Package

April 20, 2019

Title ScRNA-seq workflow CONCLUS - from CONsensus CLUSters to a meaningful CONCLUSion.

Version 0.0.0.9000

Authors Polina Pavlovich, Konstantin Chukreev, Christophe Lancrin

Maintainer Polina Pavlovich <pavlovich@phystech.edu>

Description CONCLUS is a tool for robust clustering, positive marker features selection and collecting info about marker genes using web scraping from MGI, UniProt and NCBI websites. The tool was developed to distinguish not only cell types from different tissues but also characterize heterogeneity within a tissue or characterize stages of a differentiation process.

Depends R (>= 3.4)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1.9000

Imports BiocParallel (>= 1.14.2), scran (>= 1.8.4), ggplot2 (>= 3.0.0), scater (>= 1.8.4), Matrix (>= 1.2-14), monocle (>= 2.8.0), SingleCellExperiment (>= 1.2.0), dbscan (>= 1.1-2), pheatmap (>= 1.0.10), fpc (>= 2.1-11.1), dynamicTreeCut (>= 1.63-1), factoextra (>= 1.0.5), digest (>= 0.6.15), KEGGREST (>= 1.20.2), RColorBrewer (>= 1.1-2), doParallel (>= 1.0.14), matrixStats (>= 0.54.0), AnnotationDbi (>= 1.42.1), dplyr (>= 0.7.6), biomaRt (>= 2.36.1), org.Mm.eg.db (>= 3.6.0), xlsx (>= 0.6.1), DataCombine (>= 0.2.21), grDevices, S4Vectors (>= 0.18.3), Biobase (>= 2.40.0), foreach (>= 1.4.4)

Suggests knitr, rmarkdown, prettydoc

VignetteBuilder knitr

R topics documented:

`addClusteringManually` *addClusteringManually*

Description

The function replaces the content of the column "clusters" in the colData(sceObject) with the clustering provided in the user table. The function will return the sceObject with cells which intersect with the cells from the input table.

Usage

```
addClusteringManually(fileName, sceObject, dataDirectory, experimentName,
  columnName = "clusters")
```

Arguments

`fileName`	a file with the clustering solution (for example, from previous CONCLUS runs).
`sceObject`	a SingleCellExperiment object with your experiment.
`dataDirectory`	output directory (supposed to be the same for one experiment during the workflow).
`experimentName`	name of the experiment which appears in filenames (supposed to be the same for one experiment during the workflow).
`columnName`	name of the column with the clusters.

Value

A SingleCellExperiment object with the created/renewed column "clusters" in the colData(sceObject).

calculateClustersSimilarity *Having cells similarity, calculate clusters similarity.*

Description

Having cells similarity, calculate clusters similarity.

Usage

```
calculateClustersSimilarity(cellsSimilarityMatrix, sceObject,
  clusteringMethod)
```

Arguments

cellsSimilarityMatrix	a similarity matrix, one of the results of conclus::clusterCellsInternal() function.
sceObject	a SingleCellExperiment object with your experiment.
clusteringMethod	a clustering methods passed to hclust() function.

Value

A list contating the cluster similarity matrix and cluster names (order).

choosePalette *Choose palette for a plot.*

Description

It is an internal function usually applied for choosing the palette for clusters. Depending if the number of clusters is more than 12 or not, one of two built-in palettes will be applied. If you give your vector of colors, the function will not change them. If the number of clusters is more than 26, it will copy colors to get the needed length of the palette.

Usage

```
choosePalette(colorPalette, clustersNumber)
```

Arguments

colorPalette	Either "default" or a vector of colors, for example c("yellow", "#CC79A7").
clustersNumber	number of clusters in the output palette.

Value

Color palette with the number of colors equal to the clusterNumber parameter.

`clusterCellsInternal` *Cluster cells and get similarity matrix of cells.*

Description

The function returns consensus clusters by using hierarchical clustering on the similarity matrix of cells. It provides two options: to specify an exact number of clusters (with clusterNumber parameter) or to select the depth of splitting (deepSplit parameter).

Usage

```
clusterCellsInternal(dbscanMatrix, sceObject, clusterNumber = 0,
  deepSplit, cores = 14, clusteringMethod = "ward.D2")
```

Arguments

`dbscanMatrix`	an output matrix of conclus::runDBSCAN() function.
`sceObject`	a SingleCellExperiment object with your experiment.
`clusterNumber`	a parameter, specifying the exact number of cluster.
`deepSplit`	a parameter, specifying how deep we will split the clustering tree. It takes integers from 1 to 4.
`cores`	maximum number of jobs that CONCLUS can run in parallel.
`clusteringMethod`	a clustering methods passed to hclust() function.

Value

A SingleCellExperiment object with modified/created "clusters" column in the colData, and cells similarity matrix.

`exportClusteringResults` *exportClusteringResults*

Description

The function saves clustering results into a table. Row names are cell names in the same order as in the sceObject.

Usage

```
exportClusteringResults(sceObject, dataDirectory, experimentName, fileName)
```

Arguments

sceObject	a SingleCellExperiment object with your experiment.
dataDirectory	output directory (supposed to be the same for one experiment during the workflow).
experimentName	name of the experiment which appears at the beginning of the file name (supposed to be the same for one experiment during the workflow).
fileName	the rest of output file name.

---

exportMatrix *Export matrix to a file.*

Description

The function allows you to export a matrix to a .csv file with a hard-coded filename (according to experimentName) in the "dataDirectory/output_tables" directory for further analysis.

Usage

```
exportMatrix(matrix, dataDirectory, experimentName, name)
```

Arguments

matrix	your matrix (e.g., expression matrix)
dataDirectory	CONCLUS output directory for a given experiment (supposed to be the same for one experiment during the workflow).
experimentName	name of the experiment which will appear at the beginning of the filenames (supposed to be the same for one experiment during the workflow).
name	name of the file. Will be placed after the experimentName header.

---

generateTSNECoordinates *Generate and save t-SNE coordinates with selected parameters.*

Description

The function generates several t-SNE coordinates based on given perplexity and ranges of PCs. Final number of t-SNE plots is length(PCs)*length(perplexities) It writes coordinates in "dataDirectory/tsnes" subfolder.

Usage

```
generateTSNECoordinates(sceObject, dataDirectory, experimentName,
  randomSeed = 42, cores = 14, PCs = c(4, 6, 8, 10, 20, 40, 50),
  perplexities = c(30, 40))
```

Arguments

sceObject	a SingleCellExperiment object with your experiment.
dataDirectory	output directory for CONCLUS (supposed to be the same for one experiment during the workflow).
experimentName	name of the experiment which will appear in filenames (supposed to be the same for one experiment during the workflow).
randomSeed	random seed for reproducibility.
cores	maximum number of jobs that CONCLUS can run in parallel.
PCs	a vector of first principal components. For example, to take ranges 1:5 and 1:10 write c(5, 10).
perplexities	a vector of perplexity (t-SNE parameter).

Value

An object with t-SNE results (coordinates for each plot).

---

getGenesInfo *Collect genes information to one table.*

Description

The function takes a data frame containing gene symbols and (or) ENSEMBL IDs and returns a data frame with such information as gene name, feature type, chromosome, gene IDs in different annotations, knockout information from MGI, a summary from NCBI and UniProt, and whether or not a gene belongs to GO terms containing proteins on the cell surface or involved in secretion.

Usage

```
getGenesInfo(genes, databaseDir = system.file("extdata", package =
  "conclus"), groupBy = "clusters", orderGenes = "initial",
  getUniprot = TRUE, silent = FALSE, coresGenes = 20)
```

Arguments

genes	a data frame with the first column called "geneName" containing gene symbols and (or) ENSEMBL IDs. Other columns are optional. For example, the second column could be "clusters" with the name of the cluster for which the gene is a marker.
databaseDir	a path to the database provided with CONCLUS called "Mmus_gene_database_secretedMol.tsv".
groupBy	a column in the input table used for grouping the genes in the output tables. This option is useful if a table contains genes from different clusters.
orderGenes	if "initial" then the order of genes will not be changed.
getUniprot	boolean, whether to get information from UniProt or not. Default is TRUE. Sometimes, the connection to the website is not reliable. If you tried a couple of times and it failed, select FALSE.
silent	whether to show messages from intermediate steps or not.
coresGenes	maximum number of jobs that the function can run in parallel.

Value

Returns a data frame.

`getMarkerGenes` *Get top N marker genes from each cluster.*

Description

This function reads results of conclus::rankGenes() from "dataDirectory/marker_genes" and selects top N markers for each cluster.

Usage

```
getMarkerGenes(dataDirectory, sceObject, genesNumber = 14,
  experimentName, removeDuplicates = TRUE)
```

Arguments

`dataDirectory`	output directory for a run of CONCLUS (supposed to be the same for one experiment during the workflow).
`sceObject`	a SingleCellExperiment object with your experiment.
`genesNumber`	top N number of genes to get from one cluster.
`experimentName`	name of the experiment which appears in filenames (supposed to be the same for one experiment during the workflow).
`removeDuplicates`	boolean, if duplicated genes must be deleted or not.

Value

A data frame where the first columns are marker genes ("geneName") and the second column is the groups ("clusters").

`initialisePath` *Create all needed directories for CONCLUS output.*

Description

Create all needed directories for CONCLUS output.

Usage

```
initialisePath(dataDirectory)
```

Arguments

`dataDirectory`	output directory for a given CONCLUS run (supposed to be the same for one experiment during the workflow).

normaliseCountMatrix *normaliseCountMatrix*

Description

Create a SingleCellExperiment object and perform normalization. The same as conclus::normalizeCountMatrix.

Usage

```
normaliseCountMatrix(countMatrix, species, method = "default",
  sizes = c(20, 40, 60, 80, 100), rowData = NULL, colData = NULL,
  alreadyCellFiltered = FALSE, runQuickCluster = TRUE,
  databaseDir = system.file("extdata", package = "conclus"))
```

Arguments

countMatrix	a matrix with non-normalised gene expression.
species	either 'mmu' or 'human'.
method	a method of clustering: available option is "default" using scran and scater.
sizes	a vector of size factors from scran::computeSumFactors() function.
rowData	a data frame with information about genes
colData	a data frame with information about cells
alreadyCellFiltered	if TRUE, cells quality check and filtering will not be applied. However, the function may delete some cells if they have negative size factors after scran::computeSumFactors.
runQuickCluster	if scran::quickCluster() function must be applied. Usually, it allows to improve normalization for medium-size count matrices. However, it is not recommended for datasets with less than 200 cells and may take too long for datasets with more than 10000 cells.
databaseDir	a path to annotation database provided with CONCLUS called "Mmus_gene_database_secretedMol.ts (only for MusMusculus 'mmu'). The function will work also without the database but slower because it will retrieve genes info from biomaRt.

Value

A SingleCellExperiment object with normalized gene expression, colData, and rowData.

`plotCellHeatmap` *Save markers heatmap.*

Description

This function plots heatmap with marker genes on rows and clustered cells on columns.

Usage

```
plotCellHeatmap(markersClusters, sceObject, dataDirectory, experimentName,
  fileName, meanCentered = TRUE, colorPalette = "default",
  statePalette = "default", clusteringMethod = "ward.D2",
  orderClusters = FALSE, orderGenes = FALSE, returnPlot = FALSE,
  saveHeatmapTable = FALSE, width = 10, height = 8.5, ...)
```

Arguments

`markersClusters`	a data frame where the first column is "geneName" containing genes names from sceObject, and the second column is corresponding "clusters". All names from that column must come from the column "clusters" in the colData(sceObject). The data frame can be obtained from conclus::getMarkerGenes() function or created manually.
`sceObject`	a SingleCellExperiment object with your experiment.
`dataDirectory`	output directory of a given CONCLUS run (supposed to be the same for one experiment during the workflow).
`experimentName`	name of the experiment which appears in filenames (supposed to be the same for one experiment during the workflow).
`fileName`	name of the ouput file
`meanCentered`	boolean, should mean centering be applied to the expression data or not.
`colorPalette`	"default" or a vector of colors for the column "clusters" in the colData, for example c("yellow", "#CC79A7").
`statePalette`	"default" or a vector of colors for the column "state" in the colData, for example c("yellow", "#CC79A7").
`clusteringMethod`	a clustering methods passed to hclust() function.
`orderClusters`	boolean, should the heatmap be structured by clusters.
`orderGenes`	boolean, should the heatmap be structured by genes.
`returnPlot`	boolean, whether to return a ggplot object with the plot or not.
`saveHeatmapTable`	boolean, whether to save the expression matrix used for heatmap into a .csv file or not. The file will be saved into 'dataDirectory/output_tables' with the same name as the .pdf plot.
`width`	plot width.
`height`	plot height.
`...`	other parameters from pdf() and pheatmap() functions.

Value

A ggplot object of the plot if needed. The function saves pdf in "dataDirectiry/pictures" folder.

`plotCellSimilarity` *Save a cells similarity matrix.*

Description

This function plots similarity matrix as a heatmap, so one can see similarity between parts of different clusters.

Usage

```
plotCellSimilarity(sceObject, cellsSimilarityMatrix, dataDirectory,
  experimentName, colorPalette = "default", statePalette = "default",
  clusteringMethod = "ward.D2", orderClusters = FALSE,
  plotPDF = TRUE, returnPlot = FALSE, width = 7, height = 6, ...)
```

Arguments

`sceObject`	a SingleCellExperiment object with your experiment.
`cellsSimilarityMatrix`	an output matrix from the conclus::clusterCellsInternal() function.
`dataDirectory`	output directory for CONCLUS (supposed to be the same for one experiment during the workflow).
`experimentName`	name of the experiment which will appear in filenames (supposed to be the same for one experiment during the workflow).
`colorPalette`	"default" or a vector of colors for the column "clusters" in the colData, for example c("yellow", "#CC79A7").
`statePalette`	"default" or a vector of colors for the column "state" in the colData, for example c("yellow", "#CC79A7").
`clusteringMethod`	a clustering methods passed to hclust() function.
`orderClusters`	boolean, order clusters or not.
`plotPDF`	if TRUE export to pdf, if FALSE export to png. FALSE is recommended for datasets with more than 2500 cells due to large pdf file size.
`returnPlot`	boolean, return plot or not. Default if FALSE.
`width`	plot width.
`height`	plot height.
`...`	other parameters of pdf(), pheatmap() and png() functions.

Value

A ggplot object or nothing (depends on the returnPlot parameter). It saves the pdf in "dataDirectory/pictures" folder.

`plotClusteredTSNE` *Plot t-SNE. Addtionally, it can highlight clusters or states.*

Description

Plot t-SNE. Addtionally, it can highlight clusters or states.

Usage

```
plotClusteredTSNE(sceObject, dataDirectory, experimentName,
  tSNEresExp = "", colorPalette = "default", PCs = c(4, 6, 8, 10, 20,
  40, 50), perplexities = c(30, 40), columnName = "clusters",
  returnPlot = FALSE, width = 6, height = 5, ...)
```

Arguments

`sceObject`	a SingleCellExperiment object with your experiment.
`dataDirectory`	output directory for CONCLUS (supposed to be the same for one experiment during the workflow).
`experimentName`	name of the experiment which will appear in filenames (supposed to be the same for one experiment during the workflow).
`tSNEresExp`	if t-SNE coordinates were generated in a different CONCLUS run, you can use them without renaming the files. Please copy tsnes folder from the source run to the current one and write that experimentName in the tSNEresExp argument.
`colorPalette`	"default" or a vector of colors for the column "clusters" in the colData, for example c("yellow", "#CC79A7").
`PCs`	vector of PCs (will be specified in filenames).
`perplexities`	vector of perplexities (will be specified in filenames).
`columnName`	name of the column to plot on t-SNE dimensions.
`returnPlot`	boolean, return plot or not.
`width`	plot width.
`height`	plot height.
`...`	other arguments of the pdf() function.

Value

A ggplot object or nothing (depends on the returnPlot parameter).

`plotClustersSimilarity` *Save a similarity cluster matrix.*

Description

Save a similarity cluster matrix.

Usage

```
plotClustersSimilarity(clustersSimilarityMatrix, sceObject, dataDirectory,
  experimentName, colorPalette, statePalette, clusteringMethod,
  returnPlot = FALSE, width = 7, height = 5.5, ...)
```

Arguments

`clustersSimilarityMatrix`	a matrix, result of conclus::calculateClustersSimilarity() function.
`sceObject`	a SingleCellExperiment object with your experiment.
`dataDirectory`	output directory for CONCLUS (supposed to be the same for one experiment during the workflow).
`experimentName`	name of the experiment which will appear in filenames (supposed to be the same for one experiment during the workflow).
`colorPalette`	"default" or a vector of colors for the column "clusters" in the colData, for example c("yellow", "#CC79A7").
`statePalette`	"default" or a vector of colors for the column "state" in the colData, for example c("yellow", "#CC79A7").
`clusteringMethod`	a clustering methods passed to hclust() function.
`returnPlot`	boolean, return plot or not.
`width`	plot width.
`height`	plot height.
`...`	other parameters of pdf() and pheatmap() functions.

Value

A ggplot object or nothing (depends on returnPlot parameter). It saves the pdf in "dataDirectory/pictures" folder.

plotGeneExpression *plotGeneExpression*

Description

The function saves a t-SNE plot colored by expression of a given gene. Warning: filename with t-SNE results is hardcoded, so please don't rename the output file.

Usage

```
plotGeneExpression(geneName, experimentName, dataDirectory,
  graphsDirectory = "pictures", sceObject, tSNEpicture = 1,
  commentName = "", palette = c("grey", "red", "#7a0f09", "black"),
  returnPlot = FALSE, savePlot = TRUE, alpha = 1, limits = NA,
  pointSize = 1, width = 6, height = 5, ...)
```

Arguments

geneName	name of the gene you want to plot.
experimentName	name of the experiment which appears in filenames (supposed to be the same for one experiment during the workflow).
dataDirectory	output directory for CONCLUS (supposed to be the same for one experiment during the workflow).
graphsDirectory	name of the subdirectory where to put graphs. Default is "dataDirectory/pictures".
sceObject	a SingleCellExperiment object with your experiment.
tSNEpicture	number of the picture you want to use for plotting. Please check "dataDirectory/tsnes" or "dataDirectory/pictures/tSNE_pictures/clusters" to get the number, it is usually from 1 to 14.
commentName	comment you want to specify in the filename.
palette	color palette for the legend.
returnPlot	boolean, should the function return a ggplot object or not.
savePlot	boolean, should the function export the plot to pdf or not.
alpha	opacity of the points of the plot.
limits	range of the gene expression shown in the legend. This option allows generating t-SNE plots with equal color scale to compare the expression of different genes. By default, limits are the range of expression of a selected gene.
pointSize	size of the point.
width	plot width.
height	plot height.
...	other parameters of the pdf() function.

Value

A ggplot object of the plot if needed.

rankGenes *Rank marker genes by statistical significance.*

Description

This function searches marker genes for each cluster. It saves tables in the "dataDirectory/marker_genes" directory, one table per cluster.

Usage

```
rankGenes(sceObject, clustersSimilarityMatrix, dataDirectory,
  experimentName, column = "clusters")
```

Arguments

sceObject	a SingleCellExperiment object with your experiment.
clustersSimilarityMatrix	matrix, result of conclus::calculateClustersSimilarity() function.
dataDirectory	output directory for CONCLUS (supposed to be the same for one experiment during the workflow).
experimentName	name of the experiment which will appear in filenames (supposed to be the same for one experiment during the workflow).
column	name of the column with a clustering result.

runClustering *DBSCAN clustering on t-SNE results.*

Description

This function provides consensus DBSCAN clustering based on the results of t-SNE. You can tune algorithm parameters in options to get the number of clusters you want.

Usage

```
runClustering(tSNEResults, sceObject, dataDirectory, experimentName,
  epsilon = c(1.3, 1.4, 1.5), minPoints = c(3, 4), k = 0,
  deepSplit = 4, clusteringMethod = "ward.D2", cores = 14,
  deleteOutliers = TRUE, PCs = c(4, 6, 8, 10, 20, 40, 50),
  perplexities = c(30, 40), randomSeed = 42)
```

Arguments

tSNEResults	the result of conclus::generateTSNECoordinates() function.
sceObject	a SingleCellExperiment object with your experiment.
dataDirectory	output directory of a given CONCLUS run (supposed to be the same for one experiment during the workflow).
experimentName	name of the experiment which appears in filenames (supposed to be the same for one experiment during the workflow).
epsilon	a parameter of fpc::dbscan() function.
minPoints	a parameter of fpc::dbscan() function.
k	preferred number of clusters. Alternative to deepSplit.
deepSplit	intuitive level of clustering depth. Options are 1, 2, 3, 4.
clusteringMethod	a clustering methods passed to hclust() function.
cores	maximum number of jobs that CONCLUS can run in parallel.
deleteOutliers	Whether cells which were often defined as outliers by dbscan must be deleted. It will require recalculating of the similarity matrix of cells. Default is FALSE. Usually those cells appear in an "outlier" cluster and can be easier distinguished and deleted later if necessary.
PCs	a vector of first principal components. For example, to take ranges 1:5 and 1:10 write c(5, 10).
perplexities	a vector of perplexity for t-SNE.
randomSeed	random seed for reproducibility.

Value

A list containing filtered from outliers SingleCellExperiment object and cells similarity matrix.

---

runCONCLUS *Run CONCLUS in one click*

Description

This function performs core CONCLUS workflow. It generates PCA and t-SNE coordinates, runs DBSCAN, calculates similarity matrices of cells and clusters, assigns cells to clusters, searches for positive markers for each cluster. The function saves plots and tables into dataDirectory.

Usage

```
runCONCLUS(sceObject, dataDirectory, experimentName,
  colorPalette = "default", statePalette = "default",
  clusteringMethod = "ward.D2", epsilon = c(1.3, 1.4, 1.5),
  minPoints = c(3, 4), k = 0, PCs = c(4, 6, 8, 10, 20, 40, 50),
  perplexities = c(30, 40), randomSeed = 42, deepSplit = 4,
  preClustered = F, orderClusters = FALSE, cores = 14,
  plotPDFcellSim = TRUE, deleteOutliers = TRUE,
  tSNEalreadyGenerated = FALSE, tSNEresExp = "")
```

Arguments

sceObject	a SingleCellExperiment object with your data.
dataDirectory	CONCLUS will create this directory if it doesn't exist and store there all output files.
experimentName	most of output file names of CONCLUS are hardcoded. experimentName will stay at the beginning of each output file name to distinguish different runs easily.
colorPalette	a vector of colors for clusters.
statePalette	a vector of colors for states.
clusteringMethod	a clustering methods passed to hclust() function.
epsilon	a parameter of fpc::dbscan() function.
minPoints	a parameter of fpc::dbscan() function.
k	preferred number of clusters. Alternative to deepSplit. A parameter of cutree() function.
PCs	a vector of first principal components. For example, to take ranges 1:5 and 1:10 write c(5, 10).
perplexities	a vector of perplexity for t-SNE.
randomSeed	random seed for reproducibility.
deepSplit	intuitive level of clustering depth. Options are 1, 2, 3, 4.
preClustered	if TRUE, it will not change the column clusters after the run. However, it will anyway run DBSCAN to calculate similarity matrices.
orderClusters	can be either FALSE (default) of "name". If "name", clusters in the similarity matrix of cells will be ordered by name.
cores	maximum number of jobs that CONCLUS can run in parallel.
plotPDFcellSim	if FALSE, the similarity matrix of cells will be saved in png format. FALSE is recommended for count matrices with more than 2500 cells due to large pdf file size.
deleteOutliers	whether cells which were often defined as outliers by dbscan must be deleted. It will require recalculating of the similarity matrix of cells. Default is FALSE. Usually those cells form a separate "outlier" cluster and can be easier distinguished and deleted later if necessary.
tSNEalreadyGenerated	if you already ran CONCLUS ones and have t-SNE coordinated saved You can set TRUE to run the function faster since it will skip the generation of t-SNE coordinates and use the stored ones. Option TRUE requires t-SNE coordinates to be located in your 'dataDirectory/tsnes' directory.
tSNEresExp	experimentName of t-SNE coordinates which you want to use. This argument allows copying and pasting t-SNE coordinates between different CONCLUS runs without renaming the files.

Value

A SingleCellExperiment object.

runDBSCAN *Run clustering iterations with selected parameters using DBSCAN.*

Description

This function returns a matrix of clustering iterations of DBSCAN.

Usage

```
runDBSCAN(tSNEResults, sceObject, dataDirectory, experimentName,
  cores = 14, epsilon = c(1.3, 1.4, 1.5), minPoints = c(3, 4))
```

Arguments

tSNEResults	results of conclus::generateTSNECoordinates() function.
sceObject	a SingleCellExperiment object with your experiment.
dataDirectory	output directory for CONCLUS (supposed to be the same for one experiment during the workflow).
experimentName	name of the experiment which will appear in filenames (supposed to be the same for one experiment during the workflow).
cores	maximum number of jobs that CONCLUS can run in parallel.
epsilon	a fpc::dbscan() parameter.
minPoints	a fpc::dbscan() parameter.

Value

A matrix of DBSCAN results.

saveGenesInfo *Save gene information into a table or tables for multiple inputs.*

Description

This function runs conclus::getGenesInfo() function for all tables into the inputDir and saves the result into the outputDir.

Usage

```
saveGenesInfo(dataDirectory = "", inputDir = "", outputDir = "",
  pattern = "", databaseDir = system.file("extdata", package =
  "conclus"), sep = ";", header = TRUE, startFromFile = 1,
  groupBy = "clusters", orderGenes = "initial", getUniprot = TRUE,
  silent = FALSE, coresGenes = 20)
```

Arguments

dataDirectory	a directory with CONCLUS output. You can specify either dataDirectory, then inputDir and outputDir will be hardcoded, or inputDir and outputDir only. The first is recommended during running CONCLUS workflow when the second option is comfortable when you created input tables with genes manually.
inputDir	input directory containing text files. These files can be obtained by applying conclus::saveMarkersLists() function or created manually. Each file must be a data frame with the first column called "geneName" containing gene symbols and (or) ENSEMBL IDs.
outputDir	output directory.
pattern	a pattern of file names to take.
databaseDir	a path to the database "Mmus_gene_database_secretedMol.tsv". It is provided with the conclus package.
sep	a parameter of read.delim() function.
header	whether or not your input files have a header.
startFromFile	number of the input file to start with. The function approaches files one by one. It uses web scraping method to collect publicly available info from MGI, NCBI and UniProt websites. Sometimes, if the Internet connection is not reliable, the function can drop. In this case, it is comfortable to start from the failed file and not to redo the previous ones.
groupBy	a column in the input table used for grouping the genes in the output tables.
orderGenes	if "initial" then the order of genes will not be changed.
getUniprot	boolean, whether to get information from UniProt or not. Default is TRUE. Sometimes, the connection to the website is not reliable. If you tried a couple of times and it failed, select FALSE.
silent	whether to show messages from intermediate steps or not.
coresGenes	maximum number of jobs that the function can run in parallel.

Value

It saves text files either in the 'dataDirectory/marker_genes/saveGenesInfo' or outputDir depending on whether you specify dataDirectory or (inpitDir and outputDir) explicitly.

---

saveMarkersLists *Save top N marker genes for each cluster into a format suitable for conclus::saveGenesInfo() function.*

Description

The function takes the output files of conclus::rankGenes(), extracts top N markers and saves them into the first "geneName" column of the output table. The second column "clusters" contains the name of the corresponding cluster.

Usage

```
saveMarkersLists(experimentName, dataDirectory,
  inputDir = file.path(dataDirectory, "marker_genes"),
  outputDir = file.path(dataDirectory,
  paste0("marker_genes/markers_lists")), pattern = "genes.tsv",
  Ntop = 100)
```

Arguments

experimentName	name of the experiment which appears at the beginning of the file name (supposed to be the same for one experiment during the workflow).
dataDirectory	experiment directory (supposed to be the same for one experiment during the workflow).
inputDir	input directory, usually "marker_genes" created automatically after conclus::runCONCLUS().
outputDir	output directory.
pattern	a pattern of the input file names to take.
Ntop	number of top markers to take from each cluster.

Value

It saves files into the outputDir. The number of files is equal to the number of clusters.

---

testClustering *To check one iteration of clustering before running full workflow CONCLUS.*

Description

This function generates a single clustering iteration of CONCLUS to check whether chosen parameters for dbscan are suitable for your data.

Usage

```
testClustering(sceObject, dataDirectory, experimentName,
  dbscanEpsilon = 1.4, minPts = 5, perplexities = c(30),
  PCs = c(4), randomSeed = 42, width = 7, height = 7, ...)
```

Arguments

sceObject	a SingleCellExperiment object with your experiment.
dataDirectory	output directory (supposed to be the same for one experiment during the workflow).
experimentName	name of the experiment which will appear in filenames (supposed to be the same for one experiment during the workflow).
dbscanEpsilon	a parameter of fpc::dbscan() function.

`minPts`	a parameter of fpc::dbscan() function.
`perplexities`	vector of perplexities (t-SNE parameter).
`PCs`	a vector of PCs for plotting.
`randomSeed`	random seed for reproducibility.
`width`	plot width.
`height`	plot height.
`...`	other pdf() arguments.

Value

t-SNE results, a distance graph plot, a t-SNE plot colored by test clustering solution.

Index