should be removed from the provided reference data. Defaults to TRUE
marker_genes	a data.frame of two columns. First column represents cell types in ref; second column represents gene names of marker genes. If specified, those genes will be used to construct signature matrix for mark-gene based deconvolution methods, such as CIBERSORT, OLS, nnls, FARDEEP and RLR. Default to NULL, carry out differential analysis to identify marker genes for each cell type in ref.
genes_to_remove	a vector of gene names to remove from the reference scRNA-seq data. Default to NULL.
min_pct_ct	a numeric value indicating the minimum required proportion of expressing cells per cell type for marker gene identification. Only applicable when marker_genes is NULL. Default to 0.05.
decon_method	character value specifying the deconvolution method to use. Has to be one of "scaden", "CIBERSORT", "OLS", "nnls", "FARDEEP", "RLR", "MuSiC", "SCDC", "scTAPE". See details for more information.
norm_method_sc	character value specifying the normalization method to use for reference data. Has to be one of "none","LogNormalize", "SCTransform", "scran", "scater", "Linnorm". See details for more information.
norm_method_bulk	character value specifying the normalization method to use for bulk data. Has to be one of "none", "TMM", "median_ratios", "TPM". See details for more information.
trans_method_sc	character value specifying the transformation method to use for both bulk & reference data. Has to be one of "none", "log2", "sqrt", "vst". See details for more information.

`trans_method_bulk`
: character value specifying the transformation method to use for both bulk & reference data. Has to be one of "none", "log2", "sqrt", "vst". See details for more information.

`gene_length`
: a data.frame with two columns. The first column represents gene names that match with provided bulk data. The second column represents length of each gene. Only applicable when norm_method is selected as "TPM".

`lfc_markers`
: log2 fold change cutoff used to identify marker genes for deconvolution. The option only applicable to marker-gene based approaches, such as CIBERSORT, OLS, nnls, FARDEEP and RLR. Only applicable when marker_genes is NULL.

`marker_strategy`
: further strategy in selecting marker genes besides applying the log2 fold change cutoff. Can be chosen from: "all", "pos_fc", "top_50p_logFC" or "top_50p_AveExpr". See details for more information. Only applicable when marker_genes is NULL.

`to_remove`
: character value representing the cell type to remove from reference data. Only applicable to simulation experiments in evaluating effect of cell type removal from reference.

`ffpe_artifacts`
: logical value indicating whether to add simulated ffpe artifacts in the bulk data. Only applicable to simulation experiments in evaluating the effect of FFPE artifacts.

`model`
: pre-constructed ffpe model data. Can be downloaded from github: https://github.com/Liuy12/SCdeconR/b

`prop`
: a matrix or data.frame of simulated cell proportion values with rows representing cell types, columns representing samples. Only applicable to simulation experiments in evaluating the effect of cell type removal from reference.

`cibersortpath`
: full path to CIBERSORT.R script.

`pythonpath`
: full path to python binary where scaden was installed with.

`tmpdir`
: temporary processing directory for scaden or scTAPE.

`remove_tmpdir`
: a logical value indicating whether to remove tmpdir once scaden is completed. Default to TRUE.

`seed`
: random seed used for simulating FFPE artifacts. Only applicable when ffpe_artifacts is set to TRUE.

`nsamples`
: number of artificial bulk samples to simulate for scaden. Default to 1000.

`return_value_only`
: return a list of values only without performing deconvolution. This could be helpful in cases where the user want to apply their own deconvolution algorithms. Default to FALSE.

`verbose`
: a logical value indicating whether to print messages. Default to FALSE.

Details

decon_method should be one of the following:

scaden a deep learning based method using three multi-layer deep neural nets. To use scaden, you need to firstly install scaden via `pip` or `conda`, the provide the python path to `pythonpath` option.

CIBERSORT a marker gene based support vectors regression approach. CIBERSOR does not allow redistribution. To use CIBERSORT, you need to request the source code from the authors & provide the path of CIBERSORT.R script to `cibersortpath` option.

OLS ordinary least squares.

nnls non-negative least squares.

FARDEEP robust regression using least trimmed squares

RLR robust regression using an M estimator

MuSiC multi-subject single-cell deconvolution

SCDC an ENSEMBLE method to integrate deconvolution results from different scRNA-seq datasets

scTAPE Deep autoencoder based deconvolution

norm_method should be one of the following:

none no normalization is performed.

LogNormalize `LogNormalize` method from seurat.

TMM TMM method from `calcNormFactors` function from edgeR.

median_ratios median ratio method from `estimateSizeFactors,DESeqDataSet-method` function from DESeq2.

TPM Transcript per million. TPM has to be chosen if ffpe_artifacts is set to TRUE.

SCTransform `SCTransform` method from Seurat.

scran `computeSumFactors` method from scran.

scater `librarySizeFactors` method from scater.

Linnorm `Linnorm` method from Linnorm.

trans_method should be one of the following:

none no transformation is performed.

log2 log2 transformation. 0.1 is added to the data to avoid logarithm of 0s.

sqrt square root transformation.

vst `varianceStabilizingTransformation` method from DESeq2.

marker_strategy should be one of the following:

all all genes passed fold change threshold will be used.

pos_fc only genes with positive fold changes will be used.

top_50p_logFC only genes with top 50 percent positive fold changes will be used.

top_50p_AveExpr only genes with top 50 percent average expression will be used.

Value

a list containing two or four elements.

first element a data.frame of predicted cell-type proportions, with rows representing cell types, columns representing samples.

second element a data.frame of fitting errors of the algorithm; first column represents sample names, second column represents RMSEs.

optional third element a data.frame of simulated cell proportion after removing the specified cell_type. Only applicable to simulation experiments.

optional fourth element a data.frame of marker genes used for deconvolution. Only applicable to marker-gene based deconvolution methods.

Examples

```
ref_list <- c(paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample1"),
              paste0(system.file("extdata", package = "SCdeconR"), "/refdata/sample2"))
phenopath1 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample1.txt")
phenopath2 <- paste0(system.file("extdata", package = "SCdeconR"),
"/refdata/phenodata_sample2.txt")
phenodata_list <- c(phenopath1,phenopath2)

# construct integrated reference using harmony algorithm
refdata <- construct_ref(ref_list = ref_list,
                     phenodata_list = phenodata_list,
                     data_type = "cellranger",
                     method = "harmony",
                     group_var = "subjectid",
                     nfeature_rna = 50,
                     vars_to_regress = "percent_mt", verbose = FALSE)

phenodata <- data.frame(cellid = colnames(refdata),
                        celltypes = refdata$celltype,
                        subjectid = refdata$subjectid)

prop <- data.frame(celltypes = unique(refdata$celltype),
proportion = rep(1/length(unique(refdata$celltype)), length(unique(refdata$celltype))))
bulk_sim <- bulk_generator(ref = GetAssayData(refdata, slot = "data", assay = "SCT"),
                           phenodata = phenodata,
                           num_mixtures = 20,
                           prop = prop,
                           num_mixtures_sprop = 1)

## perform deconvolution based on "OLS" algorithm
decon_res <- scdecon(bulk = bulk_sim[[1]],
                     ref = GetAssayData(refdata, slot = "data", assay = "SCT"),
                     phenodata = phenodata,
                     filter_ref = TRUE,
                     decon_method = "OLS",
                     norm_method_sc = "LogNormalize",
```

```
norm_method_bulk = "TMM",
trans_method_sc = "none",
trans_method_bulk = "log2",
marker_strategy = "all")
```

`transformation` *Transformation of gene expression data*

Description

Methods to use for data transformation.

Usage

```
transformation(matrix, option)
```

Arguments

`matrix`	a matrix-like objector of gene expression values with rows representing genes, columns representing samples or cells
`option`	character value specifying the transformation method to use. Has to be one of "none", "log", "sqrt", "vst".

Details

refer to scdecon for more details.

Value

a matrix-like object with the same dimension of input object after data transformation.

Index

aheatmap, *11*

bulk_generator, 2

calcNormFactors, *23*
celltype_expression, 4
comparedeg_scatter, 5
comparegsea_scatter, 6
compute_metrics, 7
computeSumFactors, *23*
construct_ref, 8, *14*

gsea_heatmap, 10, *16*
gsea_rwplot, 11, *16*
gsea_sumplot, 12, *16*

librarySizeFactors, *23*
load_scdata, 13
LogNormalize, *23*

p.adjust, *17*
prepare_rnk, 14
prop_barplot, 15

read_gmt *(*reformat_gmt*)*, 15
reformat_gmt, 15
run_de, *4*, *5*, *11*, *14*, 16

scaling, 19
scdecon, *20*, 20, *25*
SCTransform, *9*, *13*, *23*

transformation, 25

varianceStabilizingTransformation, *23*

write_gmt *(*reformat_gmt*)*, 15