Package 'tepr'

July 8, 2026

Title Transcription Elongation Profiling

Version 1.1.17

Description The general principle relies on calculating the cumulative signal of nascent RNA sequencing over the gene body of any given gene or transcription unit. 'tepr' can identify transcription attenuation sites by comparing profile to a null model which assumes uniform read density over the entirety of the transcription unit. It can also identify increased or diminished transcription attenuation by comparing two conditions. Besides rigorous statistical testing and high sensitivity, a major feature of 'tepr' is its ability to provide the elongation pattern of each individual gene, including the position of the main attenuation point when such a phenomenon occurs. Using 'tepr', users can visualize and refine genome-wide aggregated analyses of elongation patterns to robustly identify effects specific to subsets of genes. These metrics are suitable for internal comparisons (between genes in each condition) and for studying elongation of the same gene in different conditions or comparing it to a perfect theoretical uniform elongation.

Depends R (>= 4.4)

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

VignetteBuilder knitr

Suggests knitr, rmarkdown, testthat

Imports dplyr, parallel, tidyr, tidyselect, ggplot2, pracma, stats, ggrepel, matrixStats, rlang, magrittr, purrr, rtracklayer, GenomicRanges, GenomeInfoDb, valr, tibble, utils, methods, MASS

Collate 'allauc.R' 'attenuation.R' 'averageandfilterexprs.R' 'countna.R' 'genesECDF.R' 'kneeid.R' 'meandifference.R' 'plotauc.R' 'plotecdf.R' 'plothistoknee.R' 'plotmetagenes.R' 'plotmulti.R' 'preprocessing-blacklisthighmap-utils.R' 'preprocessing-blacklisthighmap.R' 'preprocessing-createtablescores.R' 'preprocessing-makewindows.R' 'preprocessing-retrieveanno.R' 'preprocessing.R' 'tepr.R' 'universegroup.R' 'utils.R'

NeedsCompilation no

Author Nicolas Descostes [cre],
Victor Billon [aut],
Gael Cristofari [aut]

Maintainer Nicolas Descostes <nicolas.descostes@gmail.com>

Repository CRAN

Date/Publication 2026-07-08 17:40:02 UTC

Contents

allauc 2
attenuation 4
averageandfilterexprs 6
blacklisthighmap 7
checkexptab 9
countna 10
createtablescores 11
genesECDF 13
kneeallconds 14
kneeid 16
makewindows 17
meandifference 18
plotauc 19
plotecdf 22
plothistoknee 24
plotmetagenes 25
plotmulti 27
preprocessing 29
retrieveanno 31
retrievechrom 32
showallcomp 33
tepr 33
teprmulti 37
universegroup 39

Index **42**

`allauc` *Calculate Area Under Curve (AUC) and Differences of AUC for Transcript Data*

Description

This function computes the Area Under Curve (AUC) and the differences of AUC between two conditions for a list of transcript data. It supports parallel computation for efficiency. If only one condition is given, the differences are not computed.

Usage

```
allauc(bytranslistmean, expdf, nbwindows, nbcpu = 1,
 controlcondname = "ctrl", stresscondname = "HS", showtime = FALSE,
 verbose = TRUE)
```

Arguments

bytranslistmean	A list of data frames, each containing transcript level data with mean values for one or more conditions.
expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
nbwindows	An integer specifying the number of windows to consider for AUC calculations.
nbcpu	An integer specifying the number of CPU cores to use for parallel processing on bytranslistmean. Defaults to 1.
controlcondname	A string specifying the name of the control condition Defaults to "ctrl".
stresscondname	A string specifying the name of the stress condition. Defaults to "HS".
showtime	A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to FALSE.
verbose	A logical value indicating whether to print progress messages Defaults to TRUE.

Details

The function first checks if exactly two conditions are present in 'expdf'. If so, it computes the differences in AUC between the two conditions using a Kolmogorov-Smirnov test. It then calculates the AUC for all conditions against a reference line (y=x). Results are merged by transcript and include adjusted p-values.

Value

A data frame containing the AUC and dAUC results for each transcript, along with associated statistical information.

See Also

[genesECDF]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Reading tables
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
```

```
## Computing intermediate steps
avfilt <- averageandfilterexprs(expdf, transdf, expthres,
        showtime = FALSE, verbose = FALSE)
ecdf <- genesECDF(avfilt, verbose = FALSE)
resecdf <- ecdf[[1]]
nbwindows <- ecdf[[2]]
meandiff <- meandifference(resecdf, expdf, nbwindows,
    verbose = FALSE)
bytranslistmean <- split(meandiff, factor(meandiff$transcript))

## Testing allauc
res <- allauc(bytranslistmean, expdf, nbwindows, verbose = FALSE)
```

`attenuation` *Calculate Attenuation from AUC and Other Transcript Features*

Description

This function computes the attenuation values for each window of each transcript based on the data frames obtained with the functions 'allauc', 'kneeid', and 'countna'.

Usage

```
attenuation(allaucdf, kneedf, matnatrans, bytranslistmean, expdf, dfmeandiff,
nbcpu = 1, significant = FALSE, replaceval = NA, pval = 0.1,
showtime = FALSE, verbose = TRUE)
```

Arguments

`allaucdf`	A data frame containing AUC results for transcripts (see allauc).
`kneedf`	A data frame containing the inflection points (see kneeid).
`matnatrans`	A data frame containing the number of missing values per transcript (see countna).
`bytranslistmean`	A list of data frames with mean values by transcripts.
`expdf`	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
`dfmeandiff`	A data frame containing means and differences in mean values, if more than one condition. (see meandifference).
`nbcpu`	An integer specifying the number of CPU cores to use for parallel processing. The parallelization is done on bytranslistmean whose number of elements is equal to the number of lines provided as input of 'averageandfilterexprs'. Defaults to 1.
`significant`	A logical indicating whether to filter out non-significant attenuation values. Defaults to `FALSE`.
`replaceval`	A value to replace non-significant attenuation values Defaults to `NA`.

pval	A numeric value specifying the p-value threshold for significance of the KS test. Defaults to 0.1.
showtime	A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to FALSE.
verbose	A logical value indicating whether to print progress messages Defaults to TRUE.

Details

The function merges several data frames to create a comprehensive dataset for each transcript. It computes mean values for the "up" and "down" segments of the transcript. The direction is determined by comparing the coordinates to the knee values. up = coord < knee and down = coord > knee. The up and down indexes are then retrieved and the attenuation scores are computed as: att <- 100 - downmean / upmean * 100

Value

A data frame containing the computed attenuation values along with associated transcript information.

See Also

[allauc()], [kneeid()], [countna()], [meandifference()]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Reading tables
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)

## Computing intermediate steps
avfilt <- averageandfilterexprs(expdf, transdf, expthres,
       showtime = FALSE, verbose = FALSE)
rescountna <- countna(avfilt, expdf, nbcpu = 1, verbose = FALSE)
ecdf <- genesECDF(avfilt, verbose = FALSE)
resecdf <- ecdf[[1]]
nbwindows <- ecdf[[2]]
resmeandiff <- meandifference(resecdf, expdf, nbwindows,
    verbose = FALSE)
bytranslistmean <- split(resmeandiff, factor(resmeandiff$transcript))
resknee <- kneeid(bytranslistmean, expdf, verbose = FALSE)
resauc <- allauc(bytranslistmean, expdf, nbwindows, verbose = FALSE)

## Testing attenuation
resatt <- attenuation(resauc, resknee, rescountna, bytranslistmean, expdf,
        resmeandiff, verbose = FALSE)
```

averageandfilterexprs *Calculate Average Expression and Filter Transcript Data*

Description

This function calculates the average expression levels for transcripts from a provided expression data frame and filters out transcripts based on a specified expression threshold. The function also renames the columns in the output data frame to include mean expression values.

Usage

```
averageandfilterexprs(expdf, alldf, expthres, showtime = FALSE,
verbose = TRUE)
```

Arguments

expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
alldf	A data frame containing all transcript-related information, including biotype, chromosome, coordinates, transcript, gene, strand, window, ID and scores retrieved from the bedgraph files.
expthres	A numeric value specifying the expression threshold. Transcripts with average expression values below this threshold will be filtered out from the returned transcript vector.
showtime	A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to FALSE.
verbose	A logical value indicating whether to print progress messages Defaults to TRUE.

Details

If no transcript is selected as expressed, the function throws an error.

Value

A list containing:

maintable	The original data frame containing all transcript data.
exptranstab	A character vector of transcripts that meet the filtering criteria.

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Reading tables
expdf <- read.csv(exppath)
```

```
transdf <- read.delim(transpath, header = FALSE)

## Testing averageandfilterexprs
resavfilt <- averageandfilterexprs(expdf, transdf, expthres,
       showtime = FALSE, verbose = FALSE)
```

---

blacklisthighmap *Blacklist High Mappability Regions in Genomic Data*

Description

This function processes genomic data to remove scores that fall within blacklisted regions or have low mappability, and computes weighted means for overlapping windows. The process ensures the integrity of genomic scores by focusing on high mappability regions and excluding blacklisted intervals.

Usage

```
blacklisthighmap(maptrackpath, blacklistpath, exptabpath,
   nbcputrans, allwindowsbed, windsize, genomename, saveobjectpath = NA,
   tmpfold = file.path(tempdir(), "tmptepr"), reload = FALSE, showtime = FALSE,
   showmemory = FALSE, chromtab = NA, forcechrom = FALSE, verbose = TRUE)
```

Arguments

maptrackpath	Character string. Path to the mappability track file.
blacklistpath	Character string. Path to the blacklist regions file.
exptabpath	Path to the experiment table file containing a table with columns named 'condition', 'replicate', 'strand', and 'path'.
nbcputrans	Number of CPU cores to use for transcript-level operations.
allwindowsbed	Data frame. BED-formatted data frame obtained with the function 'makewindows'.
windsize	An integer specifying the size of the genomic windows.
genomename	Character string. A valid UCSC genome name. It is used to retrieve chromosome metadata, such as names and lengths.
saveobjectpath	Path to save intermediate R objects. Default is 'NA' and R objects are not saved.
tmpfold	A character string specifying the temporary folder for saving output files. The temporary files contain the scores for each bedgraph on each chromosome. Default is `file.path(tempdir(), "tmptepr")`.
reload	Logical. If 'TRUE', reloads existing saved objects to avoid recomputation. Default is 'FALSE'. If the function failed during object saving, make sure to delete the corresponding object.
showtime	A logical value indicating whether to display processing time.

showmemory	A logical value indicating whether to display memory usage during processing.
chromtab	A Seqinfo object retrieved with the rtracklayer method SeqinfoForUCSCGenome. If NA, the method is called automatically. Default is NA.
forcechrom	Logical indicating if the presence of non-canonical chromosomes in chromtab (if not NA) should trigger an error. Default is FALSE.
verbose	A logical value indicating whether to display detailed processing messages.

Details

The 'blacklisthighmap' function iterates through chromosomes, processes genomic scores by removing those overlapping with blacklisted regions, and ensures that scores within windows are computed using a weighted mean when overlaps occur. The function uses parallel processing for efficiency and supports saving (saveobjectpath) and reloading (reload) intermediate results to optimize workflow.

The main steps include: - Reading and processing bedGraph values. - Removing scores overlapping with blacklisted or low mappability regions. - Computing weighted means for overlapping scores in genomic windows. - Saving the processed results to specified path (tmpfold).

If chromtab is left to NA, the chromosome information is automatically retrieved from the UCSC server using 'genomename'. Otherwise, the Seqinfo object can be retrieved with: chromtab <- rtracklayer::SeqinfoForUCSCGenome(genomename)

Value

This function does not return a value directly. It saves intermediate results to 'tmpfold'. These intermediates files are then combined by the function 'createtablescores'.

See Also

[createtablescores][makewindows]

Examples

```
exptabpath <- system.file("extdata", "exptab-preprocessing.csv", package="tepr")
gencodepath <- system.file("extdata", "gencode-chr13.gtf", package = "tepr")
maptrackpath <- system.file("extdata", "k50.umap.chr13.hg38.0.8.bed",
    package = "tepr")
blacklistpath <- system.file("extdata", "hg38-blacklist-chr13.v2.bed",
    package = "tepr")
windsize <- 200
genomename <- "hg38"
chromtabtest <- rtracklayer::SeqinfoForUCSCGenome(genomename)
allchromvec <- GenomeInfoDb::seqnames(chromtabtest)
chromtabtest <- chromtabtest[allchromvec[which(allchromvec == "chr13")], ]

## Copying bedgraphs to the current directory
expdfpre <- read.csv(exptabpath)
bgpathvec <- sapply(expdfpre$path, function(x) system.file("extdata", x,
    package = "tepr"))
expdfpre$path <- bgpathvec
```

```
write.csv(expdfpre, file = "exptab-preprocessing.csv", row.names = FALSE,
    quote = FALSE)
exptabpath <- "exptab-preprocessing.csv"

## Necessary result to call blacklisthighmap
allannobed <- retrieveanno(exptabpath, gencodepath, verbose = FALSE)
allwindowsbed <- makewindows(allannobed, windsize, verbose = FALSE)

## Test blacklisthighmap
blacklisthighmap(maptrackpath, blacklistpath, exptabpath,
    nbcputrans = 1, allwindowsbed, windsize, genomename,
    chromtab = chromtabtest, verbose = FALSE)
```

---

checkexptab *Check Validity of Experiment Table*

Description

The ‘checkexptab‘ function verifies the structure and content of an experiment table to ensure it meets specific formatting requirements. It checks for the presence of required columns, and validates that the ‘direction‘ and ‘strand‘ columns contain only allowable values.

Usage

```
checkexptab(exptab)
```

Arguments

exptab	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.

Details

The function performs the following checks: - The column names of ‘exptab‘ must match exactly: ‘"condition"‘, ‘"replicate"‘, ‘"direction"‘, and ‘"strand"‘. - The ‘direction‘ column must contain only ‘"forward"‘ and ‘"reverse"‘. - The ‘strand‘ column must contain only ‘"plus"‘ and ‘"minus"‘.

Value

If the experiment table is valid, the function returns ‘NULL‘. If the table is invalid, the function throws an error specifying the issue.

Examples

```
# Create a valid experiment table
exptab <- data.frame(
  condition = c("cond1", "cond2"),
  replicate = c(1, 1),
  direction = c("forward", "reverse"),
  strand = c("plus", "minus"),
  path = c("toto/", "toto/"))
checkexptab(exptab)  # Should pass without errors

# Invalid experiment table (wrong column names)
invalid_exptab <- data.frame(
    cond = c("cond1", "cond2"),
    rep = c(1, 1),
    dir = c("forward", "reverse"),
    str = c("+", "-"),
    paths = c("toto/", "toto/"))
try(checkexptab(invalid_exptab))
```

---

countna *Count NA values per transcript and condition*

Description

This function takes a list of expression data frames, a condition information data frame, and counts the number of NA values for each transcript based on strand and condition. NA represent missing scores that were filtered out from the black list and mappability track. The function operates in parallel on transcripts to speed up the process using multiple CPU cores.

Usage

```
countna(allexprsdfs, expdf, nbcpu = 1, showtime = FALSE, verbose = TRUE)
```

Arguments

allexprsdfs	A list of data frames containing expression data. The first element is assumed to be the main table. The second element is a vector of transcript names that passed the filtering of 'averageandfilterexprs'.
expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
nbcpu	An integer specifying the number of CPU cores to use for parallel computation on transcripts. The number of transcripts is equal to the number of lines provided as input of 'averageandfilterexprs'. Defaults to 1.
showtime	A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to FALSE.
verbose	A logical flag indicating whether to print progress messages. Defaults to TRUE.

Value

A data frame where each row corresponds to a transcript, along with its associated gene, strand, and the count of NA values.

See Also

[averageandfilterexprs]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Calculating averageandfilterexprs to call countNA
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
avfilttest <- averageandfilterexprs(expdf, transdf, expthres,
        showtime = FALSE, verbose = FALSE)

## Testing countna
res <- countna(avfilttest, expdf, nbcpu = 1, verbose = FALSE)
```

---

createtablescores *Create a Unified Table of Scores*

Description

This function processes and combines table scores of each bedgraph and each chromosome stored in the temporary folder into a unified table.

Usage

```
createtablescores(tmpfold, exptabpath, showmemory = FALSE, showtime = FALSE,
  savefinaltable = TRUE, finaltabpath = tempdir(), finaltabname = "anno.tsv",
 verbose)
```

Arguments

tmpfold	A string specifying the temporary folder containing the score files created with the function 'blacklisthighmap'.
exptabpath	Path to the experiment table file containing a table with columns named 'condition', 'replicate', 'strand', and 'path'.
showmemory	Logical; if 'TRUE', memory usage is printed during processing. Default is 'FALSE'.

showtime	Logical; if 'TRUE', the execution time of the function is printed. Default is 'FALSE'.
savefinaltable	Logical; if 'TRUE', the resulting table is saved to disk. Default is 'TRUE'.
finaltabpath	A string specifying the directory where the final table should be saved. Default is tempdir().
finaltabname	A string specifying the name of the final table file. Default is '"anno.tsv"'.
verbose	Logical; if 'TRUE', detailed messages are printed during execution.

Details

This function first merges files belonging to the same experiment and direction. These files are combined into a single table providing two columns per experiment. The first gives the name of the experiment and the second the scores. The resulting table also includes annotations for each transcript.

Value

A data frame containing the unified table of scores.

See Also

[blacklisthighmap]

Examples

```
exptabpath <- system.file("extdata", "exptab-preprocessing.csv", package="tepr")
gencodepath <- system.file("extdata", "gencode-chr13.gtf", package = "tepr")
maptrackpath <- system.file("extdata", "k50.umap.chr13.hg38.0.8.bed",
package = "tepr")
blacklistpath <- system.file("extdata", "hg38-blacklist-chr13.v2.bed",
    package = "tepr")
tmpfoldpath <- file.path(tempdir(), "tmptepr")
windsize <- 200
genomename <- "hg38"
chromtabtest <- rtracklayer::SeqinfoForUCSCGenome(genomename)
allchromvec <- GenomeInfoDb::seqnames(chromtabtest)
chromtabtest <- chromtabtest[allchromvec[which(allchromvec == "chr13")], ]

## Copying bedgraphs to the current directory
expdfpre <- read.csv(exptabpath)
bgpathvec <- sapply(expdfpre$path, function(x) system.file("extdata", x,
    package = "tepr"))
expdfpre$path <- bgpathvec
write.csv(expdfpre, file = "exptab-preprocessing.csv", row.names = FALSE,
    quote = FALSE)
exptabpath <- "exptab-preprocessing.csv"

## Necessary result to call createtablescores
allannobed <- retrieveanno(exptabpath, gencodepath, verbose = FALSE)
allwindowsbed <- makewindows(allannobed, windsize, verbose = FALSE)
```

```
blacklisthighmap(maptrackpath, blacklistpath, exptabpath, nbcputrans = 1,
    allwindowsbed, windsize, genomename = genomename, chromtab = chromtabtest,
    verbose = FALSE)

## Calling the function to test
finaltabtest <- createtablescores(tmpfold = tmpfoldpath, exptabpath,
    savefinaltable = FALSE, verbose = FALSE)
```

---

genesECDF *Compute ECDF for Genes Based on Expression Data*

Description

This function calculates the empirical cumulative distribution function (ECDF) for expressed genes across multiple transcripts. It processes the expression data to filter out non-expressed transcripts, compute ECDF values for each transcript, and combine the results into a unified data frame. The function operates in parallel for speed optimization.

Usage

```
genesECDF(allexprsdfs, nbcpu = 1, rounding = 10,
showtime = FALSE, verbose = TRUE)
```

Arguments

allexprsdfs	A list of data frames where the first element is the main expression data frame and the second element contains the names of the expressed transcripts (see 'averageandfilterexprs').
nbcpu	An integer specifying the number of CPU cores to use for parallel computation. Default is 1.
rounding	An integer specifying the rounding factor for computing ECDF. Default is 10.
showtime	A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to FALSE.
verbose	A logical flag indicating whether to print progress messages. Default is TRUE.

Details

The function performs several steps:

1. Filters the main expression table to retain only the expressed transcripts.
2. Splits the data by each transcript.
3. For each transcript, computes ECDF values for the score columns while respecting the strand orientation ("plus" or "minus").
4. Combines the ECDF results into a final data frame.

The function uses parallel processing to compute ECDF for each transcript simultaneously, making it faster on systems with multiple CPU cores.

Value

A list containing two elements:

concatdf	A data frame with ECDF results for each transcript.
nbrows	An integer indicating the number of rows in each transcript table.

See Also

[averageandfilterexprs]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Calculating averageandfilterexprs and countNA to call genesECDF
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
avfilttest <- averageandfilterexprs(expdf, transdf, expthres,
        showtime = FALSE, verbose = FALSE)
countnatest <- countna(avfilttest, expdf, nbcpu = 1, verbose = FALSE)

## Testing genesECDF
resecdf <- genesECDF(avfilttest, verbose = FALSE)
```

---

kneeallconds *Calculate knee for each condition separately*

Description

This function calculate knees for each condition separately

Usage

```
kneeallconds(alldf, expdf, expthres, nbcpu = 1, rounding = 10,
showtime = FALSE, verbose = TRUE)
```

Arguments

alldf	A data frame containing all transcript-related information, including biotype, chromosome, coordinates, transcript, gene, strand, window, ID and scores retrieved from the bedgraph files.
expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.

expthres	A numeric value specifying the expression threshold. Transcripts with average expression values below this threshold will be filtered out from the returned transcript vector.
nbcpu	An integer specifying the number of CPU cores to use for parallel computation on transcripts. The number of transcripts is equal to the number of lines provided as input of 'averageandfilterexprs'. Defaults to 1.
rounding	An integer specifying the rounding factor for computing ECDF. Default is 10.
showtime	A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to FALSE.
verbose	A logical flag indicating whether to print progress messages. Defaults to TRUE.

Details

The kneeallconds function calls successively for each condition:

- averageandfilterexprs This function calculates the average expression levels for transcripts from alldf that was obtained with the 'preprocessing' function. It filters out transcripts based on the 'expthres' expression threshold. The function also renames the columns in the output data frame to include mean expression values. It returns a list containing the original alldf with the mean columns added and a character vector of transcripts that meet the filtering criteria.
- genesECDF It takes the result of 'averageandfilterexprs' as input and a) filters the main expression table to retain only the expressed transcripts; b) Splits the data by each transcript; c) For each transcript, computes ECDF values for the score columns while respecting the strand orientation ("plus" or "minus"); d) Combines the ECDF results into a final data frame. It returns a list containing two elements: A data frame with ECDF results for each transcript (concatdf) and an integer indicating the number of rows in each transcript table.
- meandifference It takes the result of 'genesECDF' as input and calculates the mean values, mean Fx (ECDF) and ECDF differences (Fx) for expression data, across different experimental conditions. It returns a data frame that contains, for each condition: mean values for the "value" and "Fx" columns (e.g., mean_value_ctrl, mean_Fx_ctrl) and the differences between the Fx column and coordinate ratios (e.g., diff_Fx_ctrl).
- Splitting Split the results of 'meandifference' by transcripts and stores the list into bytranslistmean.
- allauc It uses the previously computed variable 'bytranslistmean' and it computes the Area Under Curve (AUC) and the differences of AUC between two conditions for a list of transcript data. It returns a data frame containing the AUC and dAUC results for each transcript, along with associated statistical information.
- kneeid It uses the previously computed variable 'bytranslistmean' and identifies the knee point (i.e., point of maximum change) and the maximum difference in the empirical cumulative distribution function (ECDF) for each transcript, across different experimental conditions. It returns a data frame where each row corresponds to a transcript and contains the coordinates of the knee point and the maximum ECDF difference for each condition.

Value

A data.frame with the columns transcript, gene, and strand window_size. For each condition 'cond': AUC_cond, p_AUC_cond, D_AUC_cond, MeanValueFull_cond, adjFDR_p_AUC_cond, knee_AUC_cond, and max_diff_Fx_cond.

See Also

[averageandfilterexprs()], [genesECDF()], [meandifference()], [kneeid()]

Examples

```
exptabpath <- system.file("extdata", "exptab.csv", package="tepr")
alldfpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expdf <- read.csv(exptabpath)
alldf <- read.delim(alldfpath, header = FALSE)
expthres <- 0.1
kneedf <- kneeallconds(alldf, expdf, expthres, verbose = FALSE)
```

---

kneeid *Identify the Knee and Max ECDF Differences for Each Transcript*

Description

This function identifies the knee point (i.e., point of maximum change) and the maximum difference in the empirical cumulative distribution function (ECDF) for each transcript, across different experimental conditions.

Usage

```
kneeid(transdflist, expdf, nbcpu = 1, showtime = FALSE, verbose = TRUE)
```

Arguments

transdflist	A list of data frames where each data frame contains transcript data with ECDF values for each condition.
expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
nbcpu	An integer specifying the number of CPU cores to use for parallel computation. The parallelization is performed on the elements of transdflist. Defaults to 1.
showtime	A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to FALSE.
verbose	A logical flag indicating whether to print progress messages. Defaults to TRUE.

Value

A data frame where each row corresponds to a transcript and contains the coordinates of the knee point and the maximum ECDF difference for each condition.

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Calculating necessary results
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
avfilt <- averageandfilterexprs(expdf, transdf, expthres,
        showtime = FALSE, verbose = FALSE)
ecdf <- genesECDF(avfilt, verbose = FALSE)
resecdf <- ecdf[[1]]
nbwindows <- ecdf[[2]]
meandiff <- meandifference(resecdf, expdf, nbwindows,
    verbose = FALSE)
bytranslistmean <- split(meandiff, factor(meandiff$transcript))

## Testing kneeid
reskneeid <- kneeid(bytranslistmean, expdf, verbose = FALSE)
```

`makewindows` *Split Gene Annotations into Fixed-Size Windows*

Description

This functions uses the annotations filtered from gencode (see retrieveanno). It removes any ensembl names containing "PAR_Y". It filters out intervals smaller than windsize and splits each transcript into "windsize" windows.

Usage

```
makewindows(allannobed, windsize, nbcputrans = 1, verbose = TRUE,
   saveobjectpath = NA, showtime = FALSE)
```

Arguments

`allannobed`	A data frame which is the result of 'retrieveanno'.
`windsize`	An integer specifying the number of windows into which each gene annotation should be divided.
`nbcputrans`	Number of CPU cores to use for transcript-level operations. Defaults to 1.
`verbose`	A logical value indicating whether to display progress messages. Defaults to 'TRUE'.
`saveobjectpath`	A character string specifying the directory path where the output object should be saved as an '.rds' file. If 'NA', the object is not saved. Defaults to 'NA'.
`showtime`	A logical value indicating whether to display the runtime of the function. Defaults to 'FALSE'.

Details

The function filters out annotations with intervals smaller than the specified number of windows (‘windsize‘). It uses vectorized operations to efficiently split transcripts into fixed-size windows. The result includes metadata for each window, such as its chromosome, start and end coordinates, associated gene, and the window number.

Intermediate functions, such as ‘.computewindflist‘ and ‘.divideannoinwindows‘, handle computation and validation of windows. Gene intervals with the "PAR_Y" tag are excluded from the analysis.

Value

A data frame containing the split windows for each gene annotation. The output includes fields such as ‘biotype‘, ‘chr‘, ‘coor1‘, ‘coor2‘, ‘transcript‘, ‘gene‘, ‘strand‘, and ‘window‘.

See Also

[retrieveanno]

Examples

```
exptabpath <- system.file("extdata", "exptab-preprocessing.csv", package="tepr")
gencodepath <- system.file("extdata", "gencode-chr13.gtf", package = "tepr")
windsize <- 200

## Necessary result to call makewindows
allannobed <- retrieveanno(exptabpath, gencodepath, verbose = FALSE)

## Calling makewindows
allwindowsbed <- makewindows(allannobed, windsize, verbose = FALSE)
```

---

meandifference *Compute Mean and Differences of Scores for Each Condition*

Description

This function calculates the mean values, mean Fx (ECDF) and ECDF differences (Fx) for expression data, across different experimental conditions. If only one condition is provided, skips computation of mean differences.

Usage

```
meandifference(resultsecdf, expdf, nbwindows, showtime = FALSE,
verbose = TRUE)
```

Arguments

resultsecdf	A data frame containing ECDF results for each transcript and condition (see genesECDF).
expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
nbwindows	An integer representing the number of windows (or segments) in each transcript.
showtime	A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to FALSE.
verbose	A logical flag indicating whether to print progress messages. Defaults to TRUE.

Value

A data frame that contains, for each condition:

- Mean values for the "value" and "Fx" columns (e.g., mean_value_ctrl, mean_Fx_ctrl).
- Differences between the Fx column and coordinate ratios (e.g., diff_Fx_ctrl).

If only one condition is provided, the differences on mean columns are not performed.

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Calculating necessary results
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
avfilt <- averageandfilterexprs(expdf, transdf, expthres,
        showtime = FALSE, verbose = FALSE)
countna <- countna(avfilt, expdf, nbcpu = 1, verbose = FALSE)
ecdf <- genesECDF(avfilt, verbose = FALSE)
resecdf <- ecdf[[1]]
nbwindows <- ecdf[[2]]

## Testing meandifference
meandifftest <- meandifference(resecdf, expdf, nbwindows,
    verbose = FALSE)
```

---

plotauc *Plot AUC Comparison Between Conditions*

Description

This function generates scatterplots comparing the area under the curve (AUC) for control and stress conditions, with an option to highlight specific genes or groups. The plot can be saved as a file or displayed interactively.

Usage

```
plotauc(tab, expdf, genevec = NA, auc_ctrlname = "AUC_ctrl",
auc_stressname = "AUC_HS",
pvalkstestcolname = "adjFDR_p_dAUC_Diff_meanFx_HS_ctrl",
labelx = "AUC in Control", labely = "AUC in Stress", axismin_x = -10,
axismax_x = 100, axismin_y = -10, axismax_y = 100, maintitle = "",
subtitle = "", legendpos = "bottom", formatname = "pdf", outfold = tempdir(),
outfile = "AUCcompare_pval", plottype = "pval", plot = FALSE,
universename = "Universe", groupname = "Group", verbose = TRUE)
```

Arguments

tab	A data frame containing the AUC values for control and stress conditions, and other columns required for plotting (e.g., p-values or group memberships, see allauc).
expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
genevec	A vector of gene names to highlight on the plot, applicable when plottype is set to "pval". Default is NA.
auc_ctrlname	The column name in tab for the AUC under control conditions. Default is "AUC_ctrl".
auc_stressname	The column name in tab for the AUC under stress conditions. Default is "AUC_HS".
pvalkstestcolname	The column name in tab for the adjusted FDR p-values from the KS test. Default is "adjFDR_p_dAUC_Diff_meanFx_HS_ctrl".
labelx	Label for the x-axis. Default is "AUC in Control".
labely	Label for the y-axis. Default is "AUC in Stress".
axismin_x	Minimum value for the x-axis. Default is -10.
axismax_x	Maximum value for the x-axis. Default is 100.
axismin_y	Minimum value for the y-axis. Default is -10.
axismax_y	Maximum value for the y-axis. Default is 100.
maintitle	Main title of the plot. Default is an empty string.
subtitle	Subtitle of the plot. Default is an empty string.
legendpos	Position of the legend. Default is "bottom".
formatname	Format of the saved plot (e.g., "pdf", "png"). Default is "pdf".
outfold	Output folder where the plot will be saved. Default is tempdir().
outfile	Name of the output file. Default is "AUCcompare_pval".
plottype	Type of plot to generate. Can be "pval" for p-value based plots or "groups" for group-based plots. Default is "pval".
plot	A logical flag indicating whether to display the plot interactively (TRUE) or save it to a file (FALSE). Default is FALSE.
universename	Column name in tab representing the universe group in group-based plots. Default is "Universe".

groupname	Column name in tab representing specific groups in group-based plots. Default is "Group".
verbose	A logical flag indicating whether to display detailed messages about the function's progress. Default is TRUE.

Details

The function supports two plot types:

- "pval": The plot highlights genes based on adjusted FDR p-values and can highlight specific genes provided in genevec.
- "groups": The plot highlights predefined groups, such as "Attenuated" and "Outgroup", within the data.

If plot = TRUE, the plot is displayed interactively. If plot = FALSE, the plot is saved to a file in the specified format and output folder.

Value

A plot comparing AUC values between control and stress conditions, either displayed or saved to a file.

See Also

[allauc]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Calculating necessary results
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
avfilt <- averageandfilterexprs(expdf, transdf, expthres,
       showtime = FALSE, verbose = FALSE)
rescountna <- countna(avfilt, expdf, nbcpu = 1, verbose = FALSE)
ecdf <- genesECDF(avfilt, verbose = FALSE)
resecdf <- ecdf[[1]]
nbwindows <- ecdf[[2]]
resmeandiff <- meandifference(resecdf, expdf, nbwindows,
   verbose = FALSE)
bytranslistmean <- split(resmeandiff, factor(resmeandiff$transcript))
resknee <- kneeid(bytranslistmean, expdf, verbose = FALSE)
resauc <- allauc(bytranslistmean, expdf, nbwindows, verbose = FALSE)
resatt <- attenuation(resauc, resknee, rescountna, bytranslistmean, expdf,
       resmeandiff, verbose = FALSE)
resug <- universegroup(resatt, expdf, verbose = FALSE)

## Testing plotauc
```

```
plotauc(resug, expdf, plottype = "groups", plot = TRUE)
```

---

plotecdf *Plot Empirical Cumulative Distribution Function (ECDF)*

Description

This function generates an ECDF plot to analyze transcription density relative to the distance from the transcription start site (TSS) across different conditions. The plot displays AUC values, Kolmogorov-Smirnov (KS) statistics, and knee points, with options to display or save the plot.

Usage

```
plotecdf(dfmeandiff, unigroupdf, expdf, genename,
   colvec = c("#90AFBB", "#10AFBB", "#FF9A04", "#FC4E07"),
   outfold = tempdir(), digits = 2, middlewind = 100, pval = 0.01,
   plot = FALSE, formatname = "pdf", verbose = TRUE)
```

Arguments

dfmeandiff	A data frame containing the mean differences of transcription levels and cumulative distribution values (Fx) for different windows around the TSS (see meandifference).
unigroupdf	A data frame containing gene-specific statistics, including their belonging to Universe or Group (see universegroup).
expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
genename	A string specifying the name of the gene of interest to plot.
colvec	A vector of colors used to distinguish different conditions in the plot. Default is `c("#90AFBB", "#10AFBB", "#FF9A04", "#FC4E07")`.
outfold	A string specifying the output folder where the plot will be saved if `plot = FALSE`. Default is `tempdir()`.
digits	The number of decimal places to round the AUC and KS values. Default is `2`.
middlewind	The index of the middle window representing the region centered around the TSS. Default is `100`.
pval	A numeric value for the p-value threshold to determine the significance of the KS test. Default is `0.01`.
plot	A logical flag indicating whether to display the plot interactively (`TRUE`) or save it to a file (`FALSE`). Default is `FALSE`.
formatname	String of the format of the saved plot. Possible values are "eps", "ps", "tex" (pictex), "pdf", "jpeg", "tiff", "png", "bmp", and "svg". Default is `"pdf"`.
verbose	A logical flag indicating whether to display detailed messages about the function's progress. Default is `TRUE`.

Details

The function processes data related to transcription levels and cumulative transcription density for a given gene across multiple experimental conditions. The ECDF plot is constructed with optional annotation of key statistics such as AUC values and significant KS test results. Knee points, representing significant changes in transcription density, are also displayed if the KS test passes the specified p-value threshold.

Colvec: The number of colors should be equal to the number of rows of expdf divided by two (a forward and reverse files are provided for each experiment).

Value

An ECDF plot showing the transcription density across windows around the TSS, with highlights for significant KS test results and knee points. The plot can either be displayed or saved as a file.

See Also

[meandifference], [universegroup]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Calculating necessary results
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
avfilt <- averageandfilterexprs(expdf, transdf, expthres,
       showtime = FALSE, verbose = FALSE)
rescountna <- countna(avfilt, expdf, nbcpu = 1, verbose = FALSE)
ecdf <- genesECDF(avfilt, verbose = FALSE)
resecdf <- ecdf[[1]]
nbwindows <- ecdf[[2]]
resmeandiff <- meandifference(resecdf, expdf, nbwindows,
    verbose = FALSE)
bytranslistmean <- split(resmeandiff, factor(resmeandiff$transcript))
resknee <- kneeid(bytranslistmean, expdf, verbose = FALSE)
resauc <- allauc(bytranslistmean, expdf, nbwindows, verbose = FALSE)
resatt <- attenuation(resauc, resknee, rescountna, bytranslistmean, expdf,
        resmeandiff, verbose = FALSE)
resug <- universegroup(resatt, expdf, verbose = FALSE)

## Testing plotecdf
colvec <- c("#90AFBB", "#10AFBB", "#FF9A04", "#FC4E07")
plotecdf(resmeandiff, resug, expdf, "EGFR", colvec, plot = TRUE, verbose = FALSE)
```

plothistoknee *Plot Histogram of Distance from TSS to Knee Point*

Description

This function generates a histogram showing the distribution of the distance from the transcription start site (TSS) to the knee point for attenuated genes. The distance can be plotted either as a percentage of the gene length or in kilobases (kb).

Usage

```
plothistoknee(unigroupdf, plottype = "percent", xlimvec = NA,
binwidthval = NA, kneename = "knee_AUC_HS", plot = FALSE, outfold = tempdir(),
formatname = "pdf", universename = "Universe", groupname = "Group",
verbose = TRUE)
```

Arguments

unigroupdf	A data frame containing gene-level statistics, including knee point data and group classification (see universegroup).
plottype	A string specifying the type of distance to plot. Options are "percent" for the percentage of the gene or "kb" for distance in kilobases. Default is "percent".
xlimvec	A numeric vector of length 2 specifying the limits of the x-axis. Default is NA, which automatically sets the limits based on plottype.
binwidthval	A numeric value for the width of the bins in the histogram. Default is NA, which automatically selects a bin width based on plottype.
kneename	A string specifying the name of the column in unigroupdf that contains the knee point data. Default is "knee_AUC_HS".
plot	A logical flag indicating whether to display the plot interactively (TRUE) or save it to a file (FALSE). Default is FALSE.
outfold	A string specifying the output folder where the plot will be saved if plot = FALSE. Default is the current directory.
formatname	A string specifying the format of the saved plot file. Default is "pdf".
universename	A string specifying the name of the column in unigroupdf that defines the universe of genes. Default is "Universe".
groupname	A string specifying the name of the column in unigroupdf that defines the group classification of genes. Default is "Group".
verbose	A logical flag indicating whether to display detailed messages about the function's progress. Default is TRUE.

Value

A histogram showing the distribution of the distance from the TSS to the knee point for attenuated genes. The plot can either be displayed interactively or saved to a file.

See Also

[universegroup]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Calculating necessary results
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
avfilt <- averageandfilterexprs(expdf, transdf, expthres,
        showtime = FALSE, verbose = FALSE)
rescountna <- countna(avfilt, expdf, nbcpu = 1, verbose = FALSE)
ecdf <- genesECDF(avfilt, verbose = FALSE)
resecdf <- ecdf[[1]]
nbwindows <- ecdf[[2]]
resmeandiff <- meandifference(resecdf, expdf, nbwindows,
    verbose = FALSE)
bytranslistmean <- split(resmeandiff, factor(resmeandiff$transcript))
resknee <- kneeid(bytranslistmean, expdf, verbose = FALSE)
resauc <- allauc(bytranslistmean, expdf, nbwindows, verbose = FALSE)
resatt <- attenuation(resauc, resknee, rescountna, bytranslistmean, expdf,
        resmeandiff, verbose = FALSE)
resug <- universegroup(resatt, expdf, verbose = FALSE)

## Testing plothistoknee
plothistoknee(resug, plot = TRUE)
```

---

plotmetagenes *Plot Metagenes for Gene Groups*

Description

This function plots metagene profiles based on transcript data, comparing transcription density across conditions (e.g., control vs. stress). The function allows the user to plot metagenes for different gene groups such as attenuated genes, outgroup genes, the entire universe of genes, or all genes.

Usage

```
plotmetagenes(unigroupdf, dfmeandiff, expdf, plottype = "attenuation",
daucname = "dAUC_Diff_meanFx_HS_ctrl", auc_ctrlname = "AUC_ctrl",
auc_stressname = "AUC_HS", plot = FALSE, formatname = "pdf",
outfold = tempdir(), verbose = TRUE)
```

Arguments

unigroupdf	A data frame containing gene-level information, including group classifications and dAUC data for different conditions (see universegroup).
dfmeandiff	A data frame containing mean transcription values and coordinates for each transcript (see meandifference).
expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
plottype	A string specifying the group of genes to plot. Options are "attenuation", "outgroup", "universe", or "all". Default is "attenuation".
daucname	A string specifying the column name for the delta AUC value (difference between conditions). Default is "dAUC_Diff_meanFx_HS_ctrl".
auc_ctrlname	A string specifying the column name for the control condition AUC values. Default is "AUC_ctrl".
auc_stressname	A string specifying the column name for the stress condition AUC values. Default is "AUC_HS".
plot	A logical flag indicating whether to display the plot interactively (TRUE) or save it to a file (FALSE). Default is FALSE.
formatname	A string specifying the format of the saved plot file. Default is "pdf".
outfold	A string specifying the output folder where the plot will be saved if plot = FALSE. Default is the current directory.
verbose	A logical flag indicating whether to display detailed messages about the function's progress. Default is TRUE.

Details

This function summarizes mean transcription levels across genomic coordinates for different gene groups and plots the transcription density from the transcription start site (TSS) to the transcription termination site (TTS). The function can generate metagene plots for different gene groups such as attenuated, outgroup, or all genes, and compares transcription profiles between conditions (e.g., control vs. stress). The resulting plot helps visualize differences in transcriptional response between groups of genes under different conditions.

Value

A metagene plot comparing transcription density across conditions (e.g., control vs. stress) for the selected group of genes. The plot can either be displayed interactively or saved to a file.

See Also

[universegroup], [meandifference]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1
```

```
## Calculating necessary results
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
avfilt <- averageandfilterexprs(expdf, transdf, expthres,
        showtime = FALSE, verbose = FALSE)
rescountna <- countna(avfilt, expdf, nbcpu = 1, verbose = FALSE)
ecdf <- genesECDF(avfilt, verbose = FALSE)
resecdf <- ecdf[[1]]
nbwindows <- ecdf[[2]]
resmeandiff <- meandifference(resecdf, expdf, nbwindows,
    verbose = FALSE)
bytranslistmean <- split(resmeandiff, factor(resmeandiff$transcript))
resknee <- kneeid(bytranslistmean, expdf, verbose = FALSE)
resauc <- allauc(bytranslistmean, expdf, nbwindows, verbose = FALSE)
resatt <- attenuation(resauc, resknee, rescountna, bytranslistmean, expdf,
        resmeandiff, verbose = FALSE)
resug <- universegroup(resatt, expdf, verbose = FALSE)

## Testing plotmetagenes
plotmetagenes(resug, resmeandiff, expdf, plottype = "attenuation", plot = TRUE)
```

`plotmulti` *Generate all tepr plots for all experiment comparisons*

Description

This function generates for all experiment comparisons contained in the object `resteprmulti` all plots of tepr: ECDF, auc, metagene, and histtoknee.

Usage

```
plotmulti(resteprmulti, expdf, ecdfgenevec, outfold = tempdir(), digits = 2,
middlewind = 100, pval = 0.01, colvec = c("#90AFBB", "#10AFBB",
"#FF9A04", "#FC4E07"), genaucvec = NA, aucaxisminx = -10,
aucaxismaxx = 100, aucaxisminy = -10, aucaxismaxy = 100, aucmaintitle = "",
aucsubtitle = "", auclegendpos = "bottom", formatname = "pdf",
uniname = "Universe", groupname = "Group", histkneexlim = NA,
binwidthvalhistknee = NA, verbose = TRUE)
```

Arguments

`resteprmulti`	Result returned by the function `teprmulti`.
`expdf`	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
`ecdfgenevec`	A vector specifying the names of the genes of interest to plot the ecdf of.

outfold	Path to the output folder where the plots will be written. Subfolders with the names of the comparisons are automatically created. Default is tempdir().
digits	For the ecdf plot, the number of decimal places to round the AUC and KS values. Default is 2.
middlewind	For the ecdf plot, the index of the middle window representing the region centered around the TSS. Default is 100.
pval	For the ecdf plot, a numeric value for the p-value threshold to determine the significance of the KS test. Default is 0.01.
colvec	For the ecdf plot, a vector of 4 colors used to distinguish the different conditions. Default is c("#90AFBB", "#10AFBB","#FF9A04", "#FC4E07").
genaucvec	For the auc plot, vector of gene names to highlight, Used for the plot of type "pval". Default is NA. If left to NA, the plot taking into account the p-values is not generated.
aucaxisminx	For the auc plot, minimum value for the x-axis. Default is -10.
aucaxismaxx	For the auc plot, maximum value for the x-axis. Default is 100.
aucaxisminy	For the auc plot, minimum value for the y-axis. Default is -10.
aucaxismaxy	For the auc plot, maximum value for the y-axis. Default is 100.
aucmaintitle	For the auc plot, main title of the plot. Default is an empty string.
aucsubtitle	For the auc plot, subtitle of the plot. Default is an empty string.
auclegendpos	For the auc plot, position of the legend. Default is "bottom".
formatname	Format of the saved plot (e.g., "pdf", "png"). Default is "pdf".
uniname	Column name in the second element of resteprmulti representing the universe selection. Default is "Universe".
groupname	Column name in the second element of resteprmulti representing the type of group a transcript belong to. Default is "Group".
histkneexlim	For the plot histoknee, a numeric vector of length 2 specifying the limits of the x-axis. Default is NA, which automatically sets the limits based on plottype.
binwidthvalhistknee	For the plot histoknee, a numeric value for the width of the bins in the histogram. Default is NA, which automatically selects a bin width based on plottype.
verbose	A logical flag indicating whether to display detailed messages about the function's progress. Default is TRUE.

Details

The function goes through each element of resteprmulti which corresponds to a comparison of two conditions. For each element it calls the following functions: #'

- "plotecdf": The function generates a figure for each gene given in ecdfgenevec.
- "plotauc": Generates figures by groups and pval. The lattest figure is not generated if genaucvec = NA.
- "plotmetagenes": Generates the figures by attenuation, outgroup, universe, and all.
- "plothistoknee": Generate the figures by percent and kb.

Value

Nothing is returned. Figures are written to outfold in the subfolder of the corresponding comparison.

See Also

[teprmulti], [plotecdf], [plotauc], [plotmetagenes], [plothistoknee]

Examples

```
# Assuming resteprmulti is the object returned by the function teprmulti
# and expdf contains the necessary data:
## Not run:
plotmulti(resteprmulti, expdf, ecdfgenevec = c("EGFR", "DAP", "FLI1"))
## End(Not run)
```

`preprocessing` *Preprocess Experimental Data for Genomic Analysis*

Description

This function orchestrates a pipeline for preprocessing genomic data, including filtering annotations, splitting transcripts into windows, retrieving bedgraph values, and generating a final annotated table.

Usage

```
preprocessing(exptabpath, gencodepath, windsize, maptrackpath,
blacklistpath, genomename = NA, nbcputrans = 1, finaltabpath = tempdir(),
finaltabname = "anno.tsv", tmpfold = file.path(tempdir(), "tmptepr"),
saveobjectpath = tempdir(), savefinaltable = TRUE, reload = FALSE,
showtime = FALSE, showmemory = FALSE, deletetmp = TRUE, chromtab = NA,
forcechrom = FALSE, verbose = TRUE)
```

Arguments

`exptabpath`	Character. Path to the experiment table file.
`gencodepath`	Character. Path to the Gencode annotation file.
`windsize`	Integer. Window size for splitting transcripts.
`maptrackpath`	Character. Path to the mappability track file.
`blacklistpath`	Character. Path to the blacklist file.
`genomename`	Character. Name of the genome assembly (e.g., "hg38"). Default is `NA`. If left to NA, chromtab should be provided.
`nbcputrans`	Integer. Number of CPUs to use for transcript processing. Default is `1`.
`finaltabpath`	Character. Path where the final annotated table will be saved. Default is `tempdir()`.

finaltabname	Character. Name of the final annotated table file. Default is anno.tsv.
tmpfold	Character. Path to a temporary folder for intermediate files. Default is file.path(tempdir(), "tmptepr").
saveobjectpath	Character. Path to save intermediate objects. Default is tempdir(). If set to NA, intermediate objects will not be saved.
savefinaltable	Logical. Whether to save the final table to disk. Default is TRUE.
reload	Logical. Whether to reload intermediate objects if available. Default is FALSE.
showtime	Logical. Whether to display timing information. Default is FALSE.
showmemory	Logical. Whether to display memory usage information. Default is FALSE.
deletetmp	Logical. Whether to delete temporary files after processing. Default is TRUE.
chromtab	A Seqinfo object retrieved with the rtracklayer method SeqinfoForUCSCGenome. If NA, the method is called automatically and the genomename should be provided. Default is NA.
forcechrom	Logical indicating if the presence of non-canonical chromosomes in chromtab (if not NA) should trigger an error. Default is FALSE.
verbose	Logical. Whether to display detailed progress messages. Default is TRUE.

Details

The ‘preprocessing‘ function performs several key tasks: 1. Filters Gencode annotations to retrieve "transcript" annotations. 2. Differentiates between protein-coding (MANE_Select) and long non-coding (lncRNA, Ensembl_canonical) transcripts. 3. Splits transcripts into windows of size ‘windsize‘. 4. Processes bedgraph files to retrieve values, exclude blacklisted regions, and retain high-mappability intervals. 5. Generates a final annotated table with scores derived from the above steps.

Temporary files created during processing are optionally deleted at the end.

Value

A data frame representing the final table containing transcript information and scores on 'windsize' windows for all experiments defined in the experiment table (exptabpath).

See Also

[retrieveanno], [makewindows], [blacklisthighmap], [createtablescores]

Examples

```
## Data
exptabpath <- system.file("extdata", "exptab-preprocessing.csv", package = "tepr")
gencodepath <- system.file("extdata", "gencode-chr13.gtf", package = "tepr")
maptrackpath <- system.file("extdata", "k50.umap.chr13.hg38.0.8.bed",
  package = "tepr")
blacklistpath <- system.file("extdata", "hg38-blacklist-chr13.v2.bed",
    package = "tepr")
windsize <- 200
genomename <- "hg38"
```

```
## Copying bedgraphs to the current directory
expdfpre <- read.csv(exptabpath)
bgpathvec <- sapply(expdfpre$path, function(x) system.file("extdata", x,
    package = "tepr"))
expdfpre$path <- bgpathvec
write.csv(expdfpre, file = "exptab-preprocessing.csv", row.names = FALSE,
    quote = FALSE)
exptabpath <- "exptab-preprocessing.csv"

## Testing preprocessing
finaltabtest <- preprocessing(exptabpath, gencodepath, windsize, maptrackpath,
    blacklistpath, genomename = genomename, verbose = FALSE)
```

`retrieveanno` *Retrieve and Combine Annotation Information*

Description

This function filters gencode annotations to retrieve "transcript". It then distinguishes transcripts coming from protein coding genes (MANE_Select) and those coming from long non-coding genes (lncRNA, Ensembl_canonical).

Usage

```
retrieveanno(exptabpath, gencodepath, saveobjectpath = NA, showtime = FALSE,
verbose = TRUE)
```

Arguments

`exptabpath`	Path to the experiment table file containing a table with columns named 'condition', 'replicate', 'strand', and 'path'.
`gencodepath`	Path to the GENCODE annotation file.
`saveobjectpath`	Path to save intermediate R objects. Default is 'NA' and R objects are not saved.
`showtime`	Logical. If 'TRUE', displays timing information. Default is 'FALSE'.
`verbose`	Logical. If 'TRUE', provides detailed messages during execution. Default is 'TRUE'.

Details

The function performs the following steps: 1. Reads experimental data from the provided CSV file and validates it. 2. Reads genomic annotations from the gencode file and filters for transcripts. 3. Separately processes protein-coding and long non-coding RNA transcripts: - For protein-coding genes, selects the most representative (MANE_Select or Ensembl_canonical) transcripts. - For long non-coding RNAs, filters out transcripts with undesirable evidence levels. 4. Combines these annotations into a single data frame, labeling each transcript with its biotype. 5. Optionally saves the resulting data frame as an RDS file in the specified directory. 6. Optionally reports the total time taken for analysis.

Value

A data frame containing the combined annotation information for protein-coding and long non-coding RNA transcripts. If ‘saveobjectpath‘ is not ‘NA‘, the object is also saved as an RDS file in the specified directory.

Examples

```
exptabpath <- system.file("extdata", "exptab-preprocessing.csv", package="tepr")
gencodepath <- system.file("extdata", "gencode-chr13.gtf", package = "tepr")

## Testing retrieveanno
allannobed <- retrieveanno(exptabpath, gencodepath, verbose = FALSE)
```

`retrievechrom` *Retrieve chromosome lengths and information for a specified genome.*

Description

This function connects to the UCSC Genome Browser database using the ‘rtracklayer‘ package to retrieve chromosome information. It returns a ‘Seqinfo‘ object, filtering out unwanted chromosomes such as mitochondrial DNA (‘chrM‘) and those with alternative contigs (indicated by an underscore ‘_‘).

Usage

```
retrievechrom(genomename, verbose, filterchrom = TRUE)
```

Arguments

`genomename`	A character string specifying the UCSC genome name (e.g., "hg19" or "mm10").
`verbose`	A logical value. If ‘TRUE‘, the function will print messages during execution, including a list of the chromosomes being kept.
`filterchrom`	A logical value. If ‘TRUE‘, mitochondrial and non-canonical chromosomes are removed. Default is `TRUE`.

Value

A ‘Seqinfo‘ object containing the names and lengths of the main chromosomes for the specified genome.

Examples

```
# This example requires an internet connection to the UCSC database
hg19_chroms <- retrievechrom(genomename = "hg19", verbose = TRUE)
hg19_chroms
```

showallcomp *Retrieve all the comparison names from the experiment table*

Description

The 'showallcomp' function build the string of all comparisons possible using the condition column of a provided experiment table.

Usage

```
showallcomp(expdf, verbose = FALSE)
```

Arguments

expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
verbose	A logical flag indicating whether to print progress messages. Defaults to FALSE.

Value

If less than three conditions, nothing. Otherwise a character vector of all comparisons.

Examples

```
# Create a valid experiment table
exptab <- data.frame(
  condition = c("cond1", "cond2", "cond3"),
  replicate = c(1, 1, 1),
  direction = c("forward", "reverse", "forward"),
  strand = c("plus", "minus", "plus"),
  path = c("toto/", "toto/", "toto/"))
checkexptab(exptab)
showallcomp(exptab)
```

tepr *Perform the tepr differential nascent rna-seq analysis*

Description

This function wraps the different steps of tepr to identify transcripts with a significantly different nascent rna-seq signal.

Usage

```
tepr(expdf, alldf, expthres, nbcpu = 1, rounding = 10,
controlcondname = "ctrl", stresscondname = "HS", replaceval = NA, pval = 0.1,
significant = FALSE, windsizethres = 50, countnathres = 20,
meanctrlthres = 0.5, meanstressthres = 0.5, pvaltheorythres = 0.1,
aucctrlthreshigher = -10, aucctrlthreslower = 15, aucstressthres = 15,
attenuatedpvalksthres = 2, outgrouppvalksthres = 0.2, showtime = FALSE,
verbose = TRUE)
```

Arguments

expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
alldf	A data frame containing all transcript-related information, including biotype, chromosome, coordinates, transcript, gene, strand, window, ID and scores retrieved from the bedgraph files.
expthres	A numeric value specifying the expression threshold. Transcripts with average expression values below this threshold will be filtered out from the returned transcript vector.
nbcpu	An integer specifying the number of CPU cores to use for parallel computation on transcripts. The number of transcripts is equal to the number of lines provided as input of 'averageandfilterexprs'. Defaults to 1.
rounding	An integer specifying the rounding factor for computing ECDF. Default is 10.
controlcondname	A string specifying the name of the control condition Defaults to "ctrl".
stresscondname	A string specifying the name of the stress condition. Defaults to "HS".
replaceval	A value to replace non-significant attenuation values Defaults to NA.
pval	A numeric value specifying the p-value threshold for significance of the KS test. Defaults to 0.1.
significant	A logical indicating whether to filter out non-significant attenuation values. Defaults to FALSE.
windsizethres	A numeric threshold for the minimum window size. Default is 50.
countnathres	A numeric threshold for the maximum number of missing data points for an experiment (NA values). Default is 20.
meanctrlthres	A numeric threshold for the minimum mean transcription value in the control condition. Default is 0.5.
meanstressthres	A numeric threshold for the minimum mean transcription value in the stress condition. Default is 0.5.
pvaltheorythres	A numeric threshold for the minimum p-value used to define the universe of genes. Default is 0.1.
aucctrlthreshigher	A numeric threshold for the lower bound of the control AUC value in the outgroup classification. Default is -10.

`aucctrlthreslower`
: A numeric threshold for the upper bound of the control AUC value in the outgroup classification. Default is 15.

`aucstressthres`
: A numeric threshold for the minimum stress AUC value used to classify attenuated genes. Default is 15.

`attenuatedpvalksthres`
: A numeric threshold for the negative log10 of the p-value (from KS test) for defining attenuated genes. Default is 2.

`outgrouppvalksthres`
: A numeric threshold for the maximum KS p-value used to define the outgroup. Default is 0.2.

`showtime`
: A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to `FALSE`.

`verbose`
: A logical flag indicating whether to print progress messages. Defaults to `TRUE`.

Details

The tepr function calls successively:

- averageandfilterexprs This function calculates the average expression levels for transcripts from alldf that was obtained with the 'preprocessing' function. It filters out transcripts based on the 'expthres' expression threshold. The function also renames the columns in the output data frame to include mean expression values. It returns a list containing the original alldf with the mean columns added and a character vector of transcripts that meet the filtering criteria.
- countna It takes the result of 'averageandfilterexprs' as input and counts the number of NA values for each transcript based on strand and condition. NA represent missing scores that were filtered out from the black list and mappability track. It returns a data frame where each row corresponds to a transcript, along with its associated gene, strand, and the count of NA values.
- genesECDF It takes the result of 'averageandfilterexprs' as input and a) filters the main expression table to retain only the expressed transcripts; b) Splits the data by each transcript; c) For each transcript, computes ECDF values for the score columns while respecting the strand orientation ("plus" or "minus"); d) Combines the ECDF results into a final data frame. It returns a list containing two elements: A data frame with ECDF results for each transcript (concatdf) and an integer indicating the number of rows in each transcript table.
- meandifference It takes the result of 'genesECDF' as input and calculates the mean values, mean Fx (ECDF) and ECDF differences (Fx) for expression data, across different experimental conditions. It returns a data frame that contains, for each condition: mean values for the "value" and "Fx" columns (e.g., `mean_value_ctrl`, `mean_Fx_ctrl`) and the differences between the `Fx` column and coordinate ratios (e.g., `diff_Fx_ctrl`).
- Splitting Split the results of 'meandifference' by transcripts and stores the list into bytranslistmean.
- allauc It uses the previously computed variable 'bytranslistmean' and it computes the Area Under Curve (AUC) and the differences of AUC between two conditions for a list of transcript data. It returns a data frame containing the AUC and dAUC results for each transcript, along with associated statistical information.

- kneeid It uses the previously computed variable 'bytranslistmean' and identifies the knee point (i.e., point of maximum change) and the maximum difference in the empirical cumulative distribution function (ECDF) for each transcript, across different experimental conditions. It returns a data frame where each row corresponds to a transcript and contains the coordinates of the knee point and the maximum ECDF difference for each condition.
- attenuation It uses the results of the previous functions and it computes the attenuation values for each window of each transcript. It returns a data frame containing the computed attenuation values along with associated transcript information.
- universegroup Using the table produced by 'attenuation', it categorizes genes into a "Universe" and assigns them into groups such as "Attenuated" or "Outgroup" based on transcription data and thresholds. The universe is defined by thresholds for window size, missing data count, mean transcription levels, and p-values. Genes are further classified into groups based on conditions related to AUC and p-value thresholds. A transcript belongs to "Universe" if (window_size > windsizethres & Count_NA < countnathres & meanctrl > meanctrlthres & meanstress > meanstressthres & pvaltheory > pvaltheorythres). A transcript belongs to the groups: - **Attenuated**: if Universe == TRUE & aucstress > aucstressthres & -log10(pvalks) > attenuatedpvalksthres. - **Outgroup**: if Universe == TRUE & pvalks > outgrouppvalksthres & aucctrl > aucctrlthreshigher & aucctrl < aucctrlthreslower.

Value

A list of data.frame made of two elements. The first data.frame is the result of the function 'meandifference': For each condition, it provides the mean values for the "value" and "Fx" columns (e.g. mean_value_ctrl, mean_Fx_ctrl columns) as the differences between the Fx column and coordinate ratios (e.g., diff_Fx_ctrl column). The second data.frame is the result of the function 'universegroup': It contains columns concerning AUC, knee position, attenuation information, and columns defining the universe and groups (see details).

See Also

[averageandfilterexprs()], [countna()], [genesECDF()], [meandifference()], [allauc()], [kneeid()], [attenuation()], [universegroup()], [preprocessing()]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Reading files
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)

## Testing tepr
res <- tepr(expdf, transdf, expthres, verbose = FALSE)
```

`teprmulti` *Perform tepr differential nascent rna-seq analysis for multiple conditions*

Description

This function performs tepr differential nascent RNA-seq analysis for multiple conditions. It iterates over all pairwise comparisons of conditions within the experiment table and calls the 'tepr' function for each pair.

Usage

```
teprmulti(expdf, alldf, expthres, nbcpu = 1, rounding = 10,
 dontcompare = NULL, replaceval = NA, pval = 0.1, significant = FALSE,
 windsizethres = 50, countnathres = 20, pvaltheorythres = 0.1,
 meancondonethres = 0.5, meancondtwothres = 0.5,
 auccondonethreshigher = -10, auccondonethreslower = 15, auccondtwothres = 15,
 attenuatedpvalksthres = 2, outgrouppvalksthres = 0.2,
 saveobjectpath = NA, reload = FALSE, showtime = FALSE, showmemory = FALSE,
 verbose = TRUE)
```

Arguments

`expdf`	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
`alldf`	A data frame containing all transcript-related information, including biotype, chromosome, coordinates, transcript, gene, strand, window, ID and scores retrieved from the bedgraph files.
`expthres`	A numeric value specifying the expression threshold. Transcripts with average expression values below this threshold will be filtered out from the returned transcript vector.
`nbcpu`	An integer specifying the number of CPU cores to use for parallel computation on transcripts. Defaults to `1`.
`rounding`	An integer specifying the rounding factor for computing ECDF. Default is `10`.
`dontcompare`	An optional vector specifying conditions to exclude from the comparison. It should use condition names from expdf and follow the pattern `cond1_vs_cond2`. Defaults to `NULL`.
`replaceval`	A value to replace non-significant attenuation values. Defaults to `NA`.
`pval`	A numeric value specifying the p-value threshold for significance of the KS test. Defaults to `0.1`.
`significant`	A logical indicating whether to filter out non-significant attenuation values. Defaults to `FALSE`.
`windsizethres`	A numeric threshold for the minimum window size. Default is 50.

countnathres A numeric threshold for the maximum number of missing data points for an experiment (NA values). Default is 20.

pvaltheorythres
A numeric threshold for the minimum p-value used to define the universe of genes. Default is 0.1.

meancondonethres
A numeric threshold for the minimum mean transcription value in the first condition of each comparison. Default is 0.5.

meancondtwothres
A numeric threshold for the minimum mean transcription value in the second condition of each comparison. Default is 0.5.

auccondonethreshigher
A numeric threshold for the lower bound of the first condition AUC value in the outgroup classification. Default is -10.

auccondonethreslower
A numeric threshold for the upper bound of the first condition AUC value in the outgroup classification. Default is 15.

auccondtwothres
A numeric threshold for the minimum second condition AUC value used to classify attenuated genes. Default is 15.

attenuatedpvalksthres
A numeric threshold for the negative log10 of the p-value (from KS test) for defining attenuated genes. Default is 2.

outgrouppvalksthres
A numeric threshold for the maximum KS p-value used to define the outgroup. Default is 0.2.

saveobjectpath A character string specifying the path to save the object of the results for each comparison. The file names are defined with the 'condition' column of expdf. Defaults to NA.

reload Logical. If 'TRUE', reloads existing saved objects to avoid recomputation. Default is 'FALSE'. If the function failed during object saving, make sure to delete the corresponding object.

showtime A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to FALSE.

showmemory A logical value indicating if memory usage should be printed during the execution. Defaults to FALSE.

verbose A logical flag indicating whether to print progress messages. Defaults to TRUE.

Value

A list of lists. Each inner list corresponds to a pairwise comparison and contains two data frames: - 'resmeandiff_<comparison>': Results from the 'meandifference' function for the specific comparison. - 'resunigroupatt_<comparison>': Results from the 'universegroup' function for the specific comparison.

See Also

[tepr]

Examples

```
## Supposing the data have more than one condition
## Not run:
  exptabpath <- "exp.csv"
  alldfpath <- "result-preprocessing.tsv"
  expdf <- read.csv(exptabpath)
  alldf <- read.delim(alldfpath, header = FALSE)
  expthres <- 0.1
  reslist <- teprmulti(expdf, alldf, expthres)
## End(Not run)
```

---

universegroup *Define Universe and Group of Genes Based on Expression Data*

Description

This function categorizes genes into a "Universe" and assigns them into groups such as "Attenuated" or "Outgroup" based on transcription data and thresholds. The universe is defined by thresholds for window size, missing data count, mean transcription levels, and p-values. Genes are further classified into groups based on conditions related to AUC and p-value thresholds.

Usage

```
universegroup(completedf, expdf, controlname = "ctrl", stressname = "HS",
windsizethres = 50, countnathres = 20, meanctrlthres = 0.5,
meanstressthres = 0.5, pvaltheorythres = 0.1, aucctrlthreshigher = -10,
aucctrlthreslower = 15, aucstressthres = 15, attenuatedpvalksthres = 2,
outgrouppvalksthres = 0.2, showtime = FALSE, verbose = TRUE)
```

Arguments

completedf	A data frame obtained with the function attenuation.
expdf	A data frame containing experiment data that should have columns named 'condition', 'replicate', 'strand', and 'path'.
controlname	A string representing the control condition name. Default is "ctrl".
stressname	A string representing the stress condition name. Default is "HS".
windsizethres	A numeric threshold for the minimum window size. Default is 50.
countnathres	A numeric threshold for the maximum number of missing data points (NA values). Default is 20.
meanctrlthres	A numeric threshold for the minimum mean transcription value in the control condition. Default is 0.5.

`meanstressthres`
: A numeric threshold for the minimum mean transcription value in the stress condition. Default is 0.5.

`pvaltheorythres`
: A numeric threshold for the minimum p-value used to define the universe of genes. Default is 0.1.

`aucctrlthreshigher`
: A numeric threshold for the lower bound of the control AUC value in the outgroup classification. Default is -10.

`aucctrlthreslower`
: A numeric threshold for the upper bound of the control AUC value in the outgroup classification. Default is 15.

`aucstressthres`
: A numeric threshold for the minimum stress AUC value used to classify attenuated genes. Default is 15.

`attenuatedpvalksthres`
: A numeric threshold for the negative log10 of the p-value (from KS test) for defining attenuated genes. Default is 2.

`outgrouppvalksthres`
: A numeric threshold for the maximum KS p-value used to define the outgroup. Default is 0.2.

`showtime`
: A logical value indicating if the duration of the function processing should be indicated before ending. Defaults to `FALSE`.

`verbose`
: A logical flag indicating whether to print progress messages. Defaults to `TRUE`.

Details

A transcript belongs to "Universe" if: window_size > windsizethres & Count_NA < countnathres & meanctrl > meanctrlthres & meanstress > meanstressthres & pvaltheory > pvaltheorythres

If only one condition is provided, a transcript belongs to "Universe" if: window_size > windsizethres & Count_NA < countnathres & meanctrl > meanctrlthres & pvaltheory > pvaltheorythres

A transcript belongs to the groups: - **Attenuated**: if Universe == TRUE & aucstress > aucstressthres & -log10(pvalks) > attenuatedpvalksthres - **Outgroup**: if Universe == TRUE & pvalks > outgrouppvalksthres & aucctrl > aucctrlthreshigher & aucctrl < aucctrlthreslower

If only one condition is provided: - **Attenuated**: if Universe == TRUE & aucctrl > aucctrlthreslower - **Outgroup**: if Universe == TRUE & aucctrl > aucctrlthreshigher & aucctrl < aucctrlthreslower

This function is useful for classifying genes in transcriptomics data based on their transcriptional response to different experimental conditions.

Value

A modified data frame with two additional columns: `Universe`, indicating whether each gene is part of the universe, and `Group`, classifying the genes into groups such as "Attenuated", "Outgroup", or `NA`.

See Also

[attenuation]

Examples

```
exppath <-  system.file("extdata", "exptab.csv", package="tepr")
transpath <- system.file("extdata", "cugusi_6.tsv", package="tepr")
expthres <- 0.1

## Calculating necessary results
expdf <- read.csv(exppath)
transdf <- read.delim(transpath, header = FALSE)
avfilt <- averageandfilterexprs(expdf, transdf, expthres,
       showtime = FALSE, verbose = FALSE)
rescountna <- countna(avfilt, expdf, nbcpu = 1, verbose = FALSE)
ecdf <- genesECDF(avfilt, verbose = FALSE)
resecdf <- ecdf[[1]]
nbwindows <- ecdf[[2]]
resmeandiff <- meandifference(resecdf, expdf, nbwindows,
    verbose = FALSE)
bytranslistmean <- split(resmeandiff, factor(resmeandiff$transcript))
resknee <- kneeid(bytranslistmean, expdf, verbose = FALSE)
resauc <- allauc(bytranslistmean, expdf, nbwindows, verbose = FALSE)
resatt <- attenuation(resauc, resknee, rescountna, bytranslistmean, expdf,
        resmeandiff, verbose = FALSE)
## Testing universegroup
resug <- universegroup(resatt, expdf, verbose = FALSE)
```

Index

allauc, 2
attenuation, 4
averageandfilterexprs, 6

blacklisthighmap, 7

checkexptab, 9
countna, 10
createtablescores, 11

genesECDF, 13

kneeallconds, 14
kneeid, 16

makewindows, 17
meandifference, 18

plotauc, 19
plotecdf, 22
plothistoknee, 24
plotmetagenes, 25
plotmulti, 27
preprocessing, 29

retrieveanno, 31
retrievechrom, 32

showallcomp, 33

tepr, 33
teprmulti, 37

universegroup, 39