Package ‘TFEA.ChIP’

May 30, 2024

Type Package
Title Analyze Transcription Factor Enrichment
Version 1.24.0
Author Laura Puente Santamaría, Luis del Peso
Maintainer Laura Puente Santamaría <lpsantamaria@iib.uam.es>
Description Package to analyze transcription factor enrichment in a gene set using data from ChIP-Seq experiments.
Depends R (>= 3.5)
License Artistic-2.0
Encoding UTF-8
LazyData false
RoxygenNote 7.1.2
Imports GenomicRanges, IRanges, biomaRt, GenomicFeatures, grDevices, dplyr, stats, utils, R.utils, methods, org.Hs.eg.db
Suggests knitr, rmarkdown, S4Vectors, plotly, scales, tidyr, ggplot2, DESeq2, BiocGenerics, ggrepel, rcompanion, TxDb.Hsapiens.UCSC.hg19.knownGene, RUnit
VignetteBuilder knitr
biocViews Transcription, GeneRegulation, GeneSetEnrichment, Transcriptomics, Sequencing, ChIPSeq, RNASeq, ImmunoOncology
NeedsCompilation no
git_url https://git.bioconductor.org/packages/TFEA.ChIP
git_branch RELEASE_3_19
git_last_commit 558f990
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-29
Contents

ARNT.metadata .......................................................... 2
ARNT.peaks.bed ......................................................... 3
ChIPDB ................................................................. 4
contingency_matrix .................................................... 4
DnaseHS_db ............................................................. 5
Entrez.gene.IDs .......................................................... 5
GeneID2entrez ............................................................. 6
Genes.Upreg ............................................................... 6
getCMstats ............................................................... 7
get_chip_index .......................................................... 8
get_LFC_bar .............................................................. 8
gr.list ................................................................. 9
GSEA.result .............................................................. 9
GSEA_EnrichmentScore ................................................ 10
GSEA_ESpermutations ................................................ 11
GSEA_run ............................................................. 11
highlight_TF ............................................................ 12
hypoxia ................................................................. 13
hypoxia_DESeq ......................................................... 14
log2.FC ................................................................. 14
makeChIPGeneDB ....................................................... 15
matrixDB_to_listDB .................................................... 15
MetaData ............................................................... 16
plot_CM ................................................................. 17
plot_ES ................................................................. 17
plot_RES .............................................................. 18
preprocessInputData .................................................. 19
rankTFs .............................................................. 20
Select_genes ........................................................... 21
set_user_data ........................................................... 22
txt2GR ................................................................. 22

Index ................................................................. 24

ARNT.metadata Metadata data frame

Description

Used to run examples. Data frame containing metadata information for the ChIP-Seq GSM2390643. Fields in the data frame:

- Name: Name of the file.
- Accession: Accession ID of the experiment.
- Cell: Cell line or tissue.
'Cell Type': More information about the cells.

- Treatment
- Antibody
- TF: Transcription factor tested in the ChIP-Seq experiment.

**Usage**

```r
data("ARNT.metadata")
```

**Format**

a data frame of one row and 7 variables.

---

**Description**

Used to run examples. Data frame containing peak information from the ChIP-Seq GSM2390643. Fields in the data frame:

- Name: Name of the file.
- chr: Chromosome, factor
- start: Start coordinate for each peak
- end: End coordinate for each peak
- X.10.log.pvalue.: log10(p-Value) for each peak.

**Usage**

```r
data("ARNT.peaks.bed")
```

**Format**

a data frame of 2140 rows and 4 variables.
Contingency Matrix

ChIPDB

TF-gene binding binary matrix

Description
Its rows correspond to human genes, and its columns, to every ChIP-Seq experiment in the database. The values are 1 – if the ChIP-Seq has a peak assigned to that gene – or 0 – if it hasn’t –.

Usage
data("ChIPDB")

Format
a matrix of 1060 columns and 16797 rows

Contingency Matrix

Computes 2x2 contingency matrices

Description
Function to compute contingency 2x2 matrix by the partition of the two gene ID lists according to the presence or absence of the terms in these list in a ChIP-Seq binding database.

Usage
contingency_matrix(test_list, control_list, chip_index = get_chip_index())

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>test_list</td>
<td>List of gene Entrez IDs</td>
</tr>
<tr>
<td>control_list</td>
<td>If not provided, all human genes not present in test_list will be used as control.</td>
</tr>
<tr>
<td>chip_index</td>
<td>Output of the function “get_chip_index”, a data frame containing accession IDs of ChIPs on the database and the TF each one tests. If not provided, the whole internal database will be used</td>
</tr>
</tbody>
</table>

Value
List of contingency matrices, one CM per element in chip_index (i.e. per ChIP-seq dataset).

Examples
data('Genes.Upreg', package = 'TFEA.ChIP')
CM_list_UP <- contingency_matrix(Genes.Upreg)
DnaseHS_db

**Description**

Used to run examples. Part of a DHS database storing 76 sites for the human genome in GenomicRanges format.

**Usage**

```r
data("DnaseHS_db")
```

**Format**

GenomicRanges object with 76 elements

---

Entrez.gene.IDs

**List of Entrez Gene IDs**

**Description**

Used to run examples. Array of 2754 Entrez Gene IDs extracted from an RNA-Seq experiment sorted by log(Fold Change).

**Usage**

```r
data("Entrez.gene.IDs")
```

**Format**

Array of 2754 Entrez Gene IDs.
GeneID2entrez  

*Translates gene IDs from Gene Symbol or Ensemble ID to Entrez ID.*

**Description**

Translates mouse or human gene IDs from Gene Symbol or Ensemble Gene ID to Entrez Gene ID using the IDs approved by HGNC. When translating from Gene Symbol, keep in mind that many genes have been given more than one symbol through the years. This function will return the Entrez ID corresponding to the currently approved symbols if they exist, otherwise NA is returned. In addition some genes might map to more than one Entrez ID, in this case gene is assigned to the first match and a warning is displayed.

**Usage**

```r
GeneID2entrez(gene.IDs, return.Matrix = FALSE, mode = "h2h")
```

**Arguments**

- `gene.IDs` Array of Gene Symbols or Ensemble Gene IDs.
- `return.Matrix` T/F. When TRUE, the function returns a matrix[n,2], one column with the gene symbols or Ensemble IDs, another with their respective Entrez IDs.
- `mode` Specify the organism used: 'h2h' for homo sapiens gene IDs, 'm2m' for mouse gene IDs, or 'm2h' to get the corresponding human gene IDs from a mouse input.

**Value**

Vector or matrix containing the Entrez IDs(or NA) corresponding to every element of the input.

**Examples**

```r
GeneID2entrez(c("TNMD", "DPM1", "SCYL3", "FGR", "CFH", "FUCA2", "GCLC"))
```

---

**Genes.Upreg**  

*List of Entrez Gene IDs*

**Description**

Used to run examples. Array of 342 Entrez Gene IDs extracted from upregulated genes in an RNA-Seq experiment.

**Usage**

```r
data("Genes.Upreg")
```
getCMstats

Format

Array of 2754 Entrez Gene IDs.

Description

From a list of contingency matrices, such as the output from “contingency_matrix”, this function computes a fisher’s exact test for each matrix and generates a data frame that stores accession ID of a ChIP-Seq experiment, the TF tested in that experiment, the p-value and the odds ratio resulting from the test.

Usage

getcMstats(CM_list, chip_index = get_chip_index())

Arguments

CM_list Output of “contingency_matrix”, a list of contingency matrices.
chip_index Output of the function “get_chip_index”, a data frame containing accession IDs of ChIPs on the database and the TF each one tests. If not provided, the whole internal database will be used

Value

Data frame containing accession ID of a ChIP-Seq experiment and its experimental conditions, the TF tested in that experiment, raw and adjusted p-values, odds-ratio, and euclidean distance. and FDR-adjusted p-values (-10*log10 adj.pvalue).

Examples

data('Genes.Upreg', package = 'TFEA.ChIP')
CM_list_UP <- contingency_matrix( Genes.Upreg )
stats_mat_UP <- getCMstats( CM_list_UP )
get_chip_index

*Creates df containing accessions of ChIP-Seq datasets and TF.*

**Description**

Function to create a data frame containing the ChIP-Seq dataset accession IDs and the transcription factor tested in each ChIP. This index is used in functions like “contingency_matrix” and “GSEA_run” as a filter to select specific ChIPs or transcription factors to run an analysis.

**Usage**

```
get_chip_index(encodeFilter = FALSE, TFfilter = NULL)
```

**Arguments**

- `encodeFilter` (Optional) If TRUE, only ENCODE ChIP-Seqs are included in the index.
- `TFfilter` (Optional) Transcription factors of interest.

**Value**

Data frame containing the accession ID and TF for every ChIP-Seq experiment included in the metadata files.

**Examples**

```
get_chip_index(encodeFilter = TRUE)
get_chip_index(TFfilter=c('SMAD2', 'SMAD4'))
```

get_LFC_bar

*Plots a color bar from log2(Fold Change) values.*

**Description**

Function to plot a color bar from log2(Fold Change) values from an expression experiment.

**Usage**

```
get_LFC_bar(LFC)
```

**Arguments**

- `LFC` Vector of log2(fold change) values arranged from higher to lower. Use only the values of genes that have an Entrez ID.

**Value**

Plotly heatmap plot -log2(fold change) bar-.
gr.list

*List of one ChIP-Seq dataset*

**Description**

Used to run examples. List of part of one ChIP-Seq dataset (from wgEncodeEH002402) in GenomicRanges format with 50 peaks.

**Usage**

```r
data("gr.list")
```

**Format**

List of one ChIP-Seq dataset.

---

GSEA.result

*Output of the function GSEA.run from the TFEA.ChIP package*

**Description**

Used to run examples. Output of the function GSEA.run from the TFEA.ChIP package, contains an enrichment table and two lists, one storing running enrichment scores and the other, matches/missmatches along a gene list.

**Usage**

```r
data("GSEA.result")
```

**Format**

list of three elements, an enrichment table (data frame), and two list of arrays.
GSEA_EnrichmentScore  Computes the weighted GSEA score of gene.set in gene.list.

Description

Computes the weighted GSEA score of gene.set in gene.list.

Usage

GSEA_EnrichmentScore(
  gene.list,         # The ordered gene list
  gene.set,         # A gene set, e.g. gene IDs corresponding to a ChIP-Seq experiment's peaks.
  weighted.score.type = 0,  # Type of score: weight: 0 (unweighted = Kolmogorov-Smirnov), 1 (weighted), and 2 (over-weighted)
  correl.vector = NULL  # A vector with the coorelations (such as signal to noise scores) corresponding to the genes in the gene list
)

Arguments

- **gene.list**: The ordered gene list
- **gene.set**: A gene set, e.g. gene IDs corresponding to a ChIP-Seq experiment's peaks.
- **weighted.score.type**: Type of score: weight: 0 (unweighted = Kolmogorov-Smirnov), 1 (weighted), and 2 (over-weighted)
- **correl.vector**: A vector with the coorelations (such as signal to noise scores) corresponding to the genes in the gene list

Value

- **ES**: Enrichment score (real number between -1 and +1)
- **arg.ES**: Location in gene.list where the peak running enrichment occurs (peak of the 'mountain')
- **RES**: Numerical vector containing the running enrichment score for all locations in the gene list
- **tag.indicator**: Binary vector indicating the location of the gene sets (1's) in the gene list

Examples

```R
GSEA_EnrichmentScore(gene.list=c('3091','2034','405','55818'),
  gene.set=c('2034','112399','405'))
```
GSEA_ESpermutations  

Calculate enrichment scores for a permutation test.

Description

Function to calculate enrichment scores over a randomly ordered gene list.

Usage

GSEA_ESpermutations(
  gene.list,
  gene.set,
  weighted.score.type = 0,
  correl.vector = NULL,
  perms = 1000
)

Arguments

gene.list  Vector of gene Entrez IDs.
gene.set   A gene set, e.g. gene IDs corresponding to a ChIP-Seq experiment’s peaks.
weighted.score.type  Type of score: weight: 0 (unweighted = Kolmogorov-Smirnov), 1 (weighted), and 2 (over-weighted)
correl.vector  A vector with the correlations (such as signal to noise scores) corresponding to the genes in the gene list
perms  Number of permutations

Value

Vector of Enrichment Scores for a permutation test. gene.set=c(‘2034’,’112399’,’405’), perms=10)

GSEA_run  

Function to run a GSEA analysis

Description

Function to run a GSEA to analyze the distribution of TFBS across a sorted list of genes.
highlight_TF

Highlight certain transcription factors in a plotly graph.

Function to highlight certain transcription factors using different colors in a plotly graph.

Usage

highlight_TF(table, column, specialTF, markerColors)
Arguments

- **table**: Enrichment matrix/data.frame.
- **column**: Column # that stores the TF name in the matrix/df.
- **specialTF**: Named vector containing TF names as they appear in the enrichment matrix/df and nicknames for their color group. Example: `specialTF<-c('HIF1A','EPAS1','ARNT','SIN3A')
names(specialTF)<-c('HIF','HIF','HIF','SIN3A')`
- **markerColors**: Vector specifying the shade for every color group.

Value

List of two objects: A vector to attach to the enrichment matrix/df pointing out the color group of every row. A named vector connecting each color group to the chosen color.

---

**hypoxia**

*RNA-Seq experiment*

Description

A data frame containing information of an RNA-Seq experiment on newly transcripted RNA in HUVEC cells during two conditions, 8h of normoxia and 8h of hypoxia (deposited at GEO as GSE89831). The data frame contains the following fields:

- **Gene**: Gene Symbol for each gene analyzed.
- **Log2FoldChange**: base 2 logarithm of the fold change on RNA transcription for a given gene between the two conditions.
- **pvalue**
- **padj**: p-value adjusted via FDR.

Usage

```r
data("hypoxia")
```

Format

A data frame of 17527 observations of 4 variables.
**Description**

A DESeqResults object containing information of an RNA-Seq experiment on newly transcripted RNA in HUVEC cells during two conditions, 8h of normoxia and 8h of hypoxia (deposited at GEO as GSE89831).

**Usage**

hypoxia_DESeq

**Format**

a DESeqResults object

---

**Description**

Used to run examples. Array of 2754 log2(Fold Change) values extracted from an RNA-Seq experiment.

**Usage**

data("log2.FC")

**Format**

Array of 2754 log2(Fold Change) values.
**makeChIPGeneDB**

*Make a ChIP - target database*

**Description**

MakeChIPGeneDB generates a ChIP-seq - target database through the association of ChIP-Seq peak coordinates (provided as a GenomicRange object) to overlapping genes or gene-associated genomic regions (Ref.db).

**Usage**

```r
makeChIPGeneDB(Ref.db, gr.list, distanceMargin = 10, min.Targets = 10)
```

**Arguments**

- **Ref.db**: GenomicRanges object containing a database of reference elements (either Genes or gene-associated regions) including a gene_id metacolumn.
- **gr.list**: List of GR objects containing ChIP-seq peak coordinates (output of txt2GR).
- **distanceMargin**: Maximum distance allowed between a gene or regulatory element to assign a gene to a ChIP-seq peak. Set to 10 bases by default.
- **min.Targets**: Minimum number of putative targets per ChIP-seq in gr.list. ChIPs with fewer targets will be discarded. Regulatory element to assign a gene to a ChIP-seq peak. Set to 10 bases by default.

**Value**

List containing two elements:
- Gene Keys: vector of gene IDs
- ChIP Targets: list of vectors, one per element in gr.list, containing the putative targets assigned. Each target is coded as its position in the vector 'Gene Keys'.

**Examples**

```r
data( 'DnaseHS_db', package = 'TFEA.ChIP' )
makeChIPGeneDB( DnaseHS_db, gr.list )
```

---

**matrixDB_to_listDB**

*Re-formatting ChIP-Gene database*

**Description**

Function to transform a ChIP-gene database from the former binary matrix to the current list-based format.

**Usage**

```r
matrixDB_to_listDB(Mat01)
```
Arguments

Mat01: Matrix[n,m] which rows correspond to all the human genes that have been assigned an Entrez ID, and its columns, to every ChIP-Seq experiment in the database. The values are 1 – if the ChIP-Seq has a peak assigned to that gene – or 0 – if it hasn’t –.

Value

List containing two elements: - Gene Keys: vector of gene IDs - ChIP Targets: list of vectors, one per ChIP-seq experiment in the database, containing the putative targets assigned. Each target is coded as its position in the vector ‘Gene Keys’.

Examples

Mat01 <- matrix(
    round( runif(9) ), nrow = 3,
    dimnames= list( paste0("Gene ", 1:3), paste0("ChIPseq ", 1:3)) )
matrixDB_to_listDB( Mat01 )

MetaData

TF-gene binding DB metadata

Description

A data frame containing information about the ChIP-Seq experiments used to build the TF-gene binding DB. Fields in the data frame:

- Accession: Accession ID of the experiment.
- Cell: Cell line or tissue.
- Cell Type: More information about the cells.
- Treatment
- Antibody
- TF: Transcription factor tested in the ChIP-Seq experiment.

Usage

data("MetaData")

Format

A data frame of 1060 observations of 6 variables
plot_CM

Makes an interactive html plot from an enrichment table.

Description

Function to generate an interactive html plot from a transcription factor enrichment table, output of the function `getCMstats`.

Usage

plot_CM(CM.statMatrix, plot_title = NULL, specialTF = NULL, TF_colors = NULL)

Arguments

- **CM.statMatrix**: Output of the function `getCMstats`. A data frame storing: Accession ID of every ChIP-Seq tested, Transcription Factor, Odds Ratio, p-value and adjusted p-value.
- **plot_title**: The title for the plot.
- **specialTF**: (Optional) Named vector of TF symbols -as written in the enrichment table- to be highlighted in the plot. The name of each element of the vector specifies its color group, i.e.: naming elements HIF1A and HIF1B as 'HIF' to represent them with the same color.
- **TF_colors**: (Optional) Colors to highlight TFs chosen in specialTF.

Value

plotly scatter plot.

Examples

```r
data('Genes.Upreg', package = 'TFEA.ChIP')
CM_list_UP <- contingency_matrix(Genes.Upreg)
stats_mat_UP <- getCMstats(CM_list_UP)
plot_CM(stats_mat_UP)
```

plot_ES

Plots Enrichment Score from the output of GSEA.run.

Description

Function to plot the Enrichment Score of every member of the ChIPseq binding database.
Usage

plot_ES(
  GSEA_result,
  LFC,
  plot_title = NULL,
  specialTF = NULL,
  TF_colors = NULL,
  Accession = NULL,
  TF = NULL
)

Arguments

GSEA_result  Returned by GSEA_run
LFC          Vector with log2(Fold Change) of every gene that has an Entrez ID. Arranged from higher to lower.
plot_title   (Optional) Title for the plot
specialTF    (Optional) Named vector of TF symbols -as written in the enrichment table- to be highlighted in the plot. The name of each element specifies its color group, i.e.: naming elements HIF1A and HIF1B as 'HIF' to represent them with the same color.
TF_colors    (Optional) Colors to highlight TFs chosen in specialTF.
Accession    (Optional) restricts plot to the indicated list dataset IDs.
TF            (Optional) restricts plot to the indicated list transcription factor names.

Value

Plotly object with a scatter plot -Enrichment scores- and a heatmap -log2(fold change) bar-.

Examples

data('GSEA.result','log2.FC',package = 'TFEA.ChIP')
TF.highlight <- c('E2F1' = 'E2F1')
col <- c('red')
plot_ES( GSEA.result, log2.FC, "Example", TF.highlight, col )

plot_RES  Plots all the RES stored in a GSEA_run output.

Description

Function to plot all the RES stored in a GSEA_run output.
Usage

plot_RES(
  GSEA_result,
  LFC,
  plot_title = NULL,
  line.colors = NULL,
  line.styles = NULL,
  Accession = NULL,
  TF = NULL
)

Arguments

GSEA_result  Returned by GSEA_run
LFC          Vector with log2(Fold Change) of every gene that has an Entrez ID. Arranged from higher to lower.
plot_title   (Optional) Title for the plot.
line.colors  (Optional) Vector of colors for each line.
line.styles  (Optional) Vector of line styles for each line ('solid'/'dash'/'longdash').
Accession    (Optional) restricts plot to the indicated list dataset IDs.
TF            (Optional) restricts plot to the indicated list transcription factor names.

Value

Plotly object with a line plot -running sums- and a heatmap -log2(fold change) bar-.

Examples

data('GSEA.result','log2.FC',package = 'TFEA.ChIP')
plot_RES(GSEA.result, log2.FC, TF = c('E2F4','E2F1'),
         Accession=c('ENCSR000DYY.E2F4.GM12878',
                     'ENCSR000EVJ.E2F1.HeLa-S3'))

preprocessInputData  Extracts data from a DESeqResults object or a data frame.

Description

Function to extract Gene IDs, logFoldChange, and p-val values from a DESeqResults object or data frame. Gene IDs are translated to ENTREZ IDs, if possible, and the resultant data frame is sorted according to decreasing log2(Fold Change). Translating gene IDs from mouse to their equivalent human genes is available using the variable "mode".

Usage

preprocessInputData(inputData, mode = "h2h")
Arguments

inputData | DESeqResults object or data frame. In all cases must include gene IDs. Data frame inputs should include 'pvalue' and 'log2FoldChange' as well.
mode | Specify the organism used: 'h2h' for homo sapiens gene IDs, 'm2m' for mouse gene IDs, or 'm2h' to get the corresponding human gene IDs from a mouse input.

Value

A table containing Entrez Gene IDs, LogFoldChange and p-val values (both raw p-value and fdr adjusted p-value), sorted by log2FoldChange.

Examples

data('hypoxia_DESeq', package='TFEA.ChIP')
preprocessInputData( hypoxia_DESeq )

rankTFs

Rank the TFs in the output from 'getCMstats'

Description

Rank the TFs in the output from 'getCMstats' using Wilcoxon rank-sum test or a GSEA-like approach.

Usage

rankTFs(
  resultsTable,  
  rankMethod = "gsea",  
  makePlot = FALSE,  
  plotTitle = "TF ranking"
)

Arguments

resultsTable | Output from the function 'getCMstats'
rankMethod | "wilcoxon" or "gsea".
makePlot | (Optional) For rankMethod="gsea". If TRUE, generates a plot for TFs with a p-value < 0.05.
plotTitle | (Optional) Title for the plot.
Select_genes

Value
data frame containing:

- For Wilcoxon rank-sum test: rank, TF name, test statistic (`wilc_W`), p-value, Freeman’s theta, epsilon-squared and effect size
- For GSEA-like ranking: TF name, enrichment score, argument, p-value, number of ChIPs

Examples
data('Genes.Upreg', package = 'TFEA.ChIP')
CM_list_UP <- contingency_matrix(Genes.Upreg)
stats_mat_UP <- getCMstats(CM_list_UP)
rankTFs( stats_mat_UP )

Select_genes  Extracts genes according to logFoldChange and p-val limits

Description
Function to extract Gene IDs from a dataframe according to the established limits for log2(FoldChange) and p-value. If possible, the function will use the adjusted p-value column.

Usage
Select_genes(
  GeneExpression_df,
  max_pval = 0.05,
  min_pval = 0,
  max_LFC = Inf,
  min_LFC = -Inf
)

Arguments

- GeneExpression_df: A data frame with the following fields: 'Gene', 'pvalue' or 'pval.adj', 'log2FoldChange'.

- max_pval: maximum p-value allowed, 0.05 by default.

- min_pval: minimum p-value allowed, 0 by default.

- max_LFC: maximum log2(FoldChange) allowed.

- min_LFC: minimum log2(FoldChange) allowed.

Value
A vector of gene IDs.

Examples
data('hypoxia', package='TFEA.ChIP')
Select_genes(hypoxia)
set_user_data

Sets the data objects as default.

Description

Function to set the data objects provided by the user as default to the rest of the functions.

Usage

set_user_data(metadata, ChIPDB)

Arguments

metadata Data frame/matrix/array containing the following fields: 'Name','Accession','Cell','Cell Type','Treatment','Antibody','TF'.

ChIPDB List containing two elements: - Gene Keys: vector of gene IDs - ChIP Targets: list of vectors, one per ChIP-seq experiment in the, database, containing the putative targets assigned. Each target is coded as its position in the vector 'Gene Keys'.

Value

sets the user's metadata table and TFBS matrix as the variables 'MetaData' and 'ChIPDB', used by the rest of the package.

Examples

data( 'MetaData', 'ChIPDB', package='TFEA.ChIP' )
# For this example, we will use the variables already included in the package.
set_user_data( MetaData, ChIPDB )

txt2GR

Function to filter a ChIP-Seq input.

Description

Function to filter a ChIP-Seq output (in .narrowpeak or MACS’s peaks.bed formats) and then store the peak coordinates in a GenomicRanges object, associated to its metadata.

Usage

txt2GR(fileTable, format, fileMetaData, alpha = NULL)
Arguments

fileTable  
data frame from a txt/tsv/bed file

format  
'narrowpeak', 'macs1.4' or 'macs2'. narrowPeak fields: 'chrom', 'chromStart', 'chromEnd', 'name', 'score', 'pValue', 'qValue', 'peak'. macs1.4 fields: 'chrom', 'chromStart', 'chromEnd', 'name', '-10*log10(p-value)'. macs2 fields: 'chrom', 'chromStart', 'chromEnd', 'name', '-log10(p-value)'.

fileMetaData  
Data frame/matrix/array containing the following fields: 'Name', 'Accession', 'Cell', 'Cell Type', 'Treatment', 'Antibody', 'TF'.

alpha  
max p-value to consider ChIPseq peaks as significant and include them in the database. By default alpha is 0.05 for narrow peak files and 1e-05 for MACS files.

Value

The function returns a GR object generated from the ChIP-Seq dataset input.

Examples

data('ARNT.peaks.bed', 'ARNT.metadata', package = 'TFEA.ChIP')
ARNT.gr <- txt2GR(ARNT.peaks.bed, 'macs1.4', ARNT.metadata)
Index

* datasets
  ARNT.metadata, 2
  ARNT.peaks.bed, 3
  ChIPDB, 4
  DnaseHS_db, 5
  Entrez_gene.IDs, 5
  Genes.Ureg, 6
  gr.list, 9
  GSEA.result, 9
  hypoxia, 13
  hypoxia_DESeq, 14
  log2.FC, 14
 MetaData, 16

  ARNTmetadata, 2
  ARNTpeaks.bed, 3
  ChIPDB, 4
  contingency_matrix, 4
  DnaseHS_db, 5
  Entrez_gene.IDs, 5
  GeneID2entrez, 6
  Genes_Ureg, 6
  get_chip_index, 8
  get_LFC_bar, 8
  getCMstats, 7
  gr.list, 9
  GSEA.result, 9
  GSEA_EnrichmentScore, 10
  GSEA_ESpermutations, 11
  GSEA.run, 11
  highlight_TF, 12
  hypoxia, 13
  hypoxia_DESeq, 14
  log2.FC, 14

  makeChIPGeneDB, 15
  matrixDB_to_listDB, 15
  MetaData, 16
  plot_CM, 17
  plot_ES, 17
  plot_RES, 18
  preprocessInputData, 19
  rankTfs, 20
  Select_genes, 21
  set_user_data, 22
  txt2GR, 22