Package 'TFEA.ChIP'

November 12, 2026

Version 1.32.0

Title TFEA.ChIP, a Tool Kit for Transcription Factor Enrichment

Imports GenomicRanges, IRanges, biomaRt, GenomicFeatures, GenomicRanges, grDevices, dplyr, stats, utils, R.utils, methods, org.Hs.eg.db, org.Mm.eg.db, rlang, ExperimentHub

Suggests knitr, rmarkdown, BiocStyle, S4Vectors, Seqinfo, meta, plotly, scales, tidyr, purrr, tibble, ggplot2, DESeq2, edgeR, limma, babelgene, BiocGenerics, ggrepel, rcompanion, TxDb.Hsapiens.UCSC.hg19.knownGene, TxDb.Hsapiens.UCSC.hg38.knownGene, AnnotationDbi, RColorBrewer, RUnit, testthat (>= 3.0.0)

Description Package to analyze transcription factor enrichment in a gene set using data from ChIP-Seq experiments.

Depends R (>= 4.2.0)

License Artistic-2.0

Encoding UTF-8

RoxygenNote 7.3.2

LazyData true

LazyDataCompression xz

VignetteBuilder knitr, BiocStyle

biocViews Transcription, GeneRegulation, GeneSetEnrichment, Transcriptomics, Sequencing, ChIPSeq, RNASeq, ImmunoOncology, GeneExpression, ChipOnChip

BuildVignettes true

URL https://github.com/yberda/TFEA.ChIP

BugReports https://github.com/yberda/TFEA.ChIP/issues

git_url https://git.bioconductor.org/packages/TFEA.ChIP

git_branch RELEASE_3_23

git_last_commit 2ec420f

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Yosra Berrouayel [aut, cre] (ORCID:
<https://orcid.org/0000-0002-0768-5933>),
Laura Puente-Santamaria [aut],
Luis del Peso [aut] (ORCID: <https://orcid.org/0000-0003-4014-5688>)

Maintainer Yosra Berrouayel <yosraberrouayel@gmail.com>

Contents

analysis_from_table . . . 3
ARNT.metadata . . . 4
ARNT.peaks.bed . . . 5
ChIPDB . . . 5
chip_metadata . . . 6
contingency_matrix . . . 6
DnaseHS_db . . . 7
Entrez.gene.IDs . . . 7
filter_expressed_TFs . . . 8
GeneID2entrez . . . 8
Genes.Upreg . . . 9
getCMstats . . . 9
get_chip_index . . . 10
gr.list . . . 10
GSEA.result . . . 11
GSEA_EnrichmentScore . . . 11
GSEA_ESpermutations . . . 12
GSEA_run . . . 13
highlight_TF . . . 14
hypoxia . . . 14
hypoxia_DESeq . . . 15
log2.FC . . . 15
makeChIPGeneDB . . . 16
matrixDB_to_listDB . . . 16
metaanalysis_fx . . . 17
MetaData . . . 18
plot_CM . . . 18
plot_ES . . . 19
plot_RES . . . 19
preprocessInputData . . . 21
rankTFs . . . 21
Select_genes . . . 22
set_user_data . . . 23
TF_ranking2 . . . 24
txt2GR . . . 24

Index **26**

`analysis_from_table` *Analysis from Input Table*

Description

Performs gene expression analysis, filtering genes and TFs based on specified thresholds. It calculates statistics using overrepresentation analysis (ORA) or gene set enrichment analysis (GSEA).

Usage

```
analysis_from_table(
  inputData,
  mode = "h2h",
  interest_min_LFC = -Inf,
  interest_max_LFC = Inf,
  control_min_LFC = -0.25,
  control_max_LFC = 0.25,
  interest_min_pval = 0,
  interest_max_pval = 0.05,
  control_min_pval = 0.5,
  control_max_pval = 1,
  expressed = TRUE,
  encodeFilter = FALSE,
  TFfilter = NULL,
  method = "ora"
)
```

Arguments

`inputData`	A data frame containing gene expression data.
`mode`	Character string specifying the mode: 'h2h', 'm2h', 'm2m'.
`interest_min_LFC`	Minimum LFC for selected genes of interest.
`interest_max_LFC`	Maximum LFC for selected genes of interest.
`control_min_LFC`	Minimum LFC for control genes.
`control_max_LFC`	Maximum LFC for control genes.
`interest_min_pval`	Minimum p-value for genes of interest.
`interest_max_pval`	Maximum p-value for genes of interest.
`control_min_pval`	Minimum p-value for control genes.

control_max_pval	Maximum p-value for control genes.
expressed	Logical; filter TFs expressed in input data.
encodeFilter	Logical; apply ENCODE filtering to ChIP-seq data.
TFfilter	Character vector of transcription factors to filter (optional).
method	Analysis method: 'ora' (overrepresentation) or 'gsea' (gene set enrichment).

Value

A matrix with calculated statistics (e.g., p-values, odds ratios).

Examples

```
data('hypoxia_DESeq',package='TFEA.ChIP')
res <- analysis_from_table(hypoxia_DESeq, interest_min_LFC = 1)
```

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

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

Format

a data frame of one row and 7 variables.

ARNT.peaks.bed *ChIP-Seq dataset*

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

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

Format

a data frame of 2140 rows and 4 variables.

ChIPDB *TF-Gene List*

Description

This dataset contains two elements: the first is "Gene Keys," which includes all the Entrez IDs, and the second is "ChIP Targets," a list containing information from multiple ChIP-Seq experiments. Each entry in this list contains the indices of the Entrez IDs from the first element that are associated with the peaks of that specific ChIP-Seq.

Usage

```
data("ChIPDB")
```

Format

A list with two elements: Gene Keys, a vector of Entrez IDs ChIP Targets, a list containing indices of Entrez IDs associated with peaks from various ChIP-Seq experiments.

chip_metadata *ChIP-seq transcription factor metadata*

Description

Metadata linking ChIP-seq experiments to their corresponding transcription factors. Each row provides the annotation of one TF.

Usage

```
data("chip_metadata")
```

Format

A data frame with 1267 rows. Columns include:

chip.name The TF name in the ChIP-seq experiment.

tf.name The HGNC gene symbol of the TF.

EntrezID Entrez IDs.

contingency_matrix *Compute 2x2 Contingency Matrices for ChIP-Seq Enrichment*

Description

This function computes 2x2 contingency matrices to assess the overlap between a test set of genes and TF binding targets derived from a ChIP-Seq database. The matrices are constructed for each ChIP experiment listed in the provided index, comparing the number of test and control genes that are bound (or not bound) by each transcription factor.

Usage

```
contingency_matrix(
  test_list,
  control_list = NULL,
  chip_index = get_chip_index()
)
```

Arguments

test_list	A vector of Entrez gene IDs representing the test group (e.g., differentially expressed genes).
control_list	A vector of Entrez gene IDs to be used as the control group. If NULL (default), all genes in the ChIP database not present in ‘test_list‘ are used.
chip_index	A data frame containing metadata for ChIP experiments. Must contain an ‘Accession‘ column matching entries in the ChIP database. Defaults to the result of ‘get_chip_index()‘. ‘test_list‘ and ‘control_list‘ to simulate a null distribution.

Value

A named list of 2x2 contingency matrices, one per ChIP experiment. Each matrix has: - Rows: Test group, Control group - Columns: Number of genes bound (Positive), and not bound (Negative) by the TF

Examples

```
data('Genes.Upreg', package = 'TFEA.ChIP')
cm_list <- contingency_matrix(Genes.Upreg)
```

DnaseHS_db *DHS databse*

Description

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

Usage

```
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

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

Format

Array of 2754 Entrez Gene IDs.

filter_expressed_TFs *Filter Expressed TFs*

Description

Filters TFs based on their expression status in the input dataset. This function identifies expressed TFs by intersecting the input gene list with the ‘chip_metadata‘ dataset.

Usage

```
filter_expressed_TFs(Table, chip_index, TFfilter = NULL, encodeFilter = FALSE)
```

Arguments

Table	A data frame containing gene expression data with a ‘Genes‘ column.
chip_index	A data frame containing ChIP-Seq dataset accession IDs and associated TFs.
TFfilter	(Optional) A character vector of TFs to filter.
encodeFilter	(Optional) Logical; if TRUE, applies ENCODE filtering to ChIP-Seq data.

Value

A filtered ‘chip_index‘ data frame containing only expressed TFs.

GeneID2entrez *Translates gene IDs from Gene Symbol or Ensembl ID to Entrez ID.*

Description

Translates mouse or human gene IDs from Gene Symbol or Ensembl Gene ID to Entrez Gene ID using AnnotationDbi.

Usage

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

Arguments

gene.IDs	Array of Gene Symbols or Ensembl Gene IDs.
return.Matrix	Logical. When TRUE, the function returns a matrix [n,2], one column with the gene symbols or Ensembl IDs, another with their respective Entrez IDs.
mode	Specify the organism used: ’h2h’ for human gene IDs, ’m2m’ for mouse gene IDs, or ’m2h’ for converting mouse to human gene IDs.

Value

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

Examples

```
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

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

Format

Array of 2754 Entrez Gene IDs.

---

getCMstats *Generate statistical parameters from contingency matrices*

Description

This function computes Fisher's exact test for each matrix in a list of contingency matrices (e.g., output from 'contingency_matrix'). It returns a data frame containing ChIP-Seq experiment accession IDs, tested transcription factors, p-values, odds ratios, and adjusted values.

Usage

```
getCMstats(CM_list, chip_index = get_chip_index())
```

Arguments

CM_list	A list of contingency matrices, typically from 'contingency_matrix'.
chip_index	A data frame with ChIP accession IDs and TFs from the 'get_chip_index' function. If not provided, the entire internal database is used.

Value

A data frame with the ChIP experiment ID, TF, p-value, odds-ratio, and other derived statistics.

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 containig 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'))
```

`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

```
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 runnign enrichment scores and the other, matches/missmatches along a gene list.

Usage

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

Format

list of three elements, an erihcment 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,
  gene.set,
  weighted.score.type = 0,
  correl.vector = NULL
)
```

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 correlations (such as signal to noise scores) corresponding to the genes in the gene list

Value

list of: 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

```
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.

Examples

```
GSEA_ESpermutations(gene.list=c('3091','2034','405','55818'),
gene.set=c('2034','112399','405'), perms=10)
```

GSEA_run *Function to run a GSEA analysis*

Description

Analyzes the distribution of TFBS across a sorted list of genes.

Usage

```
GSEA_run(
  gene.list,
  LFC,
  chip_index = get_chip_index(),
  get.RES = FALSE,
  RES.filter = NULL,
  perms = 1000
)
```

Arguments

gene.list	List of Entrez IDs ordered by their fold change.
LFC	Vector of log2(Fold Change) values.
chip_index	Data frame containing accession IDs of ChIPs and the tested TFs. If not provided, the entire internal database will be used.
get.RES	(Optional) boolean. If TRUE, stores Running Enrichment Scores for selected TFs.
RES.filter	(Optional) character vector. When get.RES == TRUE, specifies which TF's RES to store.
perms	(Optional) integer. Number of permutations for the enrichment test.

Value

A list containing: - Enrichment.table: Data frame with enrichment scores and p-values for each ChIP-Seq experiment. - RES (optional): List of running sums for each ChIP-Seq. - indicators (optional): List of binary vectors indicating matches between the gene list and gene sets.

Examples

```
data('hypoxia', package = 'TFEA.ChIP')
hypoxia <- preprocessInputData(hypoxia)
chip_index <- get_chip_index(TFfilter = c('HIF1A', 'EPAS1', 'ARNT'))
GSEA.result <- GSEA_run(hypoxia$Genes, hypoxia$log2FoldChange, chip_index,
get.RES = TRUE)
```

highlight_TF *Highlight special TFs in enrichment table.*

Description

Function to assign special TFs to specific colors and highlight them in the enrichment table.

Usage

```
highlight_TF(enrichTab, column, specialTF)
```

Arguments

enrichTab	Data frame containing the enrichment results.
column	The column index or name of the enrichment table used for matching TFs.
specialTF	A named vector of transcription factors to highlight in the plot.

Value

A list containing the updated highlight column and the color mapping for each TF.

hypoxia *RNA-Seq experiment*

Description

A data frame containing information of 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

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

Format

a data frame of 17527 observations of 4 variables.

hypoxia_DESeq *RNA-Seq experiment*

Description

A DESeqResults objetc containing information of 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 objtec

log2.FC *List of Entrez Gene IDs*

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

```
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

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

matrixDB_to_listDB *Re-formatting ChIP-Gene database*

Description

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

Usage

```
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 )
```

---

metaanalysis_fx *Perform Meta-analysis for each TF*

Description

Conducts a random-effects meta-analysis of odds ratios (OR) and standard errors (OR.SE) for each TF using the 'meta' package.

Usage

```
metaanalysis_fx(dat)
```

Arguments

dat	A data frame with columns: TF, OR, OR.SE, Accession, adj.pval.

Value

A list with: - summary: a data frame of ranked meta-analysis results per TF - results: a named list of raw meta-analysis objects from the 'meta' package

Examples

```
df <- data.frame(TF = c('A', 'A', 'A', 'B', 'B'),
                 OR = c(1, 1.2, 1.23, 4, 4.5),
                 OR.SE = c(1e-5, 5e-4, 2e-4, 1e-3, 1e-2),
                 Accession = c('Chip1', 'Chip2', 'Chip3', 'Chip4', 'Chip5'),
                 adj.pval = c(1e-5, 5e-4, 2e-4, 1e-3, 1e-2))
res <- metaanalysis_fx(df)
```

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 *Interactive HTML Plot for Transcription Factor Enrichment*

Description

Generates 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 containing Accession ID, Transcription Factor, Odds Ratio, p-value, and adjusted p-value.
plot_title	The title for the plot (default: "Transcription Factor Enrichment").
specialTF	(Optional) Named vector of TF symbols to be highlighted in the plot, allowing for grouped color representation.
TF_colors	(Optional) Colors to highlight TFs specified in specialTF.

Value

A plotly scatter plot.

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,
  Accession = NULL,
  TF = NULL
)
```

Arguments

GSEA_result	Returned by GSEA_run.
LFC	Vector with log2(Fold Change) of every gene with an Entrez ID, ordered from highest to lowest.
plot_title	(Optional) Title for the plot.
specialTF	(Optional) Named vector of transcription factors (TF) to highlight in the plot.
Accession	(Optional) Vector of dataset IDs to restrict the plot to.
TF	(Optional) Vector of transcription factor names to restrict the plot to.

Value

Plotly object combining scatter plot of enrichment scores and a log2(fold change) heatmap.

plot_RES *Plot Running Enrichment Scores (RES) and Log2 Fold Change (LFC)*

Description

This function plots the running enrichment scores (RES) from a GSEA result, with an additional bar plot showing the log2 fold change (LFC) of genes. The RES plot can be filtered by TFs or accession IDs.

Usage

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

Arguments

GSEA_result	List returned by GSEA_run, containing the enrichment table and RES.
LFC	Numeric vector containing the log2(Fold Change) of every gene with an Entrez ID, ordered from highest to lowest.
plot_title	(Optional) String specifying the title for the plot. Default is "Transcription Factor Enrichment".
line.colors	(Optional) Character vector specifying colors for each line in the RES plot. If NULL, default colors will be used.
line.styles	(Optional) Character vector specifying line styles for each RES line. Possible values are 'solid', 'dash', or 'longdash'.
Accession	(Optional) Character vector specifying accession IDs to restrict the plot to. If NULL, all accession IDs will be plotted.
TF	(Optional) Character vector specifying TF names to restrict the plot to. If NULL, all transcription factors will be plotted.

Value

A Plotly object containing two subplots: the top one showing the running enrichment scores (RES) for the filtered accession IDs or TFs, and the bottom one displaying the log2 fold change (LFC) as a bar plot.

Examples

```
data('GSEA.result', 'log2.FC', package = 'TFEA.ChIP')
GSEA_result <- GSEA.result
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, Gene Symbols, 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.

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 character 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 contaning 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 )
```

TF_ranking2 *Meta-analysis ranking of transcription factors*

Description

Results of a meta-analysis performed across multiple ChIP-seq experiments for transcription factors.

Usage

```
data("TF_ranking2")
```

Format

A list with two elements:

summary A data frame with 1236 rows, each corresponding to a TF.

details A list of 1236 elements, one per TF, containing the full meta-analysis results.

Details

The object is a list with two components:

- **summary**: A data frame summarizing the meta-analysis results for each TF (one row per TF).
- **details**: A list where each element contains the complete meta-analysis output for a given TF.

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 contaning 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, 4
ARNT.peaks.bed, 5
chip_metadata, 6
ChIPDB, 5
DnaseHS_db, 7
Entrez.gene.IDs, 7
Genes.Upreg, 9
gr.list, 10
GSEA.result, 11
hypoxia, 14
hypoxia_DESeq, 15
log2.FC, 15
MetaData, 18
TF_ranking2, 24

analysis_from_table, 3
ARNT.metadata, 4
ARNT.peaks.bed, 5

chip_metadata, 6
ChIPDB, 5
contingency_matrix, 6

DnaseHS_db, 7

Entrez.gene.IDs, 7

filter_expressed_TFs, 8

GeneID2entrez, 8
Genes.Upreg, 9
get_chip_index, 10
getCMstats, 9
gr.list, 10
GSEA.result, 11
GSEA_EnrichmentScore, 11
GSEA_ESpermutations, 12
GSEA_run, 13

highlight_TF, 14
hypoxia, 14
hypoxia_DESeq, 15

log2.FC, 15

makeChIPGeneDB, 16
matrixDB_to_listDB, 16
metaanalysis_fx, 17
MetaData, 18

plot_CM, 18
plot_ES, 19
plot_RES, 19
preprocessInputData, 21

rankTFs, 21

Select_genes, 22
set_user_data, 23

TF_ranking2, 24
txt2GR, 24