Package ‘AUCell’

May 1, 2024

Type Package

Title AUCell: Analysis of ‘gene set’ activity in single-cell RNA-seq data (e.g. identify cells with specific gene signatures)

Version 1.26.0

Date 2024-03-09

Author Sara Aibar, Stein Aerts. Laboratory of Computational Biology. VIB-KU Leuven Center for Brain & Disease Research. Leuven, Belgium.

Maintainer Gert Hulselmans <Gert.Hulselmans@kuleuven.be>

Description AUCell allows to identify cells with active gene sets (e.g. signatures, gene modules...) in single-cell RNA-seq data. AUCell uses the "Area Under the Curve" (AUC) to calculate whether a critical subset of the input gene set is enriched within the expressed genes for each cell. The distribution of AUC scores across all the cells allows exploring the relative expression of the signature. Since the scoring method is ranking-based, AUCell is independent of the gene expression units and the normalization procedure. In addition, since the cells are evaluated individually, it can easily be applied to bigger datasets, subsetting the expression matrix if needed.

URL http://scenic.aertslab.org

Imports DelayedArray, DelayedMatrixStats, data.table, graphics, grDevices, GSEABase, Matrix, methods, mixtools, R.utils, stats, SummarizedExperiment, BiocGenerics, utils

Enhances doMC, doRNG, doParallel, foreach

Suggests Biobase, BiocStyle, doSNOW, dynamicTreeCut, DT, GEOquery, knitr, NMF, plyr, R2HTML, rmarkdown, reshape2, plotly, Rtsne, testthat, zoo

License GPL-3

biocViews SingleCell, GeneSetEnrichment, Transcriptomics, Transcription, GeneExpression, WorkflowStep, Normalization

VignetteBuilder knitr

RoxygenNote 7.2.0

git_url https://git.bioconductor.org/packages/AUCell
aucellResults-class

Wrapper to the matrix that stores the AUC or the cell rankings.

Description

This class extends SummarizedExperiment to contain the AUC matrix and cell rankings (as 'assays').

The results are stored in the assays slot, but they can be accessed through the regular methods (i.e. nrow, rownames...)

Types:
- "AUC": The assays contains the AUC for the gene-sets (or region-sets) & cells.
- "ranking": The assays contains the gene rankings for each cell.
Usage

```r
## S4 method for signature 'aucellResults'
show(object)

getAUC(object)

## S4 method for signature 'aucellResults'
getRanking(object)

## S4 method for signature 'aucellResults'
cbind(..., deparse.level = 1)

## S4 method for signature 'aucellResults'
rbind(..., deparse.level = 1)
```

Arguments

- `object`: Results from `AUCell_buildRanking`
- `...`: (Only for `cbind` or `AUCell_calcAUC`)
- `deparse.level`: (Only for `cbind`)

Value

- `show`: Prints a summary of the object
- `getAUC`: Returns the matrix containing the AUC
- `getRanking`: Returns the matrix containing the rankings
- `cbind`: Combines objects by columns (`cbind` on assays); other other slots are conserved from the first object provided as argument. Both, ranking and AUC are calculated by column (cell or sample). Therefore, it is fine to merge objects as long as they come from equivalent datasets (and keep same genes/genesets, etc...)

Examples

```r
# This example is run using a fake expression matrix.
# Therefore, the output will be meaningless.

############# Fake run of AUCell #############
set.seed(123)
exprMatrix <- matrix(data=sample(c(rep(0, 5000), sample(1:3, 5000, replace=TRUE))),
nrow=20,
dimnames=list(paste("Gene", 1:20, sep=""),
```
paste("Cell", 1:500, sep=""))

dim(exprMatrix)

# Running AUCell
cells_rankings <- AUCell_buildRankings(exprMatrix)

fewGenes <- sample(rownames(exprMatrix), 10)
otherGenes <- sample(rownames(exprMatrix), 5)
geneSets <- list(geneSet1=fewGenes,
                 geneSet2=otherGenes)
cells_AUC <- AUCell_calcAUC(geneSets, cells_rankings, aucMaxRank=5, nCores=1)

# Exploring the output:
cells_AUC

class(cells_AUC)

# Extracting the AUC matrix:
getAUC(cells_AUC)[,1:5]

# Subsetting and regular manipulation methods are also available:
cells_AUC[1:2,]
cells_AUC[3:4]

dim(cells_AUC)
nrow(cells_AUC)
ncol(cells_AUC)
colnames(cells_AUC)
rownames(cells_AUC)

### Merging 2 objects (ranking or AUC):
sample1 <- cells_AUC[10:20]
sample2 <- cells_AUC[100:140]
cbind(sample1, sample2)

---

**AUCell_assignCells**

### Description

Assigns whether the gene sets are considered "active" on each cell based on the given thresholds

### Usage

```r
AUCell_assignCells(cellsAUC, thresholds, nCores = 1)
```

### Arguments

- **cellsAUC**: AUC object returned by `AUCell_calcAUC`.
- **thresholds**: Thresholds selected for each geneset (named vector).
- **nCores**: Number of cores to use for computation.
**Value**

List with the following elements for each gene-set:

- ‘aucThr’ threshold value, in the same format as `AUC_cell_exploreThresholds()`
- ‘assignment’ List of cells that pass the selected AUC threshold

**See Also**

Previous step in the workflow: `AUC_cell_calcAUC` and optionally `AUC_cell_exploreThresholds`

See the package vignette for examples and more details: `vignette("AUC_cell")`

**Examples**

```r
# This example is run using a fake expression matrix.
# Therefore, the output will be meaningless.

############### Fake expression matrix ###############
set.seed(123)
exprMatrix <- matrix(data=sample(c(rep(0, 5000), sample(1:3, 5000, replace=TRUE))),
nrow=20,
dimnames=list(paste("Gene", 1:20, sep=""),
paste("Cell", 1:500, sep="")))
dim(exprMatrix)

########################################################################

######### Previous steps in the workflow #########
# Step 1.
cells_rankings <- AUC_cell_buildRankings(exprMatrix, plotStats=FALSE)

# Step 2.
# (Gene sets: random genes)
geneSets <- list(geneSet1=sample(rownames(exprMatrix), 10),
                 geneSet2=sample(rownames(exprMatrix), 5))
cells_AUC <- AUC_cell_calcAUC(geneSets, cells_rankings, aucMaxRank=5)

########################################################################

############## Step 3: Assign cells ##############

# 1. Plot histograms and obtain some pre-computed thresholds
# (this example is only meant to show the interface/arguments of the function,
# see the vignette for meaningful examples)
set.seed(123)
par(mfrow=c(1,2)) # Plot is divided into one row and two columns
thresholds <- AUC_cell_exploreThresholds(cells_AUC, plotHist=TRUE)
thresholds$geneSet1$aucThr

# 2. Obtain cells over a given threshold:
names(which(getAUC(cells_AUC)["geneSet1",] > 0.19))

# Alternative 1: assign cells according to the 'automatic' threshold
cells_assignment <- AUC_cell_exploreThresholds(cells_AUC,
```
AUCell_buildRankings

Build gene expression rankings for each cell

Description
Builds the "rankings" for each cell: expression-based ranking for all the genes in each cell.
The genes with same expression value are shuffled. Therefore, genes with expression '0' are randomly sorted at the end of the ranking.
These "rankings" can be seen as a new representation of the original dataset. Once they are calculated, they can be saved for future analyses.

Usage

AUCell_buildRankings(
  exprMat,
  featureType = "genes",
  plotStats = TRUE,
  splitByBlocks = FALSE,
  BPPARAM = NULL,
  keepZeroesAsNA = FALSE,
  verbose = TRUE,
  nCores = NULL,
  mctype = NULL,
  ...
)

## S4 method for signature 'dgCMatrix'
AUCell_buildRankings(
  exprMat,
  featureType = "genes",
  plotStats = TRUE,
  splitByBlocks = TRUE,
BPPARAM = NULL,
keepZeroesAsNA = FALSE,
verbose = TRUE,
nCores = NULL,
mctype = NULL
)

## S4 method for signature 'matrix'
AUCell_buildRankings(
  exprMat,
  featureType = "genes",
  plotStats = TRUE,
  splitByBlocks = FALSE,
  BPPARAM = NULL,
  keepZeroesAsNA = FALSE,
  verbose = TRUE,
  nCores = NULL,
  mctype = NULL
)

## S4 method for signature 'ExpressionSet'
AUCell_buildRankings(
  exprMat,
  featureType = "genes",
  plotStats = TRUE,
  splitByBlocks = FALSE,
  BPPARAM = NULL,
  keepZeroesAsNA = FALSE,
  verbose = TRUE,
  nCores = NULL,
  mctype = NULL
)

## S4 method for signature 'SummarizedExperiment'
AUCell_buildRankings(
  exprMat,
  featureType = "genes",
  plotStats = TRUE,
  splitByBlocks = FALSE,
  BPPARAM = NULL,
  keepZeroesAsNA = FALSE,
  verbose = TRUE,
  assayName = NULL,
  nCores = NULL,
  mctype = NULL
)
Arguments

exprMat  
Expression matrix (genes as rows, cells as columns) The expression matrix can also be provided as one of the R/Bioconductor classes:

- **dgCMatrix-class**: Sparse matrix
- **RangedSummarizedExperiment and derived classes (e.g. SingleCellExperiment)**: The matrix will be obtained through assay(exprMatrix), - which will extract the first assay (usually the counts) - or the assay name given in 'assayName'
- **ExpressionSet**: The matrix will be obtained through exprs(exprMatrix)

featureType  
Name for the rows (e.g. "genes"). Only for naming the rankings, not used internally.

plotStats  
Should the function plot the expression boxplots/histograms? (TRUE / FALSE). These plots can also be produced with the function `plotGeneCount`.

splitByBlocks  
Whether to split the matrix by blocks in the ranking calculation. Allows using multiple cores. FALSE by default. If using sparse matrices it is automatically set to TRUE.

BPPARAM  
Set to use multiple cores. Only used if 'splitByBlocks=TRUE'

keepZeroesAsNA  
Convert zeroes to NA instead of locating randomly at the end of the ranking.

verbose  
Should the function show progress messages? (TRUE / FALSE)

nCores  
Deprecated

mctype  
Deprecated

...  
Other arguments

assayName  
Name of the assay containing the expression matrix (e.g. in SingleCellExperiment objects)

Details

It is important to check that most cells have at least the number of expressed/detected genes that are going to be used to calculate the AUC ('aucMaxRank' in `calcAUC()`). The histogram provided by `AUCell_buildRankings()` allows to quickly check this distribution. `plotGeneCount(exprMatrix)` allows to obtain only the plot before building the rankings.

Value

Ranking of the feature within the cell (features as rows, cells as columns)

See Also

Next step in the workflow: AUCell_calcAUC.

See the package vignette for examples and more details: vignette("AUCell")
Examples

# This example is run using a fake expression matrix.
# Therefore, the output will be meaningless.

############# Fake expression matrix #############
set.seed(123)
exprMatrix <- matrix(data=sample(c(rep(0, 5000), sample(1:3, 5000, replace=TRUE))),
nrow=20,
dimnames=list(paste("Gene", 1:20, sep=""),
             paste("Cell", 1:500, sep="")))

############################################################
cells_rankings <- AUCell_buildRankings(exprMatrix, plotStats=TRUE)
cells_rankings

AUCalcCell_calcaUC  Calculate AUC

Description

Calculates the 'AUC' for each gene-set in each cell.

Usage

AUCalcCell_calcaUC(
geneSets,
rankings,
nCores = 1,
normAUC = TRUE,
aucMaxRank = ceiling(0.05 * nrow(rankings)),
verbose = TRUE
)

## S4 method for signature 'list'
AUCalcCell_calcaUC(
geneSets,
rankings,
nCores = 1,
normAUC = TRUE,
aucMaxRank = ceiling(0.05 * nrow(rankings)),
verbose = TRUE
)

## S4 method for signature 'character'
AUCalcCell_calcaUC(
geneSets,
rankings,
nCores = 1,
### Arguments

- **geneSets**: List of gene-sets (or signatures) to test in the cells. The gene-sets should be provided as `GeneSet`, `GeneSetCollection` or character list (see examples).
- **rankings**: 'Rankings' created for this dataset with `AUCell_buildRankings`.
- **nCores**: Number of cores to use for computation.
- **normAUC**: Wether to normalize the maximum possible AUC to 1 (Default: TRUE).
- **aucMaxRank**: Threshold to calculate the AUC (see 'details' section)
- **verbose**: Should the function show progress messages? (TRUE / FALSE)

### Details

In a simplified way, the AUC value represents the fraction of genes, within the top X genes in the ranking, that are included in the signature. The parameter 'aucMaxRank' allows to modify the number of genes (maximum ranking) that is used to perform this computation. By default, it is set to 5% of the total number of genes in the rankings. Common values may range from 1 to 20%.

### Value

Matrix with the AUC values (gene-sets as rows, cells as columns).
See Also

Previous step in the workflow: `AUCell_buildRankings`. Next step in the workflow: `AUCell_exploreThresholds`. See the package vignette for examples and more details: `vignette("AUCell")`

Examples

# This example is run using a fake expression matrix.
# Therefore, the output will be meaningless.

###### Fake expression matrix #######
set.seed(123)
exprMatrix <- matrix(data=sample(c(rep(0, 5000), sample(1:3, 5000, replace=TRUE))),
nrow=20,
dimnames=list(paste("Gene", 1:20, sep=""),
            paste("Cell", 1:500, sep="")))

# Previous step in the workflow #
# Step 1.
cells_rankings <- AUCell_buildRankings(exprMatrix)

# Step 2: Calculate AUC

# In this example we use two gene sets: 10 and 5 random genes
# (see other formatting examples at the end)
fewGenes <- sample(rownames(exprMatrix), 10)
otherGenes <- sample(rownames(exprMatrix), 5)

geneSets <- list(geneSet1=fewGenes,
                 geneSet2=otherGenes)

geneSets

# Calculate AUC with the rankings from Step 1.
# To be able to run this fake example (which contain only 20 genes),
# we use aucMaxRank=5 (top 25% of the genes in the ranking)
cells_AUC <- AUCell_calcAUC(geneSets, cells_rankings, aucMaxRank=5, nCores=1)

cells_AUC

to subset & access the AUC slot (as matrix):
cells_AUC[1:2,]
cells_AUC[,3:4]
getAUC(cells_AUC)[,1:5]

# These methods are also available:
dim(cells_AUC)
nrow(cells_AUC)
ncol(cells_AUC)
colnames(cells_AUC)[1:4]
rownames(cells_AUC)

#########################################################
# Alternatives for the input of gene sets:
#
# a) Character vector (i.e. only one gene-set)
# It will take the default name 'geneSet'
fewGenes
test <- AUCell_calcAUC(fewGenes, cells_rankings, aucMaxRank=5)

# b) List
geneSets <- list(geneSet1=fewGenes,
                 geneSet2=otherGenes)
geneSets
test <- AUCell_calcAUC(geneSets, cells_rankings, aucMaxRank=5)

# c) GeneSet object (from GSEABase)
library(GSEABase)
geneSetOne <- GeneSet(fewGenes, setName="geneSetOne")
geneSetOne
test <- AUCell_calcAUC(geneSetOne, cells_rankings, aucMaxRank=5)

# d) GeneSetCollection object (from GSEABase)
geneSetTwo <- GeneSet(otherGenes, setName="geneSetTwo")
geneSets <- GeneSetCollection(geneSetOne, geneSetTwo)
geneSets
test <- AUCell_calcAUC(geneSets, cells_rankings, aucMaxRank=5)

AUCCell_exploreThresholds

Description
Plots all the AUC histograms (per gene-set) and calculates several likely thresholds for each gene-set

Usage
AUCCell_exploreThresholds(
cellsAUC,
thrP = 0.01,
nCores = 1,
smallestPopPercent = 0.25,
plotHist = TRUE,
densAdjust = 2,
assignCells = FALSE,
nBreaks = 100,
AUCell_exploreThresholds

verbose = TRUE
}

getThresholdSelected(aucellThresholds)

getAssignments(aucellThresholds)

Arguments

cellsAUC AUC object returned by AUCell_calcAUC.

thrP Probability to determine outliers in some of the distributions (see 'details' section).
By default it is set to 1% (thrP): if there are 3000 cells in the dataset, it is expected that approximately 30 cells are over this threshold if the AUC is normally distributed.
nCores Number of cores to use for computation.

smallestPopPercent Size (percentage) of the smallest population of cells expected. Used to calculate some of the thresholds.

plotHist Whether to plot the AUC histograms. (TRUE / FALSE)
densAdjust Parameter for the density curve. (See density for details).

assignCells Return the list of cells that pass the automatically selected threshold? (TRUE/FALSE)
nBreaks Number of bars to plot in the histograms.

verbose Should the function show progress messages? (TRUE / FALSE)

AUCellThresholds For aux functions: Output from AUCell_exploreThresholds

Details

To ease the selection of an assignment threshold, this function adjusts the AUCs of each gene-set to several distributions and calculates possible thresholds:

• minimumDens (plot in Blue): Inflection point of the density curve. This is usually a good option for the ideal situation with bimodal distributions.
  To avoid false positives, by default this threshold will not be chosen if the second distribution is higher (i.e. the majority of cells have the gene-set "active").

• L_k2 (plot in Red): Left distribution, after adjusting the AUC to a mixture of two distributions. The threshold is set to the right (prob: 1-(thrP/nCells)). Only available if 'mixtools' package is installed.

• R_k3 (plot in Pink): Right distribution, after adjusting the AUC to a mixture of three distributions. The threshold is set to the left (prob: thrP). Only available if 'mixtools' package is installed.

• Global_k1 (plot in Grey): "global" distribution (i.e. mean and standard deviations of all cells). The threshold is set to the right (prob: 1-(thrP/nCells)).
  The threshold based on the global distribution is ignored from the automatic selection unless the mixed models are overlapping.
Note: If assignCells=TRUE, the highest threshold is used to select cells. However, keep in mind that this function is only meant to ease the selection of the threshold, and we highly recommend to look at the AUC histograms and adjust the threshold manually if needed. We recommend to be specially aware on gene-sets with few genes (10-15) and thresholds that are set extremely low.

Value

List with the following elements for each gene-set:

- ‘aucThr‘ Thresholds calculated with each method (see 'details' section), and the number of cells that would be assigned using that threshold.
  If assignCells=TRUE, the threshold selected automatically is the highest value (in most cases, excluding the global distribution).
- ‘assignment‘ List of cells that pass the selected AUC threshold (if assignCells=TRUE)

If plotHist=TRUE the AUC histogram is also plot, including the distributions calculated and the corresponding thresholds in the same color (dashed vertical lines). The threshold that is automatically selected is shown as a thicker non-dashed vertical line.

See Also

Previous step in the workflow: AUCell_calcAUC.
See the package vignette for examples and more details: vignette("AUCell")

Examples

# This example is run using a fake expression matrix.
# Therefore, the output will be meaningless.

########### Fake expression matrix ###############
set.seed(123)
exprMatrix <- matrix(data=sample(c(rep(0, 5000), sample(1:3, 5000, replace=TRUE))),
nrow=20,
dimnames=list(paste("Gene", 1:20, sep=""),
paste("Cell", 1:500, sep="")))

dim(exprMatrix)

############################################

########### Previous steps in the workflow #########
# Step 1.
cells_rankings <- AUCell_buildRankings(exprMatrix, plotStats=FALSE)

# Step 2.
# (Gene sets: random genes)
geneSets <- list(geneSet1=sample(rownames(exprMatrix), 10),
geneSet2=sample(rownames(exprMatrix), 5))
cells_AUC <- AUCell_calcAUC(geneSets, cells_rankings, aucMaxRank=5)

############################################

########### Step 3: Assign cells ###############
# 1. Plot histograms and obtain some pre-computed thresholds
# (this example is only meant to show the interface/arguments of the function,
# see the vignette for meaningful examples)
set.seed(123)
par(mfrow=c(1,2)) # Plot is divided into one row and two columns
thresholds <- AUCell_exploreThresholds(cells_AUC, plotHist=TRUE)
thresholds$geneSet1$aucThr

# 2. Obtain cells over a given threshold:
names(which(getAUC(cells_AUC)["geneSet1",] > 0.19))

# Alternative 1: assign cells according to the 'automatic' threshold
cells_assignment <- AUCell_exploreThresholds(cells_AUC,
   plotHist=FALSE, assignCells=TRUE)

# Cells assigned:
getAssignments(cells_assignment)
# Threshold applied:
getThresholdSelected(cells_assignment)

# Alternative 2: choose a threshold manually and assign cells
newThresholds = getThresholdSelected(cells_assignment)
newThresholds["geneSet1"] = 0.8
newAssignments = AUCell_assignCells(cells_AUC, newThresholds)
getAssignments(newAssignments)

AUCell_plotHist  Plot AUC histogram

Description

Plots the distribution of AUC across the cells (for each gene-set) as an histogram.

Usage

AUCell_plotHist(
   cellsAUC,
   aucThr = max(cellsAUC),
   nBreaks = 100,
   onColor = "dodgerblue4",
   offColor = "slategray2",
   ...
)

Arguments

cellsAUC Subset of the object returned by AUCell_calcAUC (i.e. including only the gene-
sets to plot)
AUC value planned to use as threshold (to make sure the X axis includes it), if any. Otherwise, the X axis extends to cover only the AUC values plotted.

Number of 'bars' to plot (breaks argument for hist function).

Color for the bars that pass the AUC threshold

Color for the bars that do not pass the AUC threshold

Other arguments to pass to hist function.

Value

List of histogram objects (invisible).

See Also

See the package vignette for examples and more details: vignette("AUCell")

Examples

# This example is run using a fake expression matrix.
# Therefore, the output will be meaningless.

# Generated fake expression matrix
set.seed(123)
exprMatrix <- matrix(data=sample(c(rep(0, 5000), sample(1:3, 5000, replace=TRUE))),
nrow=20,
dimnames=list(paste("Gene", 1:20, sep=""),
paste("Cell", 1:500, sep="")))
dim(exprMatrix)

# This is the beginning of the workflow

# Step 1.
cells_rankings <- AUCcell_buildRankings(exprMatrix, plotStats=FALSE)

# Step 2.
# (Gene set: 10 random genes)
genes <- sample(rownames(exprMatrix), 10)
geneSets <- list(geneSet1=genes)
# (aucMaxRank=5 to run with this fake example, it will return 'high' AUC values)
cells_AUC <- AUCcell_calAUC(geneSets, cells_rankings, aucMaxRank=5)

# Plot histogram:
AUCcell_plotHist(cells_AUC["geneSet1"], nBreaks=10)
AUC_plotTSNE

Description

Plots the AUC histogram and t-SNE coloured by AUC, binary activity and TF expression

Usage

AUC_plotTSNE(
  tSNE,
  exprMat = NULL,
  cellsAUC = NULL,
  thresholds = NULL,
  reorderGeneSets = FALSE,
  cex = 1,
  alphaOn = 1,
  alphaOff = 0.2,
  borderColor = adjustcolor("lightgray", alpha.f = 0.1),
  offColor = "lightgray",
  plots = c("histogram", "binaryAUC", "AUC", "expression"),
  exprCols = c("goldenrod1", "darkorange", "brown"),
  asPNG = FALSE,
  ...
)

Arguments

tSNE t-SNE coordinates (e.g. tSNE$Y)
exprMat Expression matrix
cellsAUC AUC (as returned by calcAUC)
thresholds Thresholds returned by AUCCell
reorderGeneSets Whether to reorder the gene sets based on AUC similarity
cex Scaling factor for the dots in the scatterplot
alphaOn Transparency for the dots representing "active" cells
alphaOff Transparency for the dots representing "inactive" cells
borderColor Border color for the dots (scatterplot)
offColor Color for the dots representing "inactive" cells
plots Which plots to generate? Select one or multiple: plots=c("histogram", "binaryAUC", "AUC", "expression")
exprCols Color scale for the expression
asPNG Output each individual plot in a .png file? (can also be a directory)
... Other arguments to pass to hist function.
Details

To avoid calculating thresholds, set thresholds to FALSE

Value

Returns invisible: cells_trhAssignment

See Also

List of vignettes included in the package: vignette(package="AUCell")

Examples

```
####
# Fake run of AUCell
set.seed(123)
exprMatrix <- matrix(
  data=sample(c(rep(0, 5000), sample(1:3, 5000, replace=TRUE))),
  nrow=20,
  dimnames=list(paste("Gene", 1:20, sep=""),
                paste("Cell", 1:500, sep=""))
geneSets <- list(geneSet1=sample(rownames(exprMatrix), 10),
                 geneSet2=sample(rownames(exprMatrix), 5))

cells_rankings <- AUCell_buildRankings(exprMatrix, plotStats = FALSE)
cells_AUC <- AUCell_calcAUC(geneSets, cells_rankings, aucMaxRank=5, nCores=1)
selectedThresholds <- rowMeans(getAUC(cells_AUC))
cellsTsne<- Rtsne::Rtsne(t(exprMatrix),max_iter = 10)$Y
# cellsTsne<- tsne::tsne(t(exprMatrix),max_iter = 10)
rownames(cellsTsne) <- colnames(exprMatrix)
####

par(mfrow=c(2,3))
ths <- AUCell_plotTSNE(tSNE=cellsTsne, exprMat=NULL,
                        cellsAUC=cells_AUC, thresholds=selectedThresholds,
                        plots = c("histogram", "binaryAUC", "AUC"))

####
# Color based on the known phenodata:
cellInfo <- data.frame(cellType1=sample(LETTERS[1:3],ncol(exprMatrix), replace=TRUE),
                        cellType2=sample(letters[5:7],ncol(exprMatrix), replace=TRUE),
                        nGenes=abs(rnorm(ncol(exprMatrix))),
                        row.names=colnames(exprMatrix))
colVars <- list(cellType2=setNames(c("skyblue","magenta", "darkorange"),letters[5:7]))
# dev.off()
plotTsne_cellProps(cellsTsne, cellInfo, colVars=colVars)
```
Description

Runs AUCell (calculates the ranking + score genesets)

Usage

AUCell_run(
  exprMat,
  geneSets,
  featureType = "genes",
  keepZeroesAsNA = FALSE,
  normAUC = TRUE,
  aucMaxRank = ceiling(0.05 * nrow(exprMat)),
  BPPARAM = NULL,
  ...
)

## S4 method for signature 'dgCMatrix'
AUCell_run(
  exprMat,
  geneSets,
  featureType = "genes",
  keepZeroesAsNA = FALSE,
  normAUC = TRUE,
  aucMaxRank = ceiling(0.05 * nrow(exprMat)),
  BPPARAM = NULL
)

## S4 method for signature 'matrix'
AUCell_run(
  exprMat,
  geneSets,
  featureType = "genes",
  keepZeroesAsNA = FALSE,
  normAUC = TRUE,
  aucMaxRank = ceiling(0.05 * nrow(exprMat)),
  BPPARAM = NULL
)

## S4 method for signature 'SummarizedExperiment'
AUCell_run(
  exprMat,
  geneSets,
  featureType = "genes",
  keepZeroesAsNA = FALSE,
  normAUC = TRUE,
  aucMaxRank = ceiling(0.05 * nrow(exprMat)),
  BPPARAM = NULL
)
keepZeroesAsNA = FALSE,
normAUC = TRUE,
aucMaxRank = ceiling(0.05 * nrow(exprMat)),
BPPARAM = NULL,
assayName = NULL
)

Arguments

exprMat  Expression matrix (genes/regions as rows, cells as columns) The expression matrix can also be provided as one of the R/Bioconductor classes:
  - dgCMatrix-class: Sparse matrix
  - RangedSummarizedExperiment and derived classes (e.g. SingleCellExperiment ): The matrix will be obtained through assay(exprMatrix), -which will extract the first assay (usually the counts)- or the assay name given in 'assayName'
geneSets List of gene-sets (or signatures) to test in the cells. The gene-sets should be provided as GeneSet, GeneSetCollection or character list (see examples).
featureType Name for the rows (e.g. "genes"). Only for naming the rankings, not used internally.
keepZeroesAsNA Convert zeroes to NA instead of locating randomly at the end of the ranking.
normAUC Whether to normalize the maximum possible AUC to 1 (Default: TRUE).
aucMaxRank Threshold to calculate the AUC (see 'details' section)
BPPARAM Set to use multiple cores. Only used if 'splitByBlocks=TRUE'
... Other arguments
assayName Name of the assay containing the expression matrix (e.g. in SingleCellExperiment objects)

Details

In a simplified way, the AUC value represents the fraction of genes, within the top X genes in the ranking, that are included in the signature. The parameter 'aucMaxRank' allows to modify the number of genes (maximum ranking) that is used to perform this computation. By default, it is set to 5% of the total number of genes in the rankings. Common values may range from 1 to 20%.

Value

Matrix with the AUC values (gene-sets as rows, cells as columns).

See Also

Includes AUCell_buildRankings and AUCell_calcAUC. Next step in the workflow: AUCell_exploreThresholds. See the package vignette for examples and more details: vignette("AUCell")
Examples

# This example is run using a fake expression matrix.
# Therefore, the output will be meaningless.

########## Fake expression matrix ##########
set.seed(123)
exprMatrix <- matrix(data=sample(c(rep(0, 5000), sample(1:3, 5000, replace=TRUE))),
nrow=20,
dimnames=list(paste("Gene", 1:20, sep=""),
paste("Cell", 1:500, sep="")))
exprMatrix <- as(exprMatrix, "dgCMatrix")

# In this example we use two gene sets: 10 and 5 random genes
# (see other formatting examples at the end)
fewGenes <- sample(rownames(exprMatrix), 10)
otherGenes <- sample(rownames(exprMatrix), 5)
geneSets <- list(geneSet1=fewGenes,
                 geneSet2=otherGenes)
geneSets

# Calculate AUCell score for the genes in the sets
# To be able to run this fake example (which contain only 20 genes),
# we use aucMaxRank=5 (top 25% of the genes in the ranking)
cells_AUC <- AUCell_run(exprMatrix, geneSets, aucMaxRank=5)

## To run in paralell:
# cells_AUC <- AUCell_run(exprMatrix, geneSets, aucMaxRank=5,
# BPPARAM=BiocParallel::MulticoreParam(5))

# Format of the output:
cells_AUC

# To subset & access the AUC slot (as matrix):
cells_AUC[1:2,]
cells_AUC[,3:4]
getAUC(cells_AUC)[,1:5]

# These methods are also available:
dim(cells_AUC)
nrow(cells_AUC)
ncol(cells_AUC)
colnames(cells_AUC)[1:4]
rownames(cells_AUC)

###########################################################################
# Alternatives for the input of gene sets:
# a) Character vector (i.e. only one gene-set)
# It will take the default name 'geneSet'
fewGenes
test <- AUCell_run(exprMatrix, fewGenes, aucMaxRank=5)

# b) List
geneSets <- list(geneSet1=fewGenes,
                 geneSet2=otherGenes)
geneSets
test <- AUCell_run(exprMatrix, fewGenes, aucMaxRank=5)

# c) GeneSet object (from GSEABase)
library(GSEABase)
geneSetOne <- GeneSet(fewGenes, setName="geneSetOne")
geneSetOne
test <- AUCell_run(exprMatrix, fewGenes, aucMaxRank=5)

# d) GeneSetCollection object (from GSEABase)
geneSetTwo <- GeneSet(otherGenes, setName="geneSetTwo")
geneSets <- GeneSetCollection(geneSetOne, geneSetTwo)
geneSets
test <- AUCell_run(exprMatrix, fewGenes, aucMaxRank=5)

---

**getSetNames**

*Get gene set name (excluding number of genes/regions after space)*

**Description**

Returns the gene set name (i.e. selects the given pattern)

**Usage**

```
getSetNames(aucMat, patterns, startChr = "^", endChr = " ")
```

**Arguments**

- **aucMat**: AUC matrix
- **patterns**: patterns
- **startChr**: Character to indicate the start (typically "^")
- **endChr**: Character at the end of the gene set name, i.e. space or "." after a transcription factor name

**Value**

Returns the gene set name (i.e. selects the given pattern)
nGenes

**GeneSet methods**

**Description**

Functions to manipulate GeneSet and GeneSetCollection objects (from package GSEABase)

**Usage**

nGenes(geneSet)

## S4 method for signature 'GeneSet'
nGenes(geneSet)

## S4 method for signature 'GeneSetCollection'
nGenes(geneSet)

subsetGeneSets(geneSets, geneNames)

## S4 method for signature 'GeneSetCollection'
subsetGeneSets(geneSets, geneNames)

setGeneSetNames(geneSets, newNames)

## S4 method for signature 'GeneSetCollection'
setGeneSetNames(geneSets, newNames)

**Arguments**

- geneSet: One gene-set (**GeneSet**)
- geneSets: Gene-set collection (**GeneSetCollection**)
- geneNames: Gene names (for subset)
- newNames: New names (to assign to the gene sets)

**Value**

- **nGenes()**: provides the number of genes in the gene-set, or each of the gene-sets in a collection
- **subsetGeneSets()**: Subsets each of the gene-sets in a collection to contain only the genes in the given list. Equivalent to intersect(), but keeping the original gene-set name.
- **setGeneSetNames()**: Modifies the name of each gene-set in a collection
Examples

library(GSEABase)
genes_1 <- GeneSet(paste("Gene", 1:20, sep=""), setName="geneSet1")
genes_2 <- GeneSet(paste("Gene", 18:22, sep=""), setName="geneSet2")
geneSets <- GeneSetCollection(genes_1, genes_2)

nGenes(genes_1)
nGenes(geneSets)

subsetGeneSets(geneSets, paste("Gene", 15:20, sep=""))

geneSets_newNames <- setGeneSetNames(geneSets, c("one", "two"))
names(geneSets_newNames)

Description

Reorder the gene-sets based on AUC similarity

Usage

orderAUC(auc)

Arguments

auc AUC (as returned by calcAUC)

Value

gene-set names in the suggested order

Examples

# cellsAUC <- cellsAUC[orderAUC(cellsAUC),]
Description

Colors the embeddings (t-SNE/Umap) based on the activity of 3 (groups of) geneSets

Usage

```r
plotEmb_rgb(
  aucMat,
  embedding,
  geneSetsByCol,
  aucType = "AUC",
  aucMaxContrast = 0.8,
  offColor = "#c0c0c030",
  showPlot = TRUE,
  showLegend = TRUE,
  ...
)
```

Arguments

- `aucMat`: AUC matrix (continuous or binary)
- `embedding`: AUC matrix (continuous or binary)
- `geneSetsByCol`: Gene sets to plot
- `aucType`: "AUC" or "Binary"
- `aucMaxContrast`: To increase the AUC contrast decrease the value.
- `offColor`: Color por the cells completelly off. To deactivate (color as black), set to NULL.
- `showPlot`: Whether to plot the coloured embeddings.
- `showLegend`: Whether to plot add a legend to the plot.
- `...`: Other arguments to pass to the `plot` function.

Value

The cell colors (invisible)
**plotGeneCount**

**Description**

Plots a histogram and boxplot for the number of genes detected in each cell.

**Usage**

```r
plotGeneCount(exprMat, plotStats = TRUE, verbose = TRUE)
```

**Arguments**

- `exprMat`: Expression matrix (genes as rows, cells as columns)
- `plotStats`: Logical. If true, it plots the histogram, otherwise only calculates the percentages of genes detected.
- `verbose`: Should the function show progress messages? (TRUE / FALSE)

**Details**

It is important to check that most cells have at least the number of expressed/detected genes that are going to be used to calculate the AUC (‘aucMaxRank’ in ‘calcAUC’). The histogram provided by ‘AUCell_buildRankings()’ allows to quickly check this distribution. ‘plotGeneCount(exprMatrix)’ allows to obtain only the plot before building the rankings.

**Value**

Quantiles with the number of genes detected by cell (invisible). This result is also printed if verbose=TRUE.

**See Also**

See the package vignette for more details: vignette("AUCell")

**Examples**

```r
### (Fake expression matrix)
exprMatrix <- matrix(sample(c(rep(0, 500), sample(1:3, 500, replace=TRUE))),
nrow=20)
rownames(exprMatrix) <- paste("Gene", 1:20, sep="")
colnames(exprMatrix) <- paste("Sample", 1:50, sep="")
###
plotGeneCount(exprMatrix)
title(sub="Fake expression matrix")
```
Description

Plots the t-SNE coloured based on the known cell properties

Usage

plotTsne_cellProps(
  tSNE,
  cellInfo,
  colVars = NULL,
  cex = 1,
  sub = "",
  gradientCols = c("yellow", "orange", "red"),
  showLegend = TRUE
)

Arguments

tSNE           t-SNE coordinates (e.g. tSNE$Y)
cellInfo      Dataframe with cell phenodata
colVars       Colors for the cell properties (optional)
cex            Scaling factor for the dots in the scatterplot
sub            Subtitle (e.g. tSNE type)
gradientsCols  Gradient colors for numerical variables
showLegend     Whether to show the legend

Value

Plots the t-SNE

Examples

#######
# Fake run of AUCell
set.seed(123)
exprMatrix <- matrix(
  data=sample(c(rep(0, 5000), sample(1:3, 5000, replace=TRUE))),
  nrow=20,
  dimnames=list(paste("Gene", 1:20, sep=""),
                paste("Cell", 1:500, sep="")))
geneSets <- list(geneSet1=sample(rownames(exprMatrix), 10),
                 geneSet2=sample(rownames(exprMatrix), 5))
cells_rankings <- AUCell_buildRankings(exprMatrix, plotStats = FALSE)
cells_AUC <- AUCell_calcAUC(geneSets, cells_rankings, aucMaxRank=5, nCores=1)
selectedThresholds <- rowMeans(getAUC(cells_AUC))
cellsTsne <- Rtsne::Rtsne(t(exprMatrix), max_iter = 10)$Y
# cellsTsne <- tsne::tsne(t(exprMatrix), max_iter = 10)
rownames(cellsTsne) <- colnames(exprMatrix)

par(mfrow=c(2,3))
thrs <- AUCell_plotTSNE(tSNE=cellsTsne, exprMat=NULL,
cellsAUC=cells_AUC, thresholds=selectedThresholds,
plots = c("histogram", "binaryAUC", "AUC"))

# Color based on the known phenodata:
cellInfo <- data.frame(cellType1=sample(LETTERS[1:3], ncol(exprMatrix), replace=TRUE),
cellType2=sample(letters[5:7], ncol(exprMatrix), replace=TRUE),
nGenes=abs(rnorm(ncol(exprMatrix))),
rownames=colnames(exprMatrix))
colVars <- list(cellType2=setNames(c("skyblue", "magenta", "darkorange"), letters[5:7]))
# dev.off()
plotTsne_cellProps(cellsTsne, cellInfo, colVars=colVars)

updateAucellResults  

**updateAucellResults**  

**Update AUCell results**

**Description**

Updates the AUC scores provided by AUCell from a previous version.

**Usage**

updateAucellResults(oldAucObject, objectType = "AUC")

**Arguments**

- **oldAucObject**: Object to update  
- **objectType**: Either "AUC" or "ranking" indicating the object type

**Value**

Updated version of the object as `aucellResults`. 
**Examples**

```r
oldAuc <- matrix(data=1:2000, nrow=50, ncol=40)
updateAucellResults(oldAuc)
```
Index

AUCell_assignCells, 4
AUCell_buildRankings, 6, 10, 11, 20
AUCell_buildRankings,dgCMatrix-method (AUCell_buildRankings), 6
AUCell_buildRankings,ExpressionSet-method (AUCell_buildRankings), 6
AUCell_buildRankings,matrix-method (AUCell_buildRankings), 6
AUCell_buildRankings,SummarizedExperiment-method (AUCell_buildRankings), 6
AUCell_calcAUC, 4, 5, 8, 9, 13–15, 20
AUCell_calcAUC,character-method (AUCell_calcAUC), 9
AUCell_calcAUC,GeneSet-method (AUCell_calcAUC), 9
AUCell_calcAUC,GeneSetCollection-method (AUCell_calcAUC), 9
AUCell_calcAUC,list-method (AUCell_calcAUC), 9
AUCell_exploreThresholds, 5, 11, 12, 20
AUCell_plotHist, 15
AUCell_plotTSNE, 17
AUCell_run, 19
AUCell_run,dgCMatrix-method (AUCell_run), 19
AUCell_run,matrix-method (AUCell_run), 19
AUCell_run,SummarizedExperiment-method (AUCell_run), 19
aucellResults, 28
aucellResults-class, 2
aucellResults-method (aucellResults-class), 2
cbind (aucellResults-class), 2
cbind,aucellResults-method (aucellResults-class), 2
dgCMatrix-class, 8, 20
GeneSet, 10, 20, 23
GeneSetCollection, 10, 20, 23
getAssignments (AUCell_exploreThresholds), 12
getAUC (aucellResults-class), 2
generate, aucellResults-method (aucellResults-class), 2
getRanking (aucellResults-class), 2
getRanking,aucellResults-method (aucellResults-class), 2
getSetNames, 22
getThresholdSelected (AUCell_exploreThresholds), 12
hist, 16, 17
nGenes, 23
nGenes,GeneSet-method (nGenes), 23
subsetGeneSets (nGenes), 23
orderAUC, 24
plotEmb_rgb, 25
plotGeneCount, 8, 26
plotTsne_cellProps, 27
RangedSummarizedExperiment, 8, 20
rbind (aucellResults-class), 2
rbind,aucellResults-method (aucellResults-class), 2
setGeneSetNames (nGenes), 23
setGeneSetNames,GeneSetCollection-method (nGenes), 23
show (aucellResults-class), 2
show,aucellResults-method (aucellResults-class), 2
SingleCellExperiment, 8, 20
subsetGeneSets (nGenes), 23
subsetGeneSets,GeneSetCollection-method (nGenes), 23
updateAucellResults, 28