Package 'RcisTarget'

October 14, 2021

Type Package

Title RcisTarget Identify transcription factor binding motifs enriched on a list of genes or genomic regions

Version 1.12.0

Date 2021-04-15

Author Sara Aibar, Gert Hulselmans, 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 RcisTarget identifies transcription factor binding motifs (TFBS) over-represented on a gene list.
In a first step, RcisTarget selects DNA motifs that are significantly over-represented in the surroundings of the transcription start site (TSS) of the genes in the gene-set.
This is achieved by using a database that contains genome-wide cross-species rankings for each motif. The motifs that are then annotated to TFs and those that have a high Normalized Enrichment Score (NES) are retained.
Finally, for each motif and gene-set, RcisTarget predicts the candidate target genes (i.e. genes in the gene-set that are ranked above the leading edge).

URL http://scenic.aertslab.org

Depends R (>= 3.4)

Imports AUCell (>= 1.1.6), BiocGenerics, data.table, feather, graphics, GenomeInfoDb, GenomicRanges, arrow (>= 2.0.0), dplyr, tibble, GSEABase, methods, R.utils, stats, SummarizedExperiment, utils

Enhances doMC, doRNG, zoo

Suggests Biobase, BiocStyle, BiocParallel, doParallel, DT, foreach, gplots, rtracklayer, igraph, knitr, RcisTarget.hg19.motifDBs.cisbpOnly.500bp, rmarkdown, testthat, visNetwork

License GPL-3

BugReports https://github.com/aertslab/RcisTarget/issues

biocViews GeneRegulation, MotifAnnotation, Transcriptomics, Transcription, GeneSetEnrichment, GeneTarget

LazyData FALSE

VignetteBuilder knitr

RoxygenNote 7.1.1

git_url https://git.bioconductor.org/packages/RcisTarget

git_branch RELEASE_3_13

git_last_commit da8bb0d

git_last_commit_date 2021-05-19

Date/Publication 2021-10-14

R topics documented:

`addLogo` *Add motif logo to RcisTarget results table*

Description

Adds a column containing the logo URL to RcisTarget results table. Note that Transfac-Pro logos cannot be shown.

Usage

```
addLogo(motifEnrDT, addHTML = TRUE, dbVersion = "v9", motifCol = "motif")
```

Arguments

motifEnrDT	Results from RcisTarget (data.table)
addHTML	Whether to add the HTML tag <img> around the URL or not (boolean).
dbVersion	For current databases (mc9nr) use "v9"
motifCol	Name of the column which contains the logo ID.

Value

Returns the results table with a new column: 'logo'. This column contains either a URL with the logo image, or the HTML code to show the logo [e.g. with datatable()].

See Also

See the package vignette for more examples: `vignette("RcisTarget")`

Examples

```
# Run the enrichment (or load previous results)
load(paste(file.path(system.file('examples', package='RcisTarget')),
           "motifEnrichmentTable_wGenes.RData", sep="/"))

# Add link to logo
newMotifErnTable <- addLogo(motifEnrichmentTable_wGenes)

# Show table
library(DT)
datatable(newMotifErnTable[,-c("enrichedGenes"), with=FALSE],
          escape = FALSE,
          filter="top",
          options=list(pageLength=5))
```

---

addMotifAnnotation *Add motif annotation*

Description

Select significant motifs and/or annotate motifs to genes or transcription factors. The motifs are considered significantly enriched if they pass the the Normalized Enrichment Score (NES) threshold.

Usage

```
addMotifAnnotation(
  auc,
  nesThreshold = 3,
  digits = 3,
```

```
  motifAnnot = NULL,
  motifAnnot_highConfCat = c("directAnnotation", "inferredBy_Orthology"),
  motifAnnot_lowConfCat = c("inferredBy_MotifSimilarity",
    "inferredBy_MotifSimilarity_n_Orthology"),
  idColumn = "motif",
  highlightTFs = NULL,
  keepAnnotationCategory = TRUE
)
```

Arguments

auc	Output from calcAUC.
nesThreshold	Numeric. NES threshold to calculate the motif significant (3.0 by default). The NES is calculated -for each motif- based on the AUC distribution of all the motifs for the gene-set [(x-mean)/sd].
digits	Integer. Number of digits for the AUC and NES in the output table.
motifAnnot	Motif annotation database containing the annotations of the motif to transcription factors. The names should match the ranking column names.
motifAnnot_highConfCat	Categories considered as source for 'high confidence' annotations. By default, "directAnnotation" (annotated in the source database), and "inferredBy_Orthology" (the motif is annotated to an homologous/ortologous gene).
motifAnnot_lowConfCat	Categories considered 'lower confidence' source for annotations. By default, the annotations inferred based on motif similarity ("inferredBy_MotifSimilarity", "inferredBy_MotifSimilarity_n_Orthology").
idColumn	Annotation column containing the ID (e.g. motif, accession)
highlightTFs	Character. If a list of transcription factors is provided, the column TFinDB in the otuput table will indicate whether any of those TFs are included within the 'high-confidence' annotation (two asterisks, **) or 'low-confidence' annotation (one asterisk, *) of the motif. The vector can be named to indicate which TF to highlight for each gene-set. Otherwise, all TFs will be used for all geneSets.
keepAnnotationCategory	Include annotation type in the TF information?

Value

data.table with the folowing columns:

- geneSet: Name of the gene set
- motif: ID of the motif (colnames of the ranking, it might be other kind of feature)
- NES: Normalized enrichment score of the motif in the gene-set
- AUC: Area Under the Curve (used to calculate the NES)
- TFinDB: Indicates whether the highlightedTFs are included within the high-confidence annotation (two asterisks, **) or lower-confidence annotation (one asterisk, *)

- TF_highConf: Transcription factors annotated to the motif based on high-confidence annotations.
- TF_lowConf: Transcription factors annotated to the motif according to based on lower-confidence annotations.

See Also

Next step in the workflow: addSignificantGenes.

Previous step in the workflow: calcAUC.

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

Examples

```
#################################################
# Setup & previous steps in the workflow:

#### Gene sets
# As example, the package includes an Hypoxia gene set:
txtFile <- paste(file.path(system.file('examples', package='RcisTarget')),
                 "hypoxiaGeneSet.txt", sep="/")
geneLists <- list(hypoxia=read.table(txtFile, stringsAsFactors=FALSE)[,1])

#### Databases
## Motif rankings: Select according to organism and distance around TSS
## (See the vignette for URLs to download)
# motifRankings <- importRankings("hg19-500bp-upstream-7species.mc9nr.feather")

## For this example we will use a SUBSET of the ranking/motif databases:
library(RcisTarget.hg19.motifDBs.cisbpOnly.500bp)
data(hg19_500bpUpstream_motifRanking_cispbOnly)
motifRankings <- hg19_500bpUpstream_motifRanking_cispbOnly

## Motif - TF annotation:
data(motifAnnotations_hgnc) # human TFs (for motif collection 9)
motifAnnotation <- motifAnnotations_hgnc

### Run RcisTarget
# Step 1. Calculate AUC
motifs_AUC <- calcAUC(geneLists, motifRankings)

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

### (This step: Step 2)
# Before starting: Setup the paralell computation
library(BiocParallel); register(MulticoreParam(workers = 2))
# Select significant motifs, add TF annotation & format as table
motifEnrichmentTable <- addMotifAnnotation(motifs_AUC,
                                           motifAnnot=motifAnnotation)

# Alternative: Modifying some options
motifEnrichment_wIndirect <- addMotifAnnotation(motifs_AUC, nesThreshold=2,
```

```
        motifAnnot=motifAnnotation,
        highlightTFs = "HIF1A",
        motifAnnot_highConfCat=c("directAnnotation"),
        motifAnnot_lowConfCat=c("inferredBy_MotifSimilarity",
                                "inferredBy_MotifSimilarity_n_Orthology",
                                "inferredBy_Orthology"),
        digits=3)

# Getting TFs for a given TF:
motifs <- motifEnrichmentTable$motif[1:3]

getMotifAnnotation(motifs, motifAnnot=motifAnnotation)
getMotifAnnotation(motifs, motifAnnot=motifAnnotation, returnFormat="list")

### Exploring the output:
# Number of enriched motifs (Over the given NES threshold)
nrow(motifEnrichmentTable)

# Interactive exploration
motifEnrichmentTable <- addLogo(motifEnrichmentTable)
DT::datatable(motifEnrichmentTable, filter="top", escape=FALSE,
              options=list(pageLength=50))
# Note: If using the fake database, the results of this analysis are meaningless

# The object returned is a data.table (for faster computation),
# which has a diferent syntax from the standard data.frame or matrix
# Feel free to convert it to a data.frame (as.data.frame())
motifEnrichmentTable[,1:6]


################################################
# Next step (step 3, optional):
## Not run:
  motifEnrichmentTable_wGenes <- addSignificantGenes(motifEnrichmentTable,
                                                     geneSets=geneLists,
                                                     rankings=motifRankings,
                                                     method="aprox")

## End(Not run)
```

---

addSignificantGenes *Add significant genes*

Description

Identify which genes (of the gene-set) are highly ranked for each motif.

- addSignificantGenes(): adds them to the results table.
- getSignificantGenes(): Calculates the significant genes for ONE gene set. It provides the plot and the gene list (it is used by addSignificantGenes).

Usage

```
addSignificantGenes(
  resultsTable,
  geneSets,
  rankings,
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = "geneList",
  method = "aprox",
  nMean = 50,
  nCores = 1
)

## S4 method for signature 'list'
addSignificantGenes(
  resultsTable,
  geneSets,
  rankings,
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = "geneList",
  method = "aprox",
  nMean = 50,
  nCores = 1
)

## S4 method for signature 'character'
addSignificantGenes(
  resultsTable,
  geneSets,
  rankings,
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = "geneList",
  method = "aprox",
  nMean = 50,
  nCores = 1
)

## S4 method for signature 'GeneSet'
addSignificantGenes(
  resultsTable,
  geneSets,
  rankings,
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = "geneList",
  method = "aprox",
```

```
  nMean = 50,
  nCores = 1
)

## S4 method for signature 'GeneSetCollection'
addSignificantGenes(
  resultsTable,
  geneSets,
  rankings,
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = "geneList",
  method = "aprox",
  nMean = 50,
  nCores = 1
)

getSignificantGenes(
  geneSet,
  rankings,
  signifRankingNames = NULL,
  method = "iCisTarget",
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = c("geneList", "incidMatrix"),
  nCores = 1,
  digits = 3,
  nMean = 50
)

## S4 method for signature 'list'
getSignificantGenes(
  geneSet,
  rankings,
  signifRankingNames = NULL,
  method = "iCisTarget",
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = c("geneList", "incidMatrix"),
  nCores = 1,
  digits = 3,
  nMean = 50
)

## S4 method for signature 'character'
getSignificantGenes(
  geneSet,
  rankings,
```

```
  signifRankingNames = NULL,
  method = "iCisTarget",
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = c("geneList", "incidMatrix"),
  nCores = 1,
  digits = 3,
  nMean = 50
)

## S4 method for signature 'factor'
getSignificantGenes(
  geneSet,
  rankings,
  signifRankingNames = NULL,
  method = "iCisTarget",
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = c("geneList", "incidMatrix"),
  nCores = 1,
  digits = 3,
  nMean = 50
)

## S4 method for signature 'GeneSet'
getSignificantGenes(
  geneSet,
  rankings,
  signifRankingNames = NULL,
  method = "iCisTarget",
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = c("geneList", "incidMatrix"),
  nCores = 1,
  digits = 3,
  nMean = 50
)

## S4 method for signature 'GeneSetCollection'
getSignificantGenes(
  geneSet,
  rankings,
  signifRankingNames = NULL,
  method = "iCisTarget",
  maxRank = 5000,
  plotCurve = FALSE,
  genesFormat = c("geneList", "incidMatrix"),
  nCores = 1,
```

```
  digits = 3,
  nMean = 50
)
```

Arguments

resultsTable	[addSignificantGenes] Output table from addMotifAnnotation
geneSets	[addSignificantGenes] List of gene-sets which was analyzed.
rankings	Motif rankings used to analyze the gene list (They should be the same as used for calcAUC in this same analysis).
maxRank	Maximum rank to take into account for the recovery curve (Default: 5000).
plotCurve	Logical. Wether to plot the recovery curve (Default: FALSE).
genesFormat	"geneList" or "incidMatrix". Format to return the genes (Default: "geneList").
method	"iCisTarget" or "aprox". There are two methods to identify the highly ranked genes: (1) equivalent to the ones used in iRegulon and i-cisTarget (method="iCisTarget", recommended if running time is not an issue), and (2) a faster implementation based on an approximate distribution using the average at each rank (method="aprox", useful to scan multiple gene sets). (Default: "aprox")
nMean	Only used for "aprox" method: Interval to calculate the running mean and sd. Default: 50 (aprox. nGenesInRanking/400).
nCores	Number of cores to use for parallelization (Default: 1).
geneSet	[getSignificantGenes] Gene-set to analyze (Only one).
signifRankingNames	[getSignificantGenes] Motif ranking name.
digits	[getSignificantGenes] Number of digits to include in the output.

Details

The highly ranked genes are selected based on the distribution of the recovery curves of the gene set across all the motifs in the database. In the plot, the red line indicates the average of the recovery curves of all the motifs, the green line the average + standard deviation, and the blue line the recovery curve of the current motif. The point of maximum distance between the current motif and the green curve (mean+sd), is the rank selected as maximum enrichment. All the genes with lower rank will be considered enriched.

Depending on whether the method is "iCisTarget" or "aprox", the mean and SD at each rank are calculated slightly different. "iCisTarget" method calculates the recovery curves for all the motifs, and then calculates the average and SD at each rank. Due to the implementation of the function in R, this method is slower than just subsetting the ranks of the genes in for each motif, and calculating the average of the available ones at each position with a sliding window. Since there are over 18k motifs, the chances of getting several measures at each rank are very high and highly resemble the results calculated by iCisTarget, though they are often not exactly the same (hence the name: "aprox" method).

Value

Output from addMotifAnnotation adding the folowing columns:

- nEnrGenes: Number of genes highly ranked
- rankAtMax: Ranking at the maximum enrichment, used to determine the number of enriched genes.
- enrichedGenes: Genes that are highly ranked for the given motif. If genesFormat="geneList", the gene names are collapsed into a comma separated text field (alphabetical order). If genesFormat="incidMatrix", they are formatted as an indicence matrix, i.e. indicanting with 1 the genes present, and 0 absent.

If plotCurve=TRUE, the recovery curve is plotted.

See Also

Previous step in the workflow: addMotifAnnotation.

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

Examples

```
##################################################
# Setup & previous steps in the workflow:

#### Gene sets
# As example, the package includes an Hypoxia gene set:
txtFile <- paste(file.path(system.file('examples', package='RcisTarget')),
                 "hypoxiaGeneSet.txt", sep="/")
geneLists <- list(hypoxia=read.table(txtFile, stringsAsFactors=FALSE)[,1])

#### Databases
## Motif rankings: Select according to organism and distance around TSS
## (See the vignette for URLs to download)
# motifRankings <- importRankings("hg19-500bp-upstream-7species.mc9nr.feather")

## For this example we will use a SUBSET of the ranking/motif databases:
library(RcisTarget.hg19.motifDBs.cisbpOnly.500bp)
data(hg19_500bpUpstream_motifRanking_cispbOnly)
motifRankings <- hg19_500bpUpstream_motifRanking_cispbOnly

## Motif - TF annotation:
data(motifAnnotations_hgnc) # human TFs (for motif collection 9)
motifAnnotation <- motifAnnotations_hgnc

### Run RcisTarget
# Step 1. Calculate AUC
motifs_AUC <- calcAUC(geneLists, motifRankings)
# Step 2. Select significant motifs, add TF annotation & format as table
motifEnrichmentTable <- addMotifAnnotation(motifs_AUC,
           motifAnnot=motifAnnotation)

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

```
#################################################
# (This step: Step 3)
# Identify the genes that have the motif significantly enriched
# (i.e. genes from the gene set in the top of the ranking)
par(mfrow=c(1,2))
motifEnrichmentTable_wGenes <- addSignificantGenes(motifEnrichmentTable,
                                       genesFormat="geneList",
                                       plotCurve=TRUE,
                                       geneSets=geneLists,
                                       rankings=motifRankings,
                                       method="aprox")

#### Exploring the output:
# The object returned is a data.table
# Feel free to convert it to a data.frame:
motifEnrichmentTable_wGenes <- as.data.frame(motifEnrichmentTable_wGenes)

# Enriched genes
enrGenes <- motifEnrichmentTable_wGenes[1,"enrichedGenes"]
enrGenes
strsplit(enrGenes, ";")

# As incidence matrix
motifEnr_wIncidMat <- addSignificantGenes(motifEnrichmentTable,
                geneSets=geneLists, rankings=motifRankings,
                method="aprox",
                genesFormat = "incidMatrix")

motifEnr_wIncidMat <- as.data.frame(motifEnr_wIncidMat)
which(colnames(motifEnr_wIncidMat) == "rankAtMax")

incidMat <- motifEnr_wIncidMat[,8:ncol(motifEnr_wIncidMat)]
rownames(incidMat) <- motifEnr_wIncidMat[,"motif"]
incidMat <- incidMat[, colSums(incidMat)>0, drop=FALSE]

# Plot as network
par(mfrow=c(1,1))
library(igraph)
plot(graph.incidence(incidMat))

#############################################################
# Alternative method: getSignificantGenes()
selectedMotif <- rownames(incidMat)
onlyGenes <- getSignificantGenes(geneSet=geneLists$hypoxia,
                            signifRankingNames=selectedMotif,
                            genesFormat="incidMatrix",
                            plotCurve=TRUE,
                            rankings=motifRankings,
                            method="aprox")
```

aucScores-class *Class to store the AUC scores for RcisTarget.*

Description

Contains the AUC scores for each gene- or region-set. They can be accessed through `getAUC()` and the regular methods (i.e. nrow, rownames...) available for SummarizedExperiment objects.

Usage

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

## S4 method for signature 'aucScores'
getAUC(object)
```

Arguments

`object`	Results from `calcAUC`.

Value

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

Examples

```
##################################################
# Setup & previous steps in the workflow:

#### Gene sets
# As example, the package includes an Hypoxia gene set:
txtFile <- paste(file.path(system.file('examples', package='RcisTarget')),
                 "hypoxiaGeneSet.txt", sep="/")
geneLists <- list(hypoxia=read.table(txtFile, stringsAsFactors=FALSE)[,1])

#### Databases
## Motif rankings: Select according to organism and distance around TSS
## (See the vignette for URLs to download)
# motifRankings <- importRankings("hg19-500bp-upstream-7species.mc9nr.feather")

## For this example we will use a SUBSET of the ranking/motif databases:
library(RcisTarget.hg19.motifDBs.cisbpOnly.500bp)
data(hg19_500bpUpstream_motifRanking_cispbOnly)
motifRankings <- hg19_500bpUpstream_motifRanking_cispbOnly

## Motif - TF annotation:
data(motifAnnotations_hgnc) # human TFs (for motif collection 9)
motifAnnotation <- motifAnnotations_hgnc
```

```
### Run RcisTarget
# Step 1. Calculate AUC
motifs_AUC <- calcAUC(geneLists, motifRankings)

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

#Exploring the output:
motifs_AUC

class(motifs_AUC)

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

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

dim(motifs_AUC)
nrow(motifs_AUC)
ncol(motifs_AUC)
colnames(motifs_AUC)
rownames(motifs_AUC)
```

---

calcAUC *Calculate AUC*

Description

Calculates the Area Under the Curve (AUC) of each gene-set for each motif ranking. This measure is used in the following steps to identify the DNA motifs that are significantly over-represented in the gene-set.

Usage

```
calcAUC(
  geneSets,
  rankings,
  nCores = 1,
  aucMaxRank = 0.03 * getNumColsInDB(rankings),
  verbose = TRUE
)

## S4 method for signature 'list'
calcAUC(
  geneSets,
  rankings,
  nCores = 1,
```

```
  aucMaxRank = 0.03 * getNumColsInDB(rankings),
  verbose = TRUE
)

## S4 method for signature 'character'
calcAUC(
  geneSets,
  rankings,
  nCores = 1,
  aucMaxRank = 0.03 * getNumColsInDB(rankings),
  verbose = TRUE
)

## S4 method for signature 'GeneSet'
calcAUC(
  geneSets,
  rankings,
  nCores = 1,
  aucMaxRank = 0.03 * getNumColsInDB(rankings),
  verbose = TRUE
)

## S4 method for signature 'GeneSetCollection'
calcAUC(
  geneSets,
  rankings,
  nCores = 1,
  aucMaxRank = 0.03 * getNumColsInDB(rankings),
  verbose = TRUE
)
```

Arguments

geneSets	List of gene-sets to analyze. The gene-sets should be provided as GeneSet, GeneSetCollection or character list (see examples).
rankings	'Motif rankings' database for the required organism and search-space (i.e. 10kbp around- or 500bp upstream the TSS). These objects are provided in separate files, which can be imported with `importRankings()`:

- http://pyscenic.aertslab.org/databases/mm9-500bp-upstream-7species.mc9nr.feather[mm9-500bp-upstream-7species.mc9nr] (Mouse, 500bp)
- http://pyscenic.aertslab.org/databases/mm9-tss-centered-10kb-7species.mc9nr.feather[mm9-tss-centered-10kb-7species.mc9nr] (Mouse, 10kbp)
- http://pyscenic.aertslab.org/databases/hg19-500bp-upstream-7species.mc9nr.feather[hg19-500bp-upstream-7species.mc9nr] (Human, 500bp)
- http://pyscenic.aertslab.org/databases/hg19-tss-centered-10kb-7species.mc9nr.feather[hg19-tss-centered-10kb-7species.mc9nr] (Human, 10kbp)
- -Coming soon- (Fly)

See `vignette("RcisTarget")` for an exhaustive list of databases.

Since the normalized enrichment score (NES) of the motif depends on the total number of motifs in the database, we highly recommend to use the full version of the databases (20k motifs). A smaller version of the human databases, containing only the 4.6k motifs from cisbp, are available in Bioconductor:

- RcisTarget.hg19.motifDBs.cisbpOnly.500bp (Human)

`nCores` Number of cores to use for computation. Note: In general, using a higher number of cores (e.g. processes) decreases overall running time. However, it also deppends on the available memory and overall system load. Setting nCores too high might also decrease performance.

`aucMaxRank` Threshold to calculate the AUC. 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 range from 1 to 10%. See `vignette("RcisTarget")` for examples and more details.

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

Value

`aucScores` of gene-sets (columns) by motifs (rows) with the value of AUC for each pair as content.

See Also

Next step in the workflow: `addMotifAnnotation`.

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

Examples

```
# RcisTarget workflow for advanced users:
# Running the workflow steps individually

## Not run:

##################################################
#### Load your gene sets
# As example, the package includes an Hypoxia gene set:
txtFile <- paste(file.path(system.file('examples', package='RcisTarget')),
                 "hypoxiaGeneSet.txt", sep="/")
geneLists <- list(hypoxia=read.table(txtFile, stringsAsFactors=FALSE)[,1])

#### Load databases
## Motif rankings: Select according to organism and distance around TSS
## (See the vignette for URLs to download)
motifRankings <- importRankings("hg19-500bp-upstream-7species.mc9nr.feather")

## Motif - TF annotation:
data(motifAnnotations_hgnc) # human TFs (for motif collection 9)
```

```
motifAnnotation <- motifAnnotations_hgnc
#################################################

#### Run RcisTarget

# Step 1. Calculate AUC
motifs_AUC <- calcAUC(geneLists, motifRankings)

# Step 2. Select significant motifs, add TF annotation & format as table
motifEnrichmentTable <- addMotifAnnotation(motifs_AUC,
                          motifAnnot=motifAnnotation)

# Step 3 (optional). Identify genes that have the motif significantly enriched
# (i.e. genes from the gene set in the top of the ranking)
motifEnrichmentTable_wGenes <- addSignificantGenes(motifEnrichmentTable,
                                                   geneSets=geneLists,
                                                   rankings=motifRankings,
                                                   method="aprox")


## End(Not run)
```

---

cisTarget *cisTarget*

Description

Identifies DNA motifs significantly over-represented in a gene-set.

This is the main function to run RcisTarget. It includes on the following steps:

- 1. Motif enrichment analysis (calcAUC)
- 2. Motif-TF annotation (addMotifAnnotation)
- 3. Selection of significant genes (addSignificantGenes)

Usage

```
cisTarget(
  geneSets,
  motifRankings,
  motifAnnot = NULL,
  motifAnnot_highConfCat = c("directAnnotation", "inferredBy_Orthology"),
  motifAnnot_lowConfCat = c("inferredBy_MotifSimilarity",
    "inferredBy_MotifSimilarity_n_Orthology"),
  highlightTFs = NULL,
  nesThreshold = 3,
  aucMaxRank = 0.05 * ncol(motifRankings),
  geneErnMethod = "aprox",
  geneErnMmaxRank = 5000,
```

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

Arguments

geneSets	List of gene-sets to analyze. The gene-sets should be provided as GeneSet, GeneSetCollection or character list (see examples).
motifRankings	Database of the appropiate organism and search-space (i.e. 10kbp around- or 500bp upstream the TSS). These objects are provided in separate files, which can be imported with `importRankings()`: • http://pyscenic.aertslab.org/databases/mm9-500bp-upstream-7species.mc9nr.feather[mm9-500bp-upstream-7species.mc9nr] (Mouse, 500bp) • http://pyscenic.aertslab.org/databases/mm9-tss-centered-10kb-7species.mc9nr.feather[mm9-tss-centered-10kb-7species.mc9nr] (Mouse, 10kbp) • http://pyscenic.aertslab.org/databases/hg19-500bp-upstream-7species.mc9nr.feather[hg19-500bp-upstream-7species.mc9nr] (Human, 500bp) • http://pyscenic.aertslab.org/databases/hg19-tss-centered-10kb-7species.mc9nr.feather[hg19-tss-centered-10kb-7species.mc9nr] (Human, 10kbp) • -Coming soon- (Fly) See `vignette("RcisTarget")` for an exhaustive list of databases.
motifAnnot	Motif annotation database containing the annotations of the motif to transcription factors.
motifAnnot_highConfCat	Categories considered as source for 'high confidence' annotations. By default, "directAnnotation" (annotated in the source database), and "inferredBy_Orthology" (the motif is annotated to an homologous/ortologous gene).
motifAnnot_lowConfCat	Categories considered 'lower confidence' source for annotations. By default, the annotations inferred based on motif similarity ("inferredBy_MotifSimilarity", "inferredBy_MotifSimilarity_n_Orthology").
highlightTFs	Character. If a list of transcription factors is provided, the column TFinDB in the otuput table will indicate whether any of those TFs are included within the 'high-confidence' annotation (two asterisks, **) or 'low-confidence' annotation (one asterisk, *) of the motif. The vector can be named to indicate which TF to highlight for each gene-set. Otherwise, all TFs will be used for all geneSets.
nesThreshold	Numeric. NES threshold to calculate the motif significant (3.0 by default). The NES is calculated -for each motif- based on the AUC distribution of all the motifs for the gene-set [(x-mean)/sd]. The motifs are considered significantly enriched if they pass the the Normalized Enrichment Score (NES) threshold.
aucMaxRank	Threshold to calculate the AUC. 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 'aucThresholdPERC' allows to modify the percentage of genes (of the top of the 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 range from 1 to 10%.

geneErnMethod	"iCisTarget" or "aprox". Method to identify the highly ranked genes (see addSignificantGenes for details).
geneErnMmaxRank	Maximum rank to take into account for the gene enrichment recovery curve (see addSignificantGenes for details).
nCores	Number of cores to use for computation. Note: In general, using a higher number of cores (e.g. processes) decreases overall running time. However, it also deppends on the available memory and overall system load. Setting nCores too high might also decrease performance.
verbose	Should the function show progress messages? (TRUE / FALSE)

Value

data.table containing the over-represented motifs (according to the selected NES threshold), their statistics, annotation to transcription factors and the genes with high enrichment of the motif.

See Also

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

Examples

```
# Example for running RcisTarget using cisTarget() function (workflow wrapper)

## Not run:

#################################################
### Load your gene sets
# As example, the package includes an Hypoxia gene set:
txtFile <- paste(file.path(system.file('examples', package='RcisTarget')),
                 "hypoxiaGeneSet.txt", sep="/")
geneLists <- list(hypoxia=read.table(txtFile, stringsAsFactors=FALSE)[,1])

### Load databases
# Motif rankings: Select according to organism and distance around TSS
# (See the vignette for URLs to download)
motifRankings <- importRankings("hg19-500bp-upstream-7species.mc9nr.feather")

# Motif - TF annotation:
data(motifAnnotations_hgnc) # human TFs (for motif collection 9)
motifAnnotation <- motifAnnotations_hgnc
#################################################

# Run (R)cisTarget
motifEnrichmentTable_wGenes <- cisTarget(geneLists, motifRankings,
  motifAnnot_direct=hg19_direct_motifAnnotation,
  nesThreshold=3.5, geneErnMethod="aprox", nCores=2)


## End(Not run)
```

```
# Load results from analysis
load(paste(file.path(system.file('examples', package='RcisTarget')),
           "motifEnrichmentTable_wGenes.RData", sep="/"))


### Exploring the output:
# Note: If using the fake-database, the results are not meaningful

# Number of enriched motifs (Over the given NES threshold)
nrow(motifEnrichmentTable_wGenes)

# Available info (columns)
colnames(motifEnrichmentTable_wGenes)

# The object returned is a data.table (for faster computation),
# which has a diferent syntax from the standard data.frame or matrix
# Feel free to convert it to a data.frame (as.data.frame())
class(motifEnrichmentTable_wGenes)
motifEnrichmentTable_wGenes[,1:5]

# Enriched genes
enrGenes <- as.character(motifEnrichmentTable_wGenes[1,"enrichedGenes"])
strsplit(enrGenes, ";")


# Interactive exploration
motifEnrichmentTable_wGenes <- addLogo(motifEnrichmentTable_wGenes)
DT::datatable(motifEnrichmentTable_wGenes[,1:9], escape = FALSE, filter="top",
              options=list(pageLength=5))
# Note: If using the fake database, the results of this analysis are meaningless
```

---

convertToTargetRegions *convertToTargetRegions*

Description

Convert a set of input regions to the overlapping regions in the target set.

Usage

```
convertToTargetRegions(
  queryRegions,
  targetRegions,
  minOverlap = 0.4,
  overlapType = "any",
  returnCorrespondence = FALSE,
  verbose = TRUE
)
```

Arguments

queryRegions	List of regions to convert (normally the query region set).
targetRegions	List of regions in the target set (normally the database).
minOverlap	Minimum overlap to consider (in either direction, default: 0.40).
overlapType	Parameter for findOverlaps (default: "any")
returnCorrespondence	Returns a table containing the matches, or only a list of overlapping regions (default: FALSE).
verbose	Print the number of regions selected?

Value

IDs of the regions in the "target regions" overlapping with the query regions.

See Also

getDbRegionsLoc.

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

Examples

```
## Not run:
 ## To apply on a list of regionSets:
 regionSets_db <- lapply(regionSets, function(x)
    convertToTargetRegions(queryRegions=x, targetRegions=dbRegionsLoc))

## End(Not run)
```

---

dbRegionsLoc *Genomic location for the database regions*

Description

- **dbRegionsLoc_hg19**: Contains the location for **HUMAN** regions in the database **hg19-regions-9species.all_regions.mc9nr** (hg19,refseq_r45, motif collection v9).
  Source: *hg19__refseq_r45__ClusteredUniformDHS_all_merge_cleaned2_features_rm-insul_rm-exons2_extend.regionid-location.bed*
- **dbRegionsLoc_mm9**: Contains the location for **MOUSE** regions in the database **mm9-regions-9species.all_regions.mc9nr** (mm9, refseq_r70, motif collection v9).
  Source: *mm9__refseq_r70__regulatory_regions.regionid-location.bed*
- For **drosophila**, use the function getDbRegionsLoc

Details

Documentation for the data

See Also

https://resources.aertslab.org/cistarget

`getDbRegionsLoc` *Gets the region location for the given database IDs.*

Description

Gets the region location based on the region ID. Only needed for drosophila**drosophila** (fruit fly) regions.

For **human/mouse** the region locations are stored in a separate object (i.e.). This function is not needed.

Usage

```
getDbRegionsLoc(featherFilePath, spltChr = "__", indexCol = NULL)
```

Arguments

`featherFilePath`	Path to the rankings database
`spltChr`	Character(s) used to split the prefix from the region location. The default is used for current Drosophila versions. Use NULL to skip.

Value

The region locations in a GRanges object, with the original region ID as name.

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

Examples

```
## Not run:
featherFilePath <- "~/databases/dm6-regions-11species.mc9nr.feather"
dbRegionsLoc <- getDbRegionsLoc(featherFilePath)

## End(Not run)
```

`getMotifAnnotation` *Get motif annotation*

Description

Get the genes/transcription factors annotated to the given motifs

Usage

```
getMotifAnnotation(
  motifs,
  motifAnnot,
  annotCats = c("directAnnotation", "inferredBy_MotifSimilarity",
    "inferredBy_Orthology", "inferredBy_MotifSimilarity_n_Orthology"),
  idColumn = "motif",
  returnFormat = c("asCharacter", "subset", "list")[1],
  keepAnnotationCategory = TRUE
)
```

Arguments

`motifs`	Motif IDs
`motifAnnot`	Motif annotation database containing the annotations of the motif to genes or transcription factors.
`annotCats`	Annotation categories to be considered: "directAnnotation" (annotated in the source database), "inferredBy_Orthology" (the motif is annotated to an homologous/ortologous gene), or inferred based on motif similarity ("inferredBy_MotifSimilarity", "inferredBy_MotifSimilarity_n_Orthology").
`idColumn`	Annotation column containing the ID (e.g. motif, accession)
`returnFormat`	Determines the output format. Choose one of the following values:
`keepAnnotationCategory`	Include annotation type in the TF information?

- `asCharacter`: Named vector with the genes or TFs annotated to the given motifs (in the same order, including empty and duplicated values).
- `subset`: Subset of the annotation table (list split by motif)
- `list`: List of TF names (unique values), duplicated motifs or motifs without annotation are not returned.

Value

See argument `returnFormat`

See Also

`addMotifAnnotation` add the annotation directly to the motif enrichment results.

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

Examples

```
#################################################
# Setup & previous steps in the workflow:

#### Gene sets
# As example, the package includes an Hypoxia gene set:
txtFile <- paste(file.path(system.file('examples', package='RcisTarget')),
                 "hypoxiaGeneSet.txt", sep="/")
geneLists <- list(hypoxia=read.table(txtFile, stringsAsFactors=FALSE)[,1])

#### Databases
## Motif rankings: Select according to organism and distance around TSS
## (See the vignette for URLs to download)
# motifRankings <- importRankings("hg19-500bp-upstream-7species.mc9nr.feather")

## For this example we will use a SUBSET of the ranking/motif databases:
library(RcisTarget.hg19.motifDBs.cisbpOnly.500bp)
data(hg19_500bpUpstream_motifRanking_cispbOnly)
motifRankings <- hg19_500bpUpstream_motifRanking_cispbOnly

## Motif - TF annotation:
data(motifAnnotations_hgnc) # human TFs (for motif collection 9)
motifAnnotation <- motifAnnotations_hgnc

### Run RcisTarget
# Step 1. Calculate AUC
motifs_AUC <- calcAUC(geneLists, motifRankings)

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

### (This step: Step 2)
# Before starting: Setup the paralell computation
library(BiocParallel); register(MulticoreParam(workers = 2))
# Select significant motifs, add TF annotation & format as table
motifEnrichmentTable <- addMotifAnnotation(motifs_AUC,
                                           motifAnnot=motifAnnotation)

# Alternative: Modifying some options
motifEnrichment_wIndirect <- addMotifAnnotation(motifs_AUC, nesThreshold=2,
        motifAnnot=motifAnnotation,
        highlightTFs = "HIF1A",
        motifAnnot_highConfCat=c("directAnnotation"),
        motifAnnot_lowConfCat=c("inferredBy_MotifSimilarity",
                                "inferredBy_MotifSimilarity_n_Orthology",
                                "inferredBy_Orthology"),
        digits=3)

# Getting TFs for a given TF:
motifs <- motifEnrichmentTable$motif[1:3]

getMotifAnnotation(motifs, motifAnnot=motifAnnotation)
getMotifAnnotation(motifs, motifAnnot=motifAnnotation, returnFormat="list")
```

```
### Exploring the output:
# Number of enriched motifs (Over the given NES threshold)
nrow(motifEnrichmentTable)

# Interactive exploration
motifEnrichmentTable <- addLogo(motifEnrichmentTable)
DT::datatable(motifEnrichmentTable, filter="top", escape=FALSE,
              options=list(pageLength=50))
# Note: If using the fake database, the results of this analysis are meaningless

# The object returned is a data.table (for faster computation),
# which has a diferent syntax from the standard data.frame or matrix
# Feel free to convert it to a data.frame (as.data.frame())
motifEnrichmentTable[,1:6]


################################################
# Next step (step 3, optional):
## Not run:
  motifEnrichmentTable_wGenes <- addSignificantGenes(motifEnrichmentTable,
                                                     geneSets=geneLists,
                                                     rankings=motifRankings,
                                                     method="aprox")

## End(Not run)
```

`importAnnotations` *Imports the annotations of motifs to transcription factors*

Description

RcisTarget package includes the motif annotations for the rankings using motif collection version 9 ('mc9nr', 24453 motifs):

- Human: `data(motifAnnotations_hgnc)`
- Mouse: `data(motifAnnotations_mgi)`
- Fly: `data(motifAnnotations_dmel)`
- Previous versions (Annotations for motif collection version 8: motifAnnotations_hgnc_v8 (Human), motifAnnotations_mgi_v8 (Mouse), motifAnnotations_dmel_v8 (Fly)

This function (importAnnotations) allows to import annotations for other versions of the rankings, or to keep extra data columns.

e.g. Source of the annotations (motif collection 9 'mc9nr'):

- Human: https://resources.aertslab.org/cistarget/motif2tf/motifs-v9-nr.hgnc-m0.001-o0.0.tbl
- Mouse: https://resources.aertslab.org/cistarget/motif2tf/motifs-v9-nr.mgi-m0.001-o0.0.tbl

Usage

```
importAnnotations(annotFile, motifsInRanking = NULL, columnsToKeep = NULL)
```

Arguments

annotFile	File containing the motif annotations corresponding to the rankings. They should match in organism and version.
motifsInRanking	Subset of motifs to keep. e.g. `motifsInRanking=getRanking(motifRankings)$features`
columnsToKeep	Other colums from the file to keep

Value

data.table with the annotations of the motifs to transcription factors

Columns:

- **motif**: Motif ID.
- **TF**: Transcription factor (or inferred gene).
- **directAnnotation**, **inferred_Orthology**, **inferred_MotifSimil**: Boolean values indicating whether the motif is annotated to the TF in the source database ("directAnnotation"), or whether it was inferred by orthology ("inferred_Orthology") or motif similarity ("inferred_MotifSimil").
- **Description**: Description of the source of the annotation.
- **annotationSource**: Source of the annotation formatted as factor (e.g. for subsetting). Levels: directAnnotation, inferredBy_Orthology, inferredBy_MotifSimilarity, inferredBy_MotifSimilarity_n_Orthology.

See Also

See *iRegulon* paper and documentation for **details** on how the rankings and annotations were built.

Examples

```
# motifAnnotations <- importAnnotations("motifs-v9-nr.hgnc-m0.001-o0.0.tbl")

## To save (decrease from ~100MB to 1MB):
# attr(motifAnnotations, "version") <- "motifs-v9-nr.hgnc-m0.001-o0.0.tbl"
# save(motifAnnotations, file="hgnc_motifAnnotations.RData", compress='xz')

# This code would generate the equivalent to
data(motifAnnotations_hgnc)
```

`importRankings` *Import the motif databases for RcisTarget.*

Description

The rankings are typically loaded from a .feather or .parquet file with `importRankings()`.

Usage

```
importRankings(
  dbFile,
  indexCol = NULL,
  colType = "gene",
  columns = NULL,
  warnMissingColumns = TRUE,
  dbDescr = NULL
)

getRowNames(dbFile, indexCol = NULL)

getColumnNames(dbFile)
```

Arguments

`dbFile`	.feather or .parquet file containing the rankings
`indexCol`	Column name containing the feature IDs (e.g. motif names or chip-seq tracks). If NULL, it will take the first column.
`colType`	Colum type. i.e.:'gene'or 'region'
`columns`	Columns to load from the .feather or .parquet file (e.g. to read only specific genes or regions)
`warnMissingColumns`	If 'columns' is provided, warn if any ID is not available in the rankings?
`dbDescr`	Description fields (not used in the analysis, just for convenience of the user) e.g.: `dbDescr=list(colType="gene",rowType="motif",org="Human",genome="hg19",maxRank=Inf,de`

Value

rankingRcisTarget object with the following slots: #'

- rankings: data.frame containing the rankings
- colType: 'gene'or 'region'
- nColsInDB: Number of columns (e.g. genes/regions) available in the database (.feather or .parquet file). Note that not all might be loaded in the current object.
- rowType: 'motif' or the type of feature is stored (e.g. ChipSeq)
- org: human/mouse/fly

- genome: hg19, mm9, ...
- description: global description, summary, or any other information
- maxRank: Maximum ranking included in the database, higher values are converted to Inf.

Examples

```
## Loading from a .feather or .parquet file:
#motifRankings<-importRankings("hg19-500bp-upstream-7species.mc9nr.feather")
#motifRankings<-importRankings("hg19-500bp-upstream-7species.mc9nr.parquet")

## The annotations for Motif collection 9 (sufix 'mc9nr')
# are already included in RcisTarget, and can be loaded with:
data(motifAnnotations_hgnc)

## For other versions, import the appropiate annotation. e.g.:
# annotDb <- importAnnotations("motifs-v9-nr.hgnc-m0.001-o0.0.tbl")
# optional: motifsInRanking <- getRanking(motifRankings)$features
```

`motifAnnotations` *Annotations of Motifs to TFs*

Description

- **motifAnnotations_hgnc**: Annotations to **HUMAN** transcription factors for the rankings using motif collection version 9 ('mc9nr', 24453 motifs).
  Source: *motifs-v9-nr.hgnc-m0.001-o0.0.tbl*
- **motifAnnotations_mgi**: Contains the annotations to **MOUSE** transcription factors for the rankings using motif collection version 9 ('mc9nr', 24453 motifs).
  Source: *motifs-v9-nr.mgi-m0.001-o0.0.tbl*
- **motifAnnotations_dmel**: Contains the annotations to **FLY** transcription factors for the rankings using motif collection version 9 ('mc9nr', 24453 motifs).
  Source: *motifs-v9-nr.flybase-m0.001-o0.0.tbl*
- **Previous versions**: Annotations for motif collection version 8 ('mc8nr').
  - Human: motifAnnotations_hgnc_v8 (*motifs-v8-nr.hgnc-m0.001-o0.0.tbl*)
  - Mouse: motifAnnotations_mgi_v8 (*motifs-v8-nr.mgi-m0.001-o0.0.tbl*)
  - Fly: motifAnnotations_dmel_v8 (*motifs-v8-nr.flybase-m0.001-o0.0.tbl*)

These objects are meant to be provided to RcisTarget without modification, but it can also be explored by the user to obtain further information about the motifs. Columns:

- **motif**: Motif ID.
- **TF**: Transcription factor (or inferred gene).
- **directAnnotation**, **inferred_Orthology**, **inferred_MotifSimil**: Boolean values indicating whether the motif is annotated to the TF in the source database ("directAnnotation"), or whether it was inferred by orthology ("inferred_Orthology") or motif similarity ("inferred_MotifSimil").
- **Description**: Description of the source of the annotation.
- **annotationSource**: Source of the annotation formatted as factor (e.g. for subsetting). Levels: directAnnotation, inferredBy_Orthology, inferredBy_MotifSimilarity, inferredBy_MotifSimilarity_n_Orthology.

Details

Documentation for the data

See Also

importAnnotations, RcisTarget

`rankingRcisTarget-class`

Class to store the motif databases for RcisTarget.

Description

This class contains the rankings used by RcisTarget.

They are typically loaded from a .feather file with `importRankings()`.

If the associated .descr file is available, it will also load the description of the database.

Class slots:

- rankings: data.frame (tbl_df) containing the rankings
- colType: 'gene'or 'region'
- rowType: 'motif' or the type of feature is stored (e.g. ChipSeq)
- org: human/mouse/fly
- genome: hg19, mm9, ...
- nColsInDB: Number of columns (e.g. genes/regions) available in the database (.feather file). Note that not all might be loaded in the current object.
- description: global description, summary, or any other information
- maxRank: Maximum ranking included in the database, higher values are converted to Inf.

Note that the main slot is `@rankings`, which is the one used by RcisTarget (it can accessed with `getRanking()`). The 'description' slots are mostly for user convenience.

Usage

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

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

## S4 method for signature 'rankingRcisTarget'
nrow(x)

## S4 method for signature 'rankingRcisTarget'
ncol(x)
```

```
## S4 method for signature 'rankingRcisTarget'
colnames(x)

## S4 method for signature 'rankingRcisTarget'
rownames(x)

## S4 method for signature 'rankingRcisTarget'
getMaxRank(x)

## S4 method for signature 'rankingRcisTarget'
getNumColsInDB(x)
```

Arguments

object	[method: show] rankingRcisTarget object to show
x	[several methods] rankingRcisTarget object to apply the method

Value

- show: Prints a summary of the object
- getRanking: Returns the rankings
- ncol, nrow: Returns the number of columns or rows of the ranking

Examples

```
## Loading from a .feather file:
# dbFile <- "hg19-500bp-upstream-7species.mc9nr.feather"
# motifRankings <- importRankings(dbFile)
# motifRankings

## Loading a built object:
library(RcisTarget.hg19.motifDBs.cisbpOnly.500bp)
data("hg19_500bpUpstream_motifRanking_cispbOnly")
hg19_500bpUpstream_motifRanking_cispbOnly
class(hg19_500bpUpstream_motifRanking_cispbOnly)
```

---

reRank *Re-rank RcisTarget ranking*

Description

Re-ranks the genes/regions in the database for each motif. This allows to do motif enrichment over a background.

Usage

```
reRank(rankingsDb, columns = NULL)
```

Arguments

rankingsDb	Results from RcisTarget (data.table)
columns	Whether to add the HTML tag <img> around the URL or not (boolean).

Value

Returns a new ranking database with the new ranking values.

See Also

See the "background" vignette for more examples: `vignette("RcisTarget-withBackground")`

Examples

```
library(RcisTarget.hg19.motifDBs.cisbpOnly.500bp)
data(hg19_500bpUpstream_motifRanking_cispbOnly)
motifRankings <- hg19_500bpUpstream_motifRanking_cispbOnly


genes <- colnames(getRanking(motifRankings))[10:20]
reRank(motifRankings, columns=genes)
```

Index