Package ‘scMINER’

September 21, 2023

Type Package

Title scMINER

Version 1.0.0

Author Yu Lab (St.Jude Children's Research Hospital)

Maintainer Liang Ding <liang.ding@stjude.org>, Hao Shi <hao.shi@stjude.org>, Xinran Dong <xinran.dong@foxmail.com>

Description
Mutual information-based single-cell clustering and network-enabled hidden driver analysis

License Apache License

Encoding UTF-8

LazyData TRUE

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Depends R (>= 4.0.3),
Biobase (>= 2.50.0),
ggplot2 (>= 3.3.3),
RColorBrewer (>= 1.1.2),
reshape2 (>= 1.4.4),
rmarkdown (>= 2.8),
kableExtra (>= 1.3.4),
dplyr (>= 1.0.6),
grDevices (>= 4.0.3),
scales (>= 1.1.1),
limma (>= 3.46.0),
anndata (>= 0.7.5.3)

Imports plyr (>= 1.8.6),
Matrix (>= 1.5.3),
stats (>= 4.0.3),
methods (>= 4.0.3),
ComplexHeatmap (>= 2.6.2),
igraph (>= 1.2.6),
rhdf5 (>= 2.34.0),
renv

Suggests NetBID2 (>= 2.0.3),
openxlsx (>= 4.2.3),
knitr (>= 1.33),
testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

R topics documented:

combinePvalVector 2
ConvertNet2List 3
CreateSparseEset 4
DAG_ttest 4
draw.bubblePlot2 5
draw.group.barplot 5
draw.marker.bbp 6
draw.scRNAseq.QC 7
feature_heatmap 8
feature_highlighting 9
feature_vlnplot 11
generateMICAinput 12
generateSJARACNeInput 12
get.DA 14
get.network.scMINER 15
get.Topdrivers 15
GetActivityFromSJARACNe 16
getDE.limma 18
get_activity 19
MICAplot 20
preMICA.filtering 21
readMICAoutput 22
readscRNAseqData 23
scMINER.dir.create 24
SJARACNe_filter 25
SparseExpressionSet-class 25

Index **26**

`combinePvalVector` *Combine P Values Using Fisher's Method or Stouffer's Method*

Description

`combinePvalVector` is a function to combine multiple comparison's P values using Fisher's method or Stouffer's method.

Usage

```
combinePvalVector(pvals, method = "Stouffer", signed = TRUE, twosided = TRUE)
```

Arguments

`pvals` a vector of numerics, the P values from multiple comparison need to be combined.

`method` character, users can choose between "Stouffer" and "Fisher". Default is "Stouffer".

`signed` logical, if TRUE, will give a sign to the P value to indicate the direction of testing. Default is TRUE.

`twosided` logical, if TRUE, P value is calculated in a one-tailed test. If FALSE, P value is calculated in a two-tailed test, and it falls within the range 0 to 0.5. Default is TRUE.

Value

Return a vector contains the "Z-statistics" and "P.Value".

Examples

```
combinePvalVector(c(0.1,1e-3,1e-5))
combinePvalVector(c(0.1,1e-3,-1e-5))
```

`ConvertNet2List` *Convert Pairwise Network Data Frame to Driver-to-Target List* `ConvertNet2List` *is a helper function in the* `get.SJAracne.network`. *But if users have their own pairwise gene network files, they can convert it to driver-to-target list object.*

Description

Convert Pairwise Network Data Frame to Driver-to-Target List `ConvertNet2List` is a helper function in the `get.SJAracne.network`. But if users have their own pairwise gene network files, they can convert it to driver-to-target list object.

Usage

```
ConvertNet2List(net_dat = NULL)
```

Arguments

`net_dat` data.frame, must contain two columns with column names "source" (driver) and "target" (target genes). "MI" (mutual information) and "spearman" (spearman correlation coefficient) columns are optional, but strongly suggested to use. If "MI" and "spearman" columns are missing, errors may occur in some following steps (e.g. es.method='weightedmean' in `cal.Activity`).

Value

Return a list. The names of the list elements are drivers. Each element is a data frame, contains three columns. "target", target gene names; "MI", mutual information; "spearman", spearman correlation coefficient.

Examples

```
TF_file <- system.file('PBMC14KDS_DemoDataSet/SJAR/Bcell_11546_11539_77/tf_final/consensus_network_ncol_.tx
            package = "scMINER")
tf.network  <- get.network.scMINER(network_file=TF_file)
network_list <- ConvertNet2List(tf.network$network_dat)
## Not run:
```

CreateSparseEset *CreateSparseEset*

Description

Create a S4 class which utilize 'ExpressionSet' template yet compatible with sparseMatrix type of assaydata

Usage

```
CreateSparseEset(
  data = NULL,
  meta.data = NULL,
  feature.data = NULL,
  add.meta = TRUE
)
```

Arguments

`data` Sparse expression data, could be from either of these class:c('matrix','dgTMatrix','dgCMatrix').Requ

`meta.data` phenotype data which rownames should be the same as data colnames; Optional; Default as NULL

`feature.data` feature data which rownames should be the same as data rownames; Optional; Default as NULL

`add.meta` logical; Whether or not calculate extra pheonotype info including total number of UMI, number of non-zero gene for each cell, mitochondrial percentage and spike-in gene expression percentage and store them in Biobase::pData

Value

A customized S4 class using ExpressionSet class as prototype

DAG_ttest *DAG_ttest*

Description

DAG_ttest

Usage

```
DAG_ttest(d, group)
```

draw.bubblePlot2 *Inner function for simple bubbleplots*

Description

Inner function for simple bubbleplots

Usage

```
draw.bubblePlot2(
  df = NULL,
  xlab,
  ylab,
  clab,
  slab,
  low.col = "#004C99",
  high.col = "#CC0000",
  plot.title = NULL,
  xlab_angle = 0,
  xlab_hjust = 0.5
)
```

Arguments

df	re-structured data.frame for bubble plots
xlab	string
ylab	string
clab	string
slab	string
low.col	string,default as "#004C99"
high.col	string, default as "CC0000
plot.title	string

Value

a ggplot object

draw.group.barplot *Draw barplot for composition study*

Description

Draw barplot for composition study

Usage

```
draw.group.barplot(input_eset, group_by, color_by, colors = NULL)
```

Arguments

`input_eset`	ExpressionSet that include group information in phenotype data
`group_by`	Group criteria for bars, should be a variable stored in Biobase::pData(input_eset)
`color_by`	Coloring criteria of bar fractions, should be a variable stored in Biobase::pData(input_eset)
`colors`	color values to feed in scale_fill_manual, default as NULL; If NULL, then default color for ggplot will be used

Value

a ggplot object

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                         package = "scMINER")
load(demo_file)
draw.group.barplot(pbmc.14k.DS.eset.log2,
                   group_by = 'celltype',
                   color_by = 'celltype')
```

`draw.marker.bbp` *Generate visualization for marker scores via bubble plot*

Description

Marker visualizatoin from known markers/signatures, requires knowledge-based marker list as input

Usage

```
draw.marker.bbp(
  ref = NULL,
  input_eset,
  feature = "geneSymbol",
  group_name = "ClusterRes",
  save_plot = FALSE,
  width = 8,
  height = 5,
  plot_name = "AnnotationBubbleplot.png"
)
```

Arguments

`ref`	reference dataframe, includes positive or negative markers for different cell types; Specify first column as different cell types, second columns as markers, third columns as weight (postive or negative marker)
`input_eset`	expressionSet/SparseExpressionSet object with clustering membership stored in Biobase::pData
`feature`	feature type from second column of your reference , should be in colnames(Biobase::fData(eset))
`group_name`	a character, the variable containing clustering label in Biobase::pData(eset); or any other group information stored in Biobase::pData(eset)

`save_plot` logical, whether or not save your plot; if TRUE, plot will be saved as plot_name

`width` default as 8, inch as unit

`height` default as 5, inch as unit

`plot_name` plot name, please include plot type

Details

Visualize marker score of different cell types on bubbleplot

Value

A ggplot object

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                        package = "scMINER")
load(demo_file)
markers_file <- system.file('PBMC14KDS_DemoDataSet/DATA/',
 'Immune_signatures.xlsx',
  package = "scMINER")
markers <- openxlsx::read.xlsx(markers_file)
draw.marker.bbp(ref = markers, input_eset = pbmc.14k.DS.eset.log2,
               width = 6, height = 4, feature = "geneSymbol",
               group_name = "ClusterRes", save_plot = FALSE)
## Not run:
```

draw.scRNAseq.QC *draw.scRNAseq.QC*

Description

generated a scRNA-seq quality control report in html with Rmarkdown

Usage

```
draw.scRNAseq.QC(
  SparseEset,
  project.name,
  plot.dir = "./QC/",
  output.cutoff = TRUE,
  group = "group",
  only.cutoff = FALSE
)
```

Arguments

`SparseEset` an SparseEset generated by CreateSparseEset

`project.name` a character, project name to print on report

`plot.dir` a character, output directory for QC reports

`output.cutoff` logical, whether or not return a list of suggested thresholds for filtering

`group` a character, a variable name indicate groupping information (stored in Biobase::pData) to help generate violin plots

`only.cutoff` logical, whether or not only return cuttoff without plotting QC

Details

draw.scRNAseq.QC

Value

an R markdown QC report and a list of suggested threshold (if specify)

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                         package = "scMINER")
load(demo_file)
cutoffs <- scMINER::draw.scRNAseq.QC(SparseEset = pbmc.14k.DS.eset.log2,
             project.name = 'test',
             plot.dir = '.',
             group = "group",
             output.cutoff = TRUE,only.cutoff=TRUE)
```

`feature_heatmap` *Visualize gene expression level on scRNA-seq data via heatmap*

Description

This plot will visualiz feature info in scatter plot by outputing a ggplot object

Usage

```
feature_heatmap(
  input_eset,
  target,
  feature = "geneSymbol",
  group_name = "label",
  name = "log2Exp",
  save_plot = TRUE,
  width = 4,
  height = 8,
  cluster_rows = FALSE,
  colors = rev(colorRampPalette(brewer.pal(10, "RdYlBu"))(256)),
  plot_name = "GeneHeatmap.png",
  ...
)
```

Arguments

`input_eset`	Input expression set
`target`	a character or a character vector indicating feature names
`feature`	a character, which feature to visualize
`group_name`	a character, label to visualize on the top of heatmap
`name`	character, name of value visualized in color scale
`save_plot`	logical, whether to save plots or not
`width`	numerical
`height`	numerical
`cluster_rows`	logical, if or not cluster rows
`colors`	color palette
`plot_name`	character, name of heamap
`...`	parameter to be passed to ComplexHeatmap::Heatmap

Value

a ggplot object

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                        package = "scMINER")
load(demo_file)
genes_of_interest <-c("CD3D", "CD27", "IL7R",
                      "SELL", "CCR7", "IL32",
                      "GZMA", "GZMK", "DUSP2",
                      "CD8A", "GZMH", "GZMB",
                      "CD79A", "CD79B", "CD86", "CD14")
scMINER::feature_heatmap(input_eset = pbmc.14k.DS.eset.log2,
                     target = genes_of_interest,
                     group_name = "ClusterRes",
                     save_plot = FALSE,
                     width = 6, height = 6,
                     name = "log2Exp")
```

`feature_highlighting` *feature_highlighting*

Description

This plot will visualize feature info on scatter plot by outputing a ggplot object

Usage

```
feature_highlighting(
  input_eset,
  target = NULL,
  feature = "geneSymbol",
  x = "X",
  y = "Y",
  wrap_by = NULL,
  ylabel = "Expression",
  pct.size = 0.8,
  title.size = 15,
  ncol = 4,
  alpha = 0.8,
  colors = colorRampPalette(c("#E3E3E3", "#BCA2FC", "#4900FE"), interpolate =
    "linear")(8)
)
```

Arguments

`input_eset`	Input expression set
`target`	a character vector, the list of feature to visualize
`feature`	character, which feature to visualize
`x`	cordinates for x axis
`y`	cordinates for y axis
`wrap_by`	character, variable to wrap plot with
`ylabel`	a characterm, title of y axis
`pct.size`	numrical, point size
`title.size`	numerical, default as 5
`ncol`	cordinates for y axis
`alpha`	numerical, default as 0.8
`colors`	color palette for feature highlighting

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                         package = "scMINER")
load(demo_file)
genes_of_interest <-c("CD3D", "CD27", "IL7R",
                      "SELL", "CCR7", "IL32",
                      "GZMA", "GZMK", "DUSP2",
                      "CD8A", "GZMH", "GZMB",
                      "CD79A", "CD79B", "CD86", "CD14")
scMINER::feature_highlighting(input_eset = pbmc.14k.DS.eset.log2,
                      target = genes_of_interest,
                      feature = "geneSymbol",
                      ylabel = "log2Exp",
                      x = "X", y = "Y",
                      pct.size = 0.5)
```

feature_vlnplot *feature_vlnplot*

Description

This plot will visualize feature info in violin plot by outputing a ggplot object

Usage

```
feature_vlnplot(
  input_eset,
  target = NULL,
  feature = "geneSymbol",
  group_by = "celltype",
  ylabel = "Expression",
  color_by = NULL,
  colors = NULL,
  ncol = 3,
  stat = "median",
  boxplot = FALSE,
  title.size = 5
)
```

Arguments

input_eset	Input expression set
target	a character vector, the list of feature to visualize
feature	character, which feature to visualize
group_by	character, which group info to visualize as x axis
ylabel	a character, title of y axis
color_by	character, which group info to define color, if NULL, then violin plots will be colored by 'group_by'
colors	character vector, default as NULL, will use ggplot default color palette
ncol	cordinates for y axis
stat	a character, whether to plot median or mean as a black dot on violinplot
boxplot	logical, whether to plot boxplot on violinplot
title.size	numerical, default as 5

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                         package = "scMINER")
load(demo_file)
genes_of_interest <-c("CD3D", "CD27", "IL7R",
                       "SELL", "CCR7", "IL32",
                       "GZMA", "GZMK", "DUSP2",
                       "CD8A", "GZMH", "GZMB",
                       "CD79A", "CD79B", "CD86", "CD14")
scMINER::feature_vlnplot(input_eset = pbmc.14k.DS.eset.log2,
                              target = genes_of_interest,
                              feature = "geneSymbol",
                              group_by = "ClusterRes",
                              ylabel = "log2Exp", ncol = 4)
```

generateMICAinput *Generate MICA input accepted txt or h5ad file*

Description

A utility function that helps generate MICA input from a data matrix with rownames and colnames

Usage

```
generateMICAinput(d, filename="project_name_MICAinput.h5")
```

Arguments

sampleN	integer, number of cells that should be sampled for computational efficiency purpose, default is 50000. If sampleN=NULL, no sampling will be performed.
seed	integer, the random seed for sampling, default is 1.
scminer.par	list for the parameter settings in scMINER pipeline, optional.
d	matrix with colnames as cell/sample info, rownames as gene/feature info
filename	filename of your MICA input file, supported format: txt or h5

Value

A txt file or a h5 file that could be read in MICA

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                         package = "scMINER")
load(demo_file)
MICA.cmd <- generateMICAinput(eset = pbmc.14k.DS.eset.log2 ,
                       filepath = 'test.h5ad')
```

generateSJARACNeInput *Generate SJARACNE input with designed folder structure*

Description

This function helps to generate appropriate input files for SJARACNe pipeline. It can take transcription factor/signaling gene reference from internal(stored in package) or external (manual define)

Usage

```
generateSJARACNeInput(
  input_eset,
  ref = NULL,
  funcType = NULL,
  input_driver = NULL,
  sampleN = 1000,
  seed = 1,
  wd.src,
  group_name
)
```

Arguments

`input_eset`	An expressionSet
`ref`	c("hg", "mm"), could be a manually defined geneSymbol vector
`funcType`	c("TF","SIG", NULL), if NULL then both TF and SIG will be considered
`input_driver,`	a list of drivers for calculation. If NULL, curated driver list in the R package will be used.
`sampleN`	integer, number of cells that should be sampled per group for computational efficiency purpose, default is 1000. If sampleN=NULL, no sampling will be performed.
`seed`	integer, the random seed for sampling, default is 1.
`wd.src`	output path
`group_name`	name of group for sample identification

Details

generateSJARACNeInput

Value

SJARACNe input files for each subgroups

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                         package = "scMINER")
load(demo_file)
SJAR.cmd.tf <- generateSJARACNeInput(
               input_eset = pbmc.14k.DS.eset.log2,
               funcType = "TF",
               ref = "hg",  # human
               wd.src = 'test/',  # output directory
               group_name = "celltype")
## Not run:
```

get.DA *Find differential activity genes from activity matrix*

Description

get.DA is a wraper of (DAG_test, and getDE.limma), which helps to conduct two_sided t.test on all genes in specific group VS Others to find differential activity genes, a table with essential statistics will be outputted.

Usage

```
get.DA(
  input_eset = NULL,
  group_name = "celltype",
  group_case = NULL,
  group_ctrl = NULL,
  method = "t.test"
)
```

Arguments

input_eset	ExpressionSet that stores group information in Biobase::pData
group_name	a character string, column name in Biobase::pData(input_eset) that indicates group info
group_case	NULL(If do get.DA for all group vs others) or a character string (one specific group vs others) of column name in Biobase::pData(input_eset) that indicates group info
group_ctrl	NULL(If one vs Others); a character indicate case group if do pairwise analysis
method	a character from c("t.test", "limma"), which method will be used to identify differential activity gene

Value

output would be a data.frame containing: t.statistics, p.value, log2FC, z.score, and mean Activity value

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/celltype_Activity.RData',
                                        package = "scMINER")
load(demo_file)
DAG_result_tf <- get.DA(input_eset = AC_eset$AC.TF,
                     group_name = "celltype")
```

`get.network.scMINER` *Read SJARACNe Network Result and Return it as List Object(adapted from NetBID2)*

Description

`get.network.scMINER` reads SJARACNe network construction result and returns a list object with network data frame, driver-to-target list and igraph object wrapped inside.

Usage

```
get.network.scMINER(network_file = NULL)
```

Arguments

`network_file` character, the path for storing network file. For the output of SJAracne, the name of the network file will be "consensus_network_ncol_.txt" under the output directory.

Details

In the demo, "consensus_network_ncol_.txt" file will be read and convert into a list object. This list contains three elements, `network_data`, `target_list` and `igraph_obj`. `network_dat` is a data.frame, contains all the information of the network SJARACNe constructed. `target_list` is a driver-to-target list object. Please check details in `get_net2target_list`. `igraph_obj` is an igraph object used to save this directed and weighted network. Each edge of the network has two attributes, `weight` and `sign`. `weight` is the "MI (mutual information)" value and `sign` is the sign of the spearman correlation coefficient (1, positive regulation; -1, negative regulation).

Value

Return a list containing three elements, `network_dat`, `target_list` and `igraph_obj`.

Examples

```
TF_file <- system.file('PBMC14KDS_DemoDataSet/SJAR/Bcell_11546_11539_77/tf_final/consensus_network_ncol_.tx
            package = "scMINER")
tf.network  <- get.network.scMINER(network_file=TF_file)
## Not run:
```

`get.Topdrivers` *get.Topdrivers*

Description

Help quick pick top master regulators from previous differential activity analysis results

Usage

```
get.Topdrivers(
  DAG_result = DAG_result,
  n = 5,
  degree_filter = c(50, 500),
  celltype = NULL
)
```

Arguments

DAG_result	Output table from function FindDAG
n	threshold to pick top master regulators(top n)
degree_filter	filter out drivers with target number less than certain value
celltype	character, output top hits are from which celltype

Value

A list of top master regulators among different groups

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/celltype_Activity.RData',
                                        package = "scMINER")
load(demo_file)
DAG_result_tf <- get.DA(input_eset = AC_eset$AC.TF,
                        group_name = "celltype")
celltype <- levels(Biobase::pData(AC_eset$AC.TF)[,"celltype"])
TF_list <- get.Topdrivers(DAG_result = DAG_result_tf,
                          celltype = celltype,
                          n = 5, degree_filter = c(50, 600))
```

GetActivityFromSJARACNe

GetActivityFromSJARACNe

Description

Allocate network information from SJARACNe and calculate activity score for each hub genes.

Usage

```
GetActivityFromSJARACNe(
  SJARACNe_output_path = NA,
  SJARACNe_input_eset = NA,
  functype = "tf",
  group_name = NA,
  activity.method = "unweighted",
  activity.norm = TRUE,
  save_network_file = FALSE,
  save_path = NULL
)
```

Arguments

`SJARACNe_output_path`
Path to SJARACNe output folder(s)

`SJARACNe_input_eset`
Expressionset that you generate input from

`functype` character c("tf","sig"); If NULL, both activity from TF and SIG network will be calculated; default as NULL

`group_name` a string, group name stored in Biobase::pData that defines expression matrix separation

`activity.method`
c("weighted,unweighted), default to "unweighted"

`activity.norm` logical, default to TRUE.

`save_network_file`
logical, default to FALSE

`save_path` Path to save network file

Value

An expressionset with activity values

Author(s)

Chenxi Qian, <`chenxi.qian@stjude.org`>

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                         package = "scMINER")
load(demo_file)
out.dir.SJAR <- system.file('PBMC14KDS_DemoDataSet/SJAR/',
                         package = "scMINER")
acs.14k.tf <- GetActivityFromSJARACNe(
              SJARACNe_output_path = out.dir.SJAR,
              SJARACNe_input_eset = pbmc.14k.DS.eset.log2,
              activity.method="unweighted",
              activity.norm=TRUE,
              group_name =  "celltype",
              save_network_file=FALSE,
              functype="tf",
              save_path='.')
## Not run:
```

getDE.limma *Differential Expression Analysis and Differential Activity Analysis Between 2 Sample Groups Using Limma*

Description

getDE.limma is a function performs differential gene expression analysis and differential driver activity analysis between control group (parameter G0) and experimental group (parameter G1), using limma related functions.

Usage

```
getDE.limma(
  eset = NULL,
  G1 = NULL,
  G0 = NULL,
  G1_name = NULL,
  G0_name = NULL,
  verbose = TRUE,
  random_effect = NULL
)
```

Arguments

eset	ExpressionSet class object, contains gene expression data or driver activity data.
G1	a vector of characters, the sample names of experimental group.
G0	a vecotr of characters, the sample names of control group.
G1_name	character, the name of experimental group (e.g. "Male"). Default is "G1".
G0_name	character, the name of control group (e.g. "Female"). Default is "G0".
verbose	logical, if TRUE, sample names of both groups will be printed. Default is TRUE.
random_effect	a vector of characters, vector or factor specifying a blocking variable. Default is NULL, no random effect will be considered.

Value

Return a data frame. Rows are genes/drivers, columns are "ID", "logFC", "AveExpr", "t", "P.Value", "adj.P.Val", "B", "Z-statistics", "Ave.G1" and "Ave.G0". Names of the columns may vary from different group names. Sorted by P-values.

Examples

```
## Not run:
analysis.par <- list()
analysis.par$out.dir.DATA <- system.file('demo1','driver/DATA/',package = "NetBID2")
NetBID.loadRData(analysis.par=analysis.par,step='ms-tab')
phe_info <- Biobase::pData(analysis.par$cal.eset)
each_subtype <- 'G4'
G0  <- rownames(phe_info)[which(phe_info$`subgroup`!=each_subtype)] # get sample list for G0
G1  <- rownames(phe_info)[which(phe_info$`subgroup`==each_subtype)] # get sample list for G1
DE_gene_limma <- getDE.limma(eset=analysis.par$cal.eset,
                                G1=G1,G0=G0,
                                G1_name=each_subtype,
                                G0_name='other')
DA_driver_limma <- getDE.limma(eset=analysis.par$merge.ac.eset,
                                G1=G1,G0=G0,
                                G1_name=each_subtype,
                                G0_name='other')

## End(Not run)
```

get_activity *Calculate activity from network file or gene list*

Description

Calculate activity from network file or gene list

Usage

```
get_activity(
  Net = NULL,
  eset,
  tag = NULL,
  genelist = NULL,
  use.symbol = FALSE,
  feature = "geneSymbol",
  es.method = "mean",
  activity.method = "weighted",
  normalize = TRUE,
  sep.symbol = "."
)
```

Arguments

`Net`	Network data frame
`eset`	ExpressionSet/SparseExpressionSet with expression data
`tag`	If network is TF network or SIG network
`genelist`	A list of signature gene list
`use.symbol`	logical, in network file, use geneSymbol or use geneID
`feature`	character, use which feature as ID in Biobase::fData(eset)
`es.method`	character, which method to use to calculate actiivty value ("mean","maxmean")
`activity.method`	character, which method to use to estimate activity ("weighted","unweighted")
`normalize`	logical, if normalize or not
`sep.symbol`	which symbol to sparate name and tag

Details

If network object was loaded by get.network.scMINER function, then network dataframe is could be retrieved under network_dat slot.

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                         package = "scMINER")
load(demo_file)
TF_file <- system.file('PBMC14KDS_DemoDataSet/SJAR/Bcell_11546_11539_77/tf_final/consensus_network_ncol_.tx
             package = "scMINER")
tf.network  <- get.network.scMINER(network_file=TF_file)
acs1<-get_activity(Net = tf.network$network_dat,tag = "TF",
         normalize=TRUE,
         eset = pbmc.14k.DS.eset.log2,
         activity.method = 'unweighted',
         use.symbol=TRUE)
```

MICAplot *plot MICA clustering results or other meta variables*

Description

This function helps to generate a ggplot object for phenotypic visualization

Usage

```
MICAplot(
  input_eset,
  color_by = "ClusterRes",
  colors = NULL,
  X = NULL,
  Y = NULL,
  show_label = FALSE,
  label.size = 10,
  title.size = 20,
  title.name = "",
  pct = 0.5,
  alpha = 1
)
```

Arguments

`input_eset`	ExpressionSet that include visualization coordinates in phenotype data
`color_by`	Coloring criteria of data points, should be a variable stored in Biobase::pData(input_eset)
`colors`	character, color values, if NULL then use ggplot default color
`X`	character, column name of x axis
`Y`	character, column name of y axis
`show_label`	logical, whether or not to show label on tSNE plot
`label.size`	numerical, size of label ploted on figure
`title.size`	numerical, size of plot title, default as 10
`title.name`	character, title of plot, default as NULL
`pct`	numerical, size of point, default as 0.5
`aplha`	numerical, indicate point transparency

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                         package = "scMINER")
load(demo_file)
scMINER::MICAplot(input_eset = pbmc.14k.DS.eset.log2, X = "X", Y = "Y",
                  color_by = "ClusterRes", pct = 0.5)
## Not run:
```

preMICA.filtering *preMICA.filtering*

Description

scRNA-seq filtering function

Usage

```
preMICA.filtering(
  SparseEset,
  cutoffs,
  gene_filter = TRUE,
  nGene_filter = TRUE,
  nUMI_filter = TRUE,
  ERCC_filter = TRUE,
  Mito_filter = TRUE
)
```

Arguments

`SparseEset`	the sparseEset object outputted from draw.scRNAseq.QC
`cutoffs`	a list outputted by draw.scRNAseq.QC, if NULL, manual input will be required
`gene_filter`	logical; or a numerical number, indicating lower threshold for gene filtering based on how many non-zero cells each gene expressed in
`nGene_filter`	logical; a numerical number, indicating lower threshold put on number of gene expression in each cell for cell filtering
`nUMI_filter`	logical;a vector of two numerical number, indicating lower threshold and upper threshold put on number of total UMI for cell filtering
`ERCC_filter`	logical; a numerical number, indicating upper threshold put on ERCC percentage for cell filtering
`Mito_filter`	logical;a numerical number, indicating upper threshold put on Mitochondrial gene expression fraction for cell filtering

Details

preMICA.filtering

Value

A Sparse expression set

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                        package = "scMINER")
load(demo_file)
cutoffs <- draw.scRNAseq.QC(SparseEset = pbmc.14k.DS.eset.log2,
                                     project.name = project_name,
                                     plot.dir = 'test/',
                                     group = "group",
                                     output.cutoff = TRUE,
                                     only.cutoff = TRUE)
pbmc.14k.DS.eset.filter <- preMICA.filtering(SparseEset = pbmc.14k.DS.eset.log2,
      cutoffs = cutoffs)
```

readMICAoutput *readMICAoutput*

Description

Read MICA input and output to create an expressionSet for downstream analysis

Usage

```
readMICAoutput(eset = NULL, input_file, output_file, load_ClusterRes = TRUE)
```

Arguments

`eset`	a SparseMatrix Eset
`input_file`	input expression txt file of MICA pipeline
`output_file`	output ClusterMem.txt file from MICA pipeline
`load_ClusterRes`	logical, if TRUE, clustering results will be store at Biobase::pData(eset)$label

Value

A sparse expressionSet object

Examples

```
demo_file <- system.file('PBMC14KDS_DemoDataSet/DATA/pbmc.14k.DS.eset.log2.RData',
                        package = "scMINER")
load(demo_file)
MICA_output <- system.file('PBMC14KDS_DemoDataSet/MICA/clustering_umap_euclidean_19.txt',package = "scMINER"
pbmc.14k.DS.eset.log2 <- scMINER::readMICAoutput(eset = pbmc.14k.DS.eset.log2,
                          load_ClusterRes = TRUE,
                          output_file = MICA_output)
```

`readscRNAseqData` *readscRNAseqData*

Description

read scRNA-seq data, a wrapper of conventional data reading (read.delim) and 10x genomics data standarad output reading

Usage

```
readscRNAseqData(
  file,
  is.10x = TRUE,
  CreateSparseEset = TRUE,
  add.meta = F,
  sep = ","
)
```

Arguments

`file`	data path to 10x genomics output folder, which normally contains 3 files (matrix.mtx, gene or feature.tsv and barcode.csv), or data path to data txt/csv/tsv file
`is.10x`	logical, whether or not inputs are from CellRanger standard output
`CreateSparseEset`	logical, whether or not create sparse matrix incorporated expression set
`add.meta`	logical, whether or not calculate metadata info from expression matrixm, this is not suggested before merging/downsampling your data
`...`	parameters pass to read.delim if is.10x = FALSE

Value

A list or sparse matrix expression set

Examples

```
demo_dir <- system.file('PBMC14KDS_DemoDataSet/DATA/10X/',package = "scMINER")
pbmc.14k.DS.eset <- scMINER::readscRNAseqData(file = demo_dir,
                                             is.10x = TRUE,
                                             CreateSparseEset = TRUE,
                                             add.meta = TRUE)
```

scMINER.dir.create *Manipulation of Working Directories for scMINER pipeline*

Description

scMINER.dir.create is used to help users create an organized working directory for the network construction step in scMINER analysis. However, it is not essential for the analysis. It creates a hierarchical working directory and returns a list contains this directory information.

Usage

```
scMINER.dir.create(project_main_dir = NULL, project_name = NULL)
```

Arguments

project_main_dir	character, name or absolute path of the main working directory.
project_name	character, name of the project folder.

Details

This function needs users to define the main working directory and the project's name. It creates a main working directory with a subdirectory of the project. It also automatically creates five subfolders (DATA, SJAR, MICA, QC, PLOT) within the project folder. DATA/, storing data files; SJAR/, storing files needed for running SJAracne; MICA/, storing files needed for running MICA; QC/, storing Quality Control related plots; PLOT/, storing plot files; This function also returns a list object (example, scminer.par in the demo) with directory information wrapped inside.

Value

scMINER.dir.create returns a list object, containing main.dir (path of the main working directory), project.name (project name), out.dir (path of the project folder).

Examples

```
project_main_dir <- '../test'
project_name <- 'PBMC14KDS'
scminer.par <- scMINER::scMINER.dir.create(project_main_dir = project_main_dir,
                                           project_name = project_name)
## Not run:
# Creating a main working directory under the current working directory by folder name
scminer.par <- scMINER.dir.create("MyMainDir","MyProject")
# Or creating a main working directory under the current working directory by relative path
scminer.par <- scMINER.dir.create("./MyMainDir","MyProject")
# Or creating a main working directory to a specific path by absolute path
scminer.par <- scMINER.dir.create("~/Desktop/MyMainDir","MyProject")

## End(Not run)
```

SJARACNe_filter *SJARACNe_filter*

Description

This is the inner function to help generate SJARACNe input for scRNA-seq data, all non-informative (zero genes) will be filtered in by this function

Usage

```
SJARACNe_filter(eset.sel, tf.ref, sig.ref, wd.src, grp.tag)
```

Arguments

`eset.sel`	ExpressionSet to generate SJaracne input
`tf.ref`	A vector of reference transcription factors
`sig.ref`	A vector of reference signaling genes
`wd.src`	path to store SjAracne input
`grp.tag`	name of group for identification

Details

Non-expressed genes in subgroups are filtered. tf.ref should be coordinate with featureNames(eset.sel).

Value

A folder with picked master regulator and filtered gene expression matrix

SparseExpressionSet-class *SparseExpressionSet*

Description

SparseExpressionSet

Index

∗ **GetActivity**
GetActivityFromSJARACNe, 16
∗ **SJARACNe**
generateSJARACNeInput, 12

combinePvalVector, 2
ConvertNet2List, 3
CreateSparseEset, 4

DAG_ttest, 4
draw.bubblePlot2, 5
draw.group.barplot, 5
draw.marker.bbp, 6
draw.scRNAseq.QC, 7

feature_heatmap, 8
feature_highlighting, 9
feature_vlnplot, 11

generateMICAinput, 12
generateSJARACNeInput, 12
get.DA, 14
get.network.scMINER, 15
get.Topdrivers, 15
get_activity, 19
GetActivityFromSJARACNe, 16
getDE.limma, 18

MICAplot, 20

preMICA.filtering, 21

readMICAoutput, 22
readscRNAseqData, 23

scMINER.dir.create, 24
SJARACNe_filter, 25
SparseExpressionSet-class, 25