Package ‘AUCell’

April 11, 2018

Type Package

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

Version 1.0.0

Date 2017-09-26

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

Maintainer Sara Aibar <sara.aibar@kuleuven.vib.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 R.utils, utils, graphics, stats, data.table, mixtools, GSEABase, SummarizedExperiment, methods

Enhances doMC, doRNG, doParallel, foreach

Suggests Biobase, GEOquery, devtools, zoo, DT, NMF, plotly, BiocStyle, knitr, rmarkdown, testthat

License GPL-3

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

LazyData TRUE

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

R topics documented:

`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

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

getAUC(object)

getRanking(object)
```

Arguments

`object`	Results from `AUCell_buildRanking` or `AUCell_calcAUC`.

Value

- show: Prints a summary of the object
- getAUC: Returns the matrix containing the AUC
- getRanking: Returns the matrix containing the rankings

Examples

```
# 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)
rownames(exprMatrix) <- paste("Gene", 1:20, sep="")
colnames(exprMatrix) <- 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)
```

`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, plotStats = TRUE, nCores = 1,
  verbose = TRUE)

## S4 method for signature 'matrix'
AUCell_buildRankings(exprMat, plotStats = TRUE,
  nCores = 1, verbose = TRUE)

## S4 method for signature 'SummarizedExperiment'
AUCell_buildRankings(exprMat,
  plotStats = TRUE, nCores = 1, verbose = TRUE)

## S4 method for signature 'ExpressionSet'
AUCell_buildRankings(exprMat, plotStats = TRUE,
  nCores = 1, verbose = TRUE)
```

Arguments

`exprMat`	Expression matrix (genes as rows, cells as columns) The expression matrix can also be provided as one of the Bioconductor classes: • ExpressionSet: The matrix will be obtained through exprs(exprMatrix) • RangedSummarizedExperiment: The matrix will be obtained through assay(exprMatrix), wich will extract the first assay (usually the counts)
`plotStats`	Should the function plot the expression boxplots/histograms? (TRUE / FALSE). These plots can also be produced with the function `plotGeneCount`.
`nCores`	Number of cores to use for computation.
`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

data.table of genes (row) by cells (columns) with the ranking of the gene within the cell.

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)
rownames(exprMatrix) <- paste("Gene", 1:20, sep="")
colnames(exprMatrix) <- paste("Cell", 1:500, sep="")
dim(exprMatrix)
##############################################

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

`AUCell_calcAUC` *Calculate AUC*

Description

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

Usage

```
AUCell_calcAUC(geneSets, rankings, nCores = 1, aucMaxRank = ceiling(0.05 *
  nrow(rankings)), verbose = TRUE)

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

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

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

## S4 method for signature 'GeneSetCollection'
AUCell_calcAUC(geneSets, rankings, nCores = 1,
  aucMaxRank = ceiling(0.05 * nrow(rankings)), verbose = TRUE)
```

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.
`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)
rownames(exprMatrix) <- paste("Gene", 1:20, sep="")
colnames(exprMatrix) <- paste("Cell", 1:500, sep="")
dim(exprMatrix)
##################################################

######### 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)
set.seed(123)
cells_AUC <- AUCell_calcAUC(geneSets, cells_rankings, aucMaxRank=5, nCores=1)

# 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)
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)
```

`AUCell_exploreThresholds`

AUCell_exploreThresholds

Description

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

Usage

```
AUCell_exploreThresholds(cellsAUC, thrP = 0.01, nCores = 1,
  smallestPopPercent = 0.25, plotHist = TRUE, densAdjust = 2,
  assignCells = FALSE, nBreaks = 100, verbose = TRUE)
```

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)

Details

To ease the selection of an assignment theshold, 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)
rownames(exprMatrix) <- paste("Gene", 1:20, sep="")
colnames(exprMatrix) <- 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: assign cells according to the 'automatic' threshold
cells_assignment <- AUCell_exploreThresholds(cells_AUC,
                                             plotHist=FALSE, assignCells=TRUE)
# Cells assigned:
lapply(cells_assignment, function(x) x$assignment)
# Threshold applied:
sapply(cells_assignment, function(x) x$aucThr$selected)
```

AUCell_plot — *Plot AUC histogram*

Description

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

Usage

```
AUCell_plot(cellsAUC, aucThr = max(cellsAUC), nBreaks = 100, ...)
```

Arguments

`cellsAUC`	Subset of the object returned by `AUCell_calcAUC` (i.e. including only the gene-sets to plot)
`aucThr`	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.
`nBreaks`	Number of 'bars' to plot (breaks argument for hist function).
`...`	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.

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

############# Begining of the workflow ###########
# Step 1.
cells_rankings <- AUCell_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 <- AUCell_calcAUC(geneSets, cells_rankings, aucMaxRank=5)
```

```
####################################################

# Plot histogram:
AUCell_plot(cells_AUC["geneSet1",], nBreaks=10)
```

---

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 inthe 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)
```

`plotGeneCount` *plotGeneCount*

Description

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

Usage

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

Arguments

`exprMat`	Expression matrix (genes as rows, cells as columns)
`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). his result is also printed if verbose=TRUE.

See Also

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

Examples

```
### (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")
```

---

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

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

Index