Package ‘scapGNN’

August 8, 2023

Type Package

Title Graph Neural Network-Based Framework for Single Cell Active Pathways and Gene Modules Analysis

Version 0.1.4

Date 2023-8-7

Author Xudong Han [aut, cre, cph], Xujiang Guo [fnd]

Maintainer Xudong Han <hanxd1217@163.com>

Description It is a single cell active pathway analysis tool based on the graph neural network (F. Scarselli (2009) <doi:10.1109/TNN.2008.2005605>; Thomas N. Kipf (2017) <arXiv:1609.02907v4>) to construct the gene-cell association network, infer pathway activity scores from different single cell modalities data, integrate multiple modality data on the same cells into one pathway activity score matrix, identify cell phenotype activated gene modules and parse association networks of gene modules under multiple cell phenotype. In addition, abundant visualization programs are provided to display the results.

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

RoxygenNote 7.2.3

Imports ActivePathways, AdaptGauss, coop, igraph, mixtools, reticulate, methods

Suggests rmarkdown, knitr

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2023-08-08 08:10:02 UTC

R topics documented:

`ATAC_net` *Results of ConNetGNN() for scATAC-seq data from SNARE-seq dataset*

Description

A list to store the gene association network of scATAC-seq data. Case data from the SNARE-seq dataset.

Usage

```
ATAC_net
```

Format

a list of three adjacency matrices.

Examples

```
data(ATAC_net)
```

`BIC_LTMG` *BIC_LTMG*

Description

The internal functions of the `scapGNN` package.

Usage

```
BIC_LTMG(y, rrr, Zcut)
```

Arguments

`y`	Internal parameters.
`rrr`	Internal parameters.
`Zcut`	Internal parameters.

Details

BIC_LTMG

`BIC_ZIMG` *BIC_ZIMG*

Description

The internal functions of the `scapGNN` package.

Usage

```
BIC_ZIMG(y, rrr, Zcut)
```

Arguments

`y`	Internal parameters.
`rrr`	Internal parameters.
`Zcut`	Internal parameters.

Details

BIC_ZIMG

ConNetGNN	*Construct association networks for gene-gene, cell-cell, and gene-cell based on graph neural network (GNN)*

Description

This function implements a graph neural network with two autoencoders. 1. AutoEncoder (AE) based on deep neural network: Infer latent associations between genes and cells. 2. Graph AutoEncoder (GAE) based on graph convolutional neural network: Construct association networks for gene-gene, cell-cell.

Usage

```
ConNetGNN(
  Prep_data,
  python.path = NULL,
  miniconda.path = NULL,
  AE.epochs = 1000,
  AE.learning.rate = 0.001,
  AE.reg.alpha = 0.5,
  use.VGAE = TRUE,
  GAE.epochs = 300,
  GAE.learning.rate = 0.01,
  GAE_val_ratio = 0.05,
  parallel = FALSE,
  seed = 125,
  GPU.use = FALSE,
  verbose = TRUE
)
```

Arguments

`Prep_data`	The input data is the result from the `Preprocessing` function.
`python.path`	The path to a Python binary. If python.path="default", the program will use the current system path to python.
`miniconda.path`	The path in which miniconda will be installed. If the `python.path` is NULL and conda or miniconda is not installed in the system, the program will automatically install miniconda according to the path specified by `miniconda.path`.
`AE.epochs`	The number of epoch for the deep neural network (AE). Default: `1000`.
`AE.learning.rate`	Initial learning rate of AE. Default: `0.001`.
`AE.reg.alpha`	The LTMG regularized intensity. Default: `0.5`.
`use.VGAE`	Whether to use Variational Graph AutoEncoder (VGAE). Default: `TRUE`.
`GAE.epochs`	The number of epoch for the GAE. Default: `300`.

`GAE.learning.rate`
: Initial learning rate of GAE. Default: `0.01`.

`GAE_val_ratio`
: For GAE, the proportion of edges that are extracted as the validation set. Default: `0.05`.

`parallel`
: Whether to use multiple processors to run GAE. Default: `FALSE` When `parallel=TRUE` (default), tow processors will be used to run GAE.

`seed`
: Random number generator seed.

`GPU.use`
: Whether to use GPU for GNN modules. Default: `FALSE`. If GPU.use=TRUE, CUDA needs to be installed.

`verbose`
: Gives information about each step. Default: `TRUE`.

Details

ConNetGNN

The `ConNetGNN` function establishes a graph neural network (GNN) framework to mine latent relationships between genes and cells and within themselves. This framework mainly includes two capabilities:

- 1.Deep neural network-based AutoEncoder inferring associations between genes and cells and generating gene features and cell features for the GAE.
- 2.The GAE takes the gene feature and cell feature as the node features of the initial gene correlation network and cell correlation network, and constructs the gene association network and cell association network through the graph convolution process.

The GNN is implemented based on `pytorch`, so an appropriate python environment is required:

- python >=3.9.7
- pytorch >=1.10.0
- sklearn >=0.0
- scipy >=1.7.3
- numpy >=1.19.5

If the user has already configured the python environment, the path of the python binary file can be directly entered into `python.path`. If the parameter `python.path` is NULL, the program will build a miniconda environment called `scapGNN_env` and configure python. We also provide environment files for conda: `/inst/extdata/scapGNN_env.yaml`. Users can install it with the command: `conda env create -f scapGNN_env.yaml`.

Value

A list:

cell_network Constructed cell association network.

gene_network Constructed gene association network.

cell_gene_network Constructed gene-cell association network.

Examples

```
require(coop)
require(reticulate)
require(parallel)
# Data preprocessing
data("Hv_exp")
Hv_exp <- Hv_exp[,1:20]
Hv_exp <- Hv_exp[which(rowSums(Hv_exp) > 0),]
Prep_data <- Preprocessing(Hv_exp[1:10,])

## Not run:
# Specify the python path
ConNetGNN_data <- ConNetGNN(Prep_data,python.path="../miniconda3/envs/scapGNN_env/python.exe")

## End(Not run)
```

`ConNetGNN_data` *The results of ConNetGNN() function*

Description

Results of ConNetGNN() function with Hv_exp as input.

Usage

```
ConNetGNN_data
```

Format

a list.

Examples

```
data(ConNetGNN_data)
```

`cpGModule` *Identify cell phenotype activated gene module*

Description

Mining activated gene modules in specific cell phenotype.

Usage

```
cpGModule(
  network.data,
  cellset,
  nperm = 100,
  cut.pvalue = 0.01,
  cut.fdr = 0.05,
  parallel.cores = 2,
  rwr.gamma = 0.7,
  normal_dist = TRUE,
  verbose = TRUE
)
```

Arguments

network.data	Network data constructed by the ConNetGNN function.
cellset	A vector of cell id. The specified cell set, which will be used as the restart set.
nperm	Number of random permutations. Default: 100.
cut.pvalue	The threshold of P-value, and genes below this threshold are regarded as gene modules activated by the cell set. Default: 0.01.
cut.fdr	The threshold of false discovery rate (FDR), and genes below this threshold are regarded as gene modules activated by the cell set. Default: 0.05.
parallel.cores	Number of processors to use when doing the calculations in parallel (default: 2). If parallel.cores=0, then it will use all available core processors unless we set this argument with a smaller number.
rwr.gamma	Restart parameter. Default: 0.7.
normal_dist	Whether to use pnorm to calculate P values. Default: TRUE.Note that if normal_dist is FALSE, we need to increase nperm (we recommend 100).
verbose	Gives information about each step. Default: TRUE.

Details

cpGModule

The cpGModule function takes a user-defined cell set as a restart set to automatically identify activated gene modules. A perturbation analysis was used to calculate a significant P-value for each gene. The Benjamini & Hochberg (BH) method was used to adjust the P-value to obtain the FDR. Genes with a significance level less than the set threshold are considered as cell phenotype activated gene modules.

Value

A data frame contains four columns:

Genes Gene ID.

AS Activity score.

Pvalue Significant P-value.

FDR False discovery rate.

Examples

```
require(parallel)
require(stats)

# Load the result of the ConNetGNN function.
data(ConNetGNN_data)
data(Hv_exp)

# Construct the cell set corresponding to 0h.
index<-grep("0h",colnames(Hv_exp))
cellset<-colnames(Hv_exp)[index]
cpGModule_data<-cpGModule(ConNetGNN_data,cellset,nperm=10,parallel.cores=1)
```

---

create_scapGNN_env — *Create the create_scapGNN_env environment on miniconda*

Description

The internal functions of the scapGNN package.

Usage

```
create_scapGNN_env()
```

Details

create_scapGNN_env

Fit_LTMG — *Fitting function for Left-truncated mixed Gaussian*

Description

The internal functions of the scapGNN package.

Usage

```
Fit_LTMG(x, n, q, k, err = 1e-10)
```

Arguments

x	Internal parameters.
n	Internal parameters.
q	Internal parameters.
k	Internal parameters.
err	Internal parameters.

Details

Fit_LTMG

`Global_Zcut` *Global_Zcut*

Description

The internal functions of the `scapGNN` package.

Usage

```
Global_Zcut(MAT, seed = 123)
```

Arguments

`MAT`	Internal parameters.
`seed`	Random number generator seed.

Details

Global_Zcut

`H9_0h_cpGM_data` *Cell-activated gene modules under the 0-hour phenotype*

Description

Results of cpGModule() function.

Usage

```
H9_0h_cpGM_data
```

Format

a list.

Examples

```
data(H9_0h_cpGM_data)
```

H9_24h_cpGM_data *Cell-activated gene modules under the 24-hour phenotype*

Description

Results of cpGModule() function.

Usage

```
H9_24h_cpGM_data
```

Format

a list.

Examples

```
data(H9_24h_cpGM_data)
```

H9_36h_cpGM_data *Cell-activated gene modules under the 36-hour phenotype*

Description

Results of cpGModule() function.

Usage

```
H9_36h_cpGM_data
```

Format

a list.

Examples

```
data(H9_36h_cpGM_data)
```

Hv_exp *Single-cell gene expression profiles*

Description

A log-transformed gene-cell matrix containing highly variable features.

Usage

```
Hv_exp
```

Format

a matrix.

Examples

```
data(Hv_exp)
```

instPyModule *Install the pyhton module through the reticulate R package*

Description

The internal functions of the `scapGNN` package.

Usage

```
instPyModule(module)
```

Arguments

`module`	Internal parameters.

Details

instPyModule

InteNet *Integrate network data from single-cell RNA-seq and ATAC-seq*

Description

For the SNARE-seq dataset, a droplet-based method to simultaneously profile gene expression and chromatin accessibility in each of thousands of single nuclei, the `InteNet` function can integrate network data of scRNA-seq data and scATAC-seq data (results of the `ConNetGNN` function) to into a gene-cell network.

Usage

```
InteNet(RNA_net, ATAC_net, parallel.cores = 2, verbose = TRUE)
```

Arguments

`RNA_net`	Network data for RNA datasets. Produced by the `ConNetGNN` function.
`ATAC_net`	Network data for ATAC datasets. Produced by the `ConNetGNN` function.
`parallel.cores`	Number of processors to use when doing the calculations in parallel (default: `2`). If `parallel.cores=0`, then it will use all available core processors unless we set this argument with a smaller number.
`verbose`	Gives information about each step. Default: `TRUE`.

Details

InteNet

The scATAC-seq dataset needs to be converted into a gene activity matrix according to the process of `Signac(https://satijalab.org/signac/articles/snareseq.html)`. The subsequent process is consistent with the scRNA-seq dataset. The `InteNet` function then integrates the network data of RNA-seq data and ATAC-seq data into a gene-cell network. With integrated network data as input, `scPathway` and `cpGModule` functions will infer pathway activity score matrix and gene modules supported by single-cell multi-omics.

Value

A list.

Examples

```
require(ActivePathways)
require(parallel)
data(RNA_net)
data(ATAC_net)
## Not run:
RNA_ATAC_IntNet<-InteNet(RNA_net,ATAC_net,parallel.cores=1)

## End(Not run)
```

```
# View data
data(RNA_ATAC_IntNet)
summary(RNA_ATAC_IntNet)
```

`isLoaded` *The internal functions of the* `scapGNN` *package*

Description

Determine if the package is loaded.

Usage

```
isLoaded(name)
```

Arguments

`name`	Internal parameters.

Details

isLoaded

`load_path_data` *load pathway or gene set's gmt file*

Description

The internal functions of the `scapGNN` package.

file format: 1. first index: pathway's name or ID. 2. second index: pathway's url or others, it dosen't matter. 3. third to all: gene symbols in pathway.

Usage

```
load_path_data(gmt_file_path)
```

Arguments

`gmt_file_path`	Internal parameters.

Details

load_path_data

Value

a list

`LTMG` *Left-truncated mixed Gaussian*

Description

Functional implementation of Left-truncated mixed Gaussian. The internal functions of the `scapGNN` package.

Usage

```
LTMG(VEC, Zcut_G, k = 5)
```

Arguments

`VEC`	Internal parameters.
`Zcut_G`	Internal parameters.
`k`	Internal parameters.

Details

LTMG

`LTMG-class` *An S4 class to represent the input data for LTMG.*

Description

An S4 class to represent the input data for LTMG.

Slots

`InputData` Input data for LTMG.

`OrdinalMatrix` LTMG output data.

`plotCCNetwork` *Visualize cell cluster association network graph*

Description

The `plotCCNetwork` function takes cells belonging to the same phenotype as a cluster. When cell phenotypes are not provided, the `plotCCNetwork` functions identify cell clusters based on edge betweenness. Cell interactions between cell clusters are merged into one edge by mean. The thickness of the edge indicates the strength of interaction between cell clusters.

Usage

```
plotCCNetwork(
  network.data,
  cell_id = NULL,
  cell_cluster = FALSE,
  cluster_method = "louvain",
  vertex.colors = NULL,
  vertex.size = 10,
  vertex.label.cex = 0.8,
  vertex.label.dist = 1,
  vertex.label.color = "black",
  edge.width = 5,
  margin = 0,
  layout = layout_with_lgl,
  legend.cex = 1.5,
  legend.pt.cex = 3,
  proportion = 1,
  plotgraph = TRUE
)
```

Arguments

`network.data`	The input network data is the result from the `ConNetGNN` function.
`cell_id`	A vector of cell phenotype.Methods include louvain (default), leading eigen and edge betweenness.
`cell_cluster`	A binary value. Whether to automatically identify cell clusters based on edge betweenness. Default: `FALSE`.
`cluster_method`	Community structure detection method
`vertex.colors`	The fill color of the vertex. The number of colors should match the number of cell phenotypes. If `NULL (default)`, the system will automatically assign colors.
`vertex.size`	The size of the vertex. Default: `10`.
`vertex.label.cex`	The font size for vertex labels. Default: `0.8`.

`vertex.label.dist`
: The distance of the label from the center of the vertex. If it is 0 then the label is centered on the vertex. Default: `1`.

`vertex.label.color`
: The color of the labels. Default: `black`.

`edge.width`
: The width of the edge. This does not affect the relative size of the edge weights. Default: `5`.

`margin`
: The amount of empty space below, over, at the left and right of the plot, it is a numeric vector of length four. Usually values between 0 and 0.5 are meaningful, but negative values are also possible, that will make the plot zoom in to a part of the graph. If it is shorter than four then it is recycled. Default: `0`.

`layout`
: Either a function or a numeric matrix. It specifies how the vertices will be placed on the plot. For details, please refer to the `igraph`Package. Default: `layout_with_lgl`.

`legend.cex`
: The font size of legend. Default: `1.5`.

`legend.pt.cex`
: Expansion factor(s) for the points. Default: `3`.

`proportion`
: This parameter specifies what percentage of edges to display (edges are sorted by their weight in descending order). Default: `1`, all edges are used.

`plotgraph`
: Whether to draw the picture. Default: `TRUE`. If `FALSE`, the image will not be displayed but the network data will be returned in the igraph data format.

Details

plotCCNetwork

Value

Graph or network data.

Examples

```
require(igraph)
require(graphics)

data(ConNetGNN_data)

# Construct the cell phenotype vector.
cell_id<-colnames(ConNetGNN_data[["cell_network"]])
temp<-unlist(strsplit(cell_id,"_"))
cell_phen<-temp[seq(2,length(temp)-1,by=3)]
names(cell_id)<-cell_phen
head(cell_id)
plotCCNetwork(ConNetGNN_data,cell_id,edge.width=10)
```

plotGANetwork — *Visualize gene association network graph of a gene module or pathway at the specified cell phenotype*

Description

Based on the gene set input by the user, `plotGANetwork` functional draws the gene association network in the specified cell phenotype. The node size in the network reflects the activation strength of the gene. The thickness of the edge indicates the strength of interaction between genes.

Usage

```
plotGANetwork(
  network.data,
  cellset,
  geneset,
  rwr.gamma = 0.7,
  vertex.colors = NULL,
  vertex.size = 10,
  vertex.label.cex = 0.8,
  vertex.label.dist = 1,
  vertex.label.color = "black",
  edge.width = 5,
  margin = 0,
  layout = layout_as_star,
  main = NULL,
  plotgraph = TRUE
)
```

Arguments

`network.data` Network data constructed by the `ConNetGNN` function.

`cellset` A vector of cell id. A cell set corresponding to the specified cell phenotype.

`geneset` A vector of gene id. A gene module or pathway.

`rwr.gamma` Restart parameter. Default: `0.7`.

`vertex.colors` The fill color of the vertex. The number of colors should match the number of cell phenotypes. If `NULL (default)`, the system will automatically assign colors.

`vertex.size` The size of the vertex. Default: `10`.

`vertex.label.cex`
The font size for vertex labels. Default: `0.8`.

`vertex.label.dist`
The distance of the label from the center of the vertex. If it is 0 then the label is centered on the vertex. Default: `1`.

`vertex.label.color`
The color of the labels. Default: `black`.

edge.width — The width of the edge. This does not affect the relative size of the edge weights. Default: 5.

margin — The amount of empty space below, over, at the left and right of the plot, it is a numeric vector of length four. Usually values between 0 and 0.5 are meaningful, but negative values are also possible, that will make the plot zoom in to a part of the graph. If it is shorter than four then it is recycled. Default: 0.

layout — Either a function or a numeric matrix. It specifies how the vertices will be placed on the plot. For details, please refer to the igraphPackage. Default: layout_as_star.

main — A main title for the plot.

plotgraph — Whether to draw the picture. Default: TRUE. If FALSE, the image will not be displayed but the network data will be returned in the igraph data format.

Details

plotGANetwork

Value

A graph or list.

Examples

```
require(igraph)

# Load the result of the ConNetGNN function.
data(ConNetGNN_data)

data("Hv_exp")
index<-grep("0h",colnames(Hv_exp))
cellset<-colnames(Hv_exp)[index]
pathways<-load_path_data(system.file("extdata", "KEGG_human.gmt", package = "scapGNN"))
geneset<-pathways[[which(names(pathways)=="Tight junction [PATH:hsa04530]")]]
plotGANetwork(ConNetGNN_data,cellset,geneset,main = "Tight junction [PATH:hsa04530]")
```

---

plotMulPhenGM — *Visualize gene association network graph for activated gene modules under multiple cell phenotypes*

Description

For multiple cell phenotypes, the plotMulPhenGM function will display the activated gene modules for each phenotype and show the connection and status of genes in different cell phenotypes.

Usage

```
plotMulPhenGM(
  data.list,
  network.data,
  vertex.colors = NULL,
  vertex.size = 10,
  vertex.label.cex = 0.8,
  vertex.label.dist = 1,
  vertex.label.color = "black",
  edge.width = 5,
  margin = 0,
  layout = layout_with_lgl,
  legend.position = "bottomright",
  legend.cex = 1.5,
  legend.pt.cex = 3,
  plotgraph = TRUE
)
```

Arguments

`data.list` a list. Each element represents the `cpGModule` function result of a cell phenotype and the names of the lists are the corresponding cell phenotype.

`network.data` Network data constructed by the `ConNetGNN` function.

`vertex.colors` The fill color of the vertex. The number of colors should match the number of cell phenotypes. If `NULL (default)`, the system will automatically assign colors.

`vertex.size` The size of the vertex. Default: `10`.

`vertex.label.cex` The font size for vertex labels. Default: `0.8`.

`vertex.label.dist` The distance of the label from the center of the vertex. If it is 0 then the label is centered on the vertex. Default: `1`.

`vertex.label.color` The color of the labels. Default: `black`.

`edge.width` The width of the edge. This does not affect the relative size of the edge weights. Default: `5`.

`margin` The amount of empty space below, over, at the left and right of the plot, it is a numeric vector of length four. Usually values between 0 and 0.5 are meaningful, but negative values are also possible, that will make the plot zoom in to a part of the graph. If it is shorter than four then it is recycled. Default: `0`.

`layout` Either a function or a numeric matrix. It specifies how the vertices will be placed on the plot. For details, please refer to the `igraph` Package. Default: `layout_with_lgl`.

`legend.position` This places the legend on the inside of the plot frame at the given location. See the `legend()` function for details.

legend.cex	The font size of legend. Default: 1.5.
legend.pt.cex	Expansion factor(s) for the points. Default: 3.
plotgraph	Whether to draw the picture. Default: TRUE. If FALSE, the image will not be displayed but the network data will be returned in the igraph data format.

Details

plotMulPhenGM

If a gene is significantly activated in more than one cell phenotype, we call it a co-activated gene. These co-activated genes are shown on the sector diagram. Each interval of the sector diagram represents the activation strength of the gene in this cell phenotype relative to other cell phenotypes.

Value

A graph or list.

Examples

```
require(igraph)
require(grDevices)
# Load the result of the ConNetGNN function.
data(ConNetGNN_data)
# Obtain cpGModule results for each cell phenotype.
data(H9_0h_cpGM_data)
data(H9_24h_cpGM_data)
data(H9_36h_cpGM_data)
data.list<-list(H9_0h=H9_0h_cpGM_data,H9_24h=H9_24h_cpGM_data,H9_36h=H9_36h_cpGM_data)
plotMulPhenGM(data.list,ConNetGNN_data)
```

Preprocessing *Data preprocessing*

Description

This function is to prepare data for the ConNetGNN function.

Usage

```
Preprocessing(data, parallel.cores = 1, verbose = TRUE)
```

Arguments

data	The input data should be a data frame or a matrix where the rows are genes and the columns are cells. The seurat object are also accepted.
parallel.cores	Number of processors to use when doing the calculations in parallel (default: 2). If parallel.cores=0, then it will use all available core processors unless we set this argument with a smaller number.
verbose	Gives information about each step. Default: TRUE.

Details

Preprocessing

The function is able to interface with the `seurat` framework. The process of `seurat` data processing refers to `Examples`. The input data should be containing hypervariable genes and log-transformed. Left-truncated mixed Gaussian (LTMG) modeling to calculate gene regulatory signal matrix. Positively correlated gene-gene and cell-cell are used as the initial gene correlation matrix and cell correlation matrix.

Value

A list:

orig_dara User-submitted raw data, rows are highly variable genes and columns are cells.

cell_features Cell feature matrix.

gene_features Gene feature matrix.

ltmg_matrix Gene regulatory signal matrix for LTMG.

cell_adj The adjacency matrix of the cell correlation network.

gene_adj The adjacency matrix of the gene correlation network.

Examples

```
# Load dependent packages.
# require(coop)

# Seurat data processing.
# require(Seurat)

# Load the PBMC dataset (Case data for seurat)
# pbmc.data <- Read10X(data.dir = "../data/pbmc3k/filtered_gene_bc_matrices/hg19/")

# Our recommended data filtering is that only genes expressed as non-zero in more than
# 1% of cells, and cells expressed as non-zero in more than 1% of genes are kept.
# In addition, users can also filter mitochondrial genes according to their own needs.
# pbmc <- CreateSeuratObject(counts = pbmc.data, project = "case",
#                                   min.cells = 3, min.features = 200)
# pbmc[["percent.mt"]] <- PercentageFeatureSet(pbmc, pattern = "^MT-")
# pbmc <- subset(pbmc, subset = nFeature_RNA > 200 & nFeature_RNA < 2500 & percent.mt < 5)

# Normalizing the data.
# pbmc <- NormalizeData(pbmc, normalization.method = "LogNormalize")

# Identification of highly variable features.
# pbmc <- FindVariableFeatures(pbmc, selection.method = 'vst', nfeatures = 2000)

# Run Preprocessing.
# Prep_data <- Preprocessing(pbmc)
```

```
# Users can also directly input data
# in data frame or matrix format
# containing highly variable genes.
data("Hv_exp")
Hv_exp <- Hv_exp[,1:20]
Hv_exp <- Hv_exp[which(rowSums(Hv_exp) > 0),]
Prep_data <- Preprocessing(Hv_exp[1:10,])
```

Pure_CDF *Pure_CDF*

Description

The internal functions of the scapGNN package.

Usage

```
Pure_CDF(Vec)
```

Arguments

Vec	Internal parameters.

Details

Pure_CDF

RNA_ATAC_IntNet *Results of InteNet() for integrating scRNA-seq and scATAC-seq data.*

Description

An integrated network of scRNA-seq and scATAC-seq data from SNARE-seq.

Usage

```
RNA_ATAC_IntNet
```

Format

a list of three adjacency matrices.

Examples

```
data(RNA_ATAC_IntNet)
```

RNA_net — *Results of ConNetGNN() for scRNA-seq data from SNARE-seq dataset*

Description

A list to store the gene association network of scRNA-seq data. Case data from the SNARE-seq dataset.

Usage

```
RNA_net
```

Format

a list of three adjacency matrices.

Examples

```
data(RNA_net)
```

RunLTMG — *Run Left-truncated mixed Gaussian*

Description

Functional implementation of Left-truncated mixed Gaussian. The internal functions of the `scapGNN` package.

Usage

```
.RunLTMG(object, Gene_use = NULL, k = 5, verbose, seed = 123)

RunLTMG(object, Gene_use = NULL, k = 5, verbose, seed = 123)

## S4 method for signature 'LTMG'
RunLTMG(object, Gene_use = NULL, k = 5, verbose, seed = 123)
```

Arguments

`object`	A LTMG object
`Gene_use`	using X numebr of top variant gene. input a number, recommend 2000.
`k`	Constant, defaults 5.
`verbose`	Gives information about each step.
`seed`	Random number generator seed.

Details

RunLTMG

For more information, please refer to: `DOI: 10.1093/nar/gkz655` and `https://github.com/zy26/LTMGSCA`.

Value

A list contains raw input data and LTMG results.

`RWR` *Function that performs a random Walk with restart (RWR) on a given graph*

Description

Function that performs a random Walk with restart (RWR) on a given graph

Usage

```
RWR(W, ind.positives, gamma = 0.6)
```

Arguments

`W`	: adjacency matrix of the graph
`ind.positives`	: indices of the "core" positive examples of the graph. They represent to the indices of W corresponding to the positive examples
`gamma`	: restart parameter (def: 0.6)

Value

a list with three elements: - p : the probability at the steady state - ind.positives : indices of the "core" positive examples of the graph (it is equal to the same input parameter - n.iter : number of performed iterations

a vector

scPathway *Infer pathway activation score matrix at single-cell resolution*

Description

Calculate pathway activity score of single-cell by random walk with restart (RWR).

Usage

```
scPathway(
  network.data,
  gmt.path = NULL,
  pathway.min = 10,
  pathway.max = 500,
  nperm = 50,
  parallel.cores = 2,
  rwr.gamma = 0.7,
  normal_dist = TRUE,
  seed = 1217,
  verbose = TRUE
)
```

Arguments

`network.data`	The input network data is the result from the `ConNetGNN` function.
`gmt.path`	Pathway database in `GMT` format.
`pathway.min`	Minimum size (in genes) for pathway to be considered. Default: `10`.
`pathway.max`	Maximum size (in genes) for database gene sets to be considered. Default: `500`.
`nperm`	Number of random permutations. Default: `50`. We recommend the setting of 100.
`parallel.cores`	Number of processors to use when doing the calculations in parallel (default: `2`). If `parallel.cores=0`, then it will use all available core processors unless we set this argument with a smaller number.
`rwr.gamma`	Restart parameter. Default: `0.7`.
`normal_dist`	Whether to use pnorm to calculate P values. Default: `TRUE`.Note that if normal_dist is FALSE, we need to increase nperm (we recommend 100).
`seed`	Random number generator seed.
`verbose`	Gives information about each step. Default: `TRUE`.

Details

scPathway

The `scPathway` function integrates the results of ConNetGNN into a gene-cell association network. The genes included in each pathway are used as a restart set in the gene-cell association network to calculate the strength of its association with each cell through RWR. Perturbation analysis was performed to remove noise effects in the network and to obtain the final single-cell pathway activity score matrix.

Value

A matrix of single-cell pathway activity score.

Examples

```
require(parallel)
require(utils)
# Load the result of the ConNetGNN function.
data(ConNetGNN_data)
kegg.path<-system.file("extdata", "KEGG_human.gmt", package = "scapGNN")
# We recommend the use of a compiler.
# The compiler package can be used to speed up the operation.
# library(compiler)
# scPathway<- cmpfun(scPathway)
scPathway_data<-scPathway(ConNetGNN_data,gmt.path=kegg.path,
                          pathway.min=25,nperm=2,parallel.cores=1)
```

---

scPathway_data   *Single cell pathway activity matrix*

Description

Results of scPathway() function.

Usage

```
scPathway_data
```

Format

a matrix.

Examples

```
data(scPathway_data)
```

Index