Package 'TCseq'

April 16, 2019

Type Package

Title Time course sequencing data analysis

Version 1.6.1

Author Mengjun Wu <minervajunjun@gmail.com>, Lei Gu <leigu@broadinstitute.org>

Maintainer Mengjun Wu <minervajunjun@gmail.com>

Description Quantitative and differential analysis of epigenomic and transcriptomic time course sequencing data, clustering analysis and visualization of temporal patterns of time course data.

Depends R (>= 3.4)

License GPL (>= 2)

LazyData TRUE

Imports edgeR, BiocGenerics, reshape2, GenomicRanges, IRanges, SummarizedExperiment, GenomicAlignments, Rsamtools, e1071, cluster, ggplot2, grid, grDevices, stats, utils, methods, locfit

Suggests testthat

biocViews ImmunoOncology, Epigenetics, TimeCourse, Sequencing, ChIPSeq, RNASeq, DifferentialExpression, Clustering, Visualization

RoxygenNote 5.0.1

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

git_branch RELEASE_3_8

git_last_commit bd2c43e

git_last_commit_date 2019-01-04

Date/Publication 2019-04-15

R topics documented:

`clust-class` *clust class*

Description

`clust` is a S4 class for storing results of a clustering analysis for time course data.

Details

: The clust objects are returned from `timeclust` and have a show method printing a compact summary of their contents

Slots

Oject of this class contains the following slots:

`method` clustering method that has been used

`dist` distance method that has been used

`data` a matrix of original or standardized data that has been used for the analysis

`centers` a matrix of class centers

`cluster` an integer vector of length n (n is the number of data points each integer indicates the cluster a data point belongs to. For the fuzzy cmeans clustering method, a data point is assigned to the closest cluster to which the data point has highest membership value.

`membership` a matrix with membership values of the data points to all the clusters

Author(s)

Mengjun Wu

See Also

`timeclust`, `@`

clust.accessors *Accessors to extract slots of a clust class.*

Description

Accessors are provided to extract data, centers, cluster, membership, membership slots of a clust class.

Usage

```
clustData(object)

## S4 method for signature 'clust'
clustData(object)

clustCenters(object)

## S4 method for signature 'clust'
clustCenters(object)

clustCluster(object)

## S4 method for signature 'clust'
clustCluster(object)

clustMembership(object)

## S4 method for signature 'clust'
clustMembership(object)
```

Arguments

object	clust object object

Value

clustData returns data matrix. clustCenters returns a matrix of centers. clustCluster returns an integer vector. clustMembership returns a matrix of membership, see clust for details.

Author(s)

Mengjun Wu

See Also

clust

countReads *count mapped reads overlap genomic intervals*

Description

This function counts mapped reads from multiple BAM files that overlap genomic intervals in genomicFeature in a TCA object. The counting result is stored in 'count' slot of the TCA object.

Usage

```
countReads(object, dir, method = "summarizeoverlaps", zero.based = TRUE,
  ...)
```

Arguments

object	a TCA object
dir	character string giving the directory where BAM files are stored.
method	character string giving the counting method. Options are 'summarizeOverlaps' and 'featureCounts'. For Windows system, only 'summarizeOverlaps' can be used, For Linux system, both methods can be used.
zero.based	Logical. If TRUE, the start positions of the genomic intervals are *0-based*, if FALSE, the start positions will be *1-based*.
...	additional arguments passed to summarizeOverlaps and featureCounts in Rsubread package

Details

This function provides two options 'summarizeOverlaps' from GenomicAlignments package and featureCounts' from Rsubread package to count the aligned reads. As Rsubread package is only avaible for linux systems, Windows users can only use 'summarizeOverlaps'. The user could specify counting details by passing additional arguments (...), otherwise the default settings of the two methods are used. For counting details, see summarizeOverlaps, featureCounts in Rsubread package

Value

A TCA object with updated 'count' slot.

Author(s)

Mengjun Wu

See Also

summarizeOverlaps, featureCounts in Rsubread package

counts *Extracts counts of a TCA object.*

Description

counts extract raw read counts stored in a TCA object or compute normalized counts.

Usage

```
## S4 method for signature 'TCA'
counts(object, normalization = "none", lib.norm = TRUE,
  log = FALSE, ...)

## S4 replacement method for signature 'TCA'
counts(object) <- value
```

Arguments

object	a TCA object
normalization	character string giving the normalization method. Options are 'none' (original raw counts), 'cpm' (counts per million), 'rpkm' (reads per kilobase per million).
lib.norm	logical indicating whether or not use effective library size (see 'Details' below) when normalization is 'cpm' or 'rpkm'.
log	logical if TRUE, the returned value will be on a log2 scale.
...	additional arguments passed to cpm or rpkm
value	an integer matrix

Details

when calculating normalized counts, library size can be rescaled to minimize the log-fold changes between samples for most genomic features (e.g. genes, binding sites) by multiplying a scale factor. The rescaled library size is called effective library size. In this function, the scale factor is calculated using the weighted trimmed mean of M-values (TMM, Robinson et al (2010))

If log2 values are computed, a small count would be added to avoid logarithm of zero. a small count is set proportional to the library size, the average value of such small counts of all libraries counts is set to 0.25 by default.

Value

An integer matrix

Author(s)

Mengjun Wu

References

Robinson, M. D., & Oshlack, A. (2010). A scaling normalization method for differential expression analysis of RNA-seq data. Genome biology, 11(3), 1.

Examples

```
data(tca_ATAC)
c <- counts(tca_ATAC)
# normalized counts table
c_norm <- counts(tca_ATAC, normalization='rpkm')
```

---

countsTable *An example read Counts table*

Description

A dataset of exemplary read counts

Usage

```
data(countsTable)
```

Format

A data frame containing experiment design information for 12 samples/libraries.

Value

A data frame

Examples

```
data(countsTable)
```

DBanalysis *Perform differential binding analysis*

Description

This function performs differetial analysis by fitting read counts to a negative binomial generalized linear model.

Usage

```
DBanalysis(object, categories = "timepoint", norm.lib = TRUE,
  filter.type = NULL, filter.value = NULL, samplePassfilter = 2, ...)
```

Arguments

object	a TCA object.
categories	character string indicating levels of which factor (column in the design slot) are compared in the differential analysis. For time course analysis, the default factor is timepoint'.
norm.lib	logical indicating whether or not use effective library size when perform normalization. See 'Details' of counts
filter.type	character string indicating which type of count (raw or normalized) is used when doing filtering. Options are 'raw', cpm', 'rpkm', 'NULL'. NULL' means no filtering will be performed.
filter.value	A numberic value; if values of selected filter.type ('raw', cpm', 'rpkm') of a genomic feature are larger than the filter.value in at least a certain number (samplePassfilter) of samples/libraries for any of the conditions, such genomic feature will be kept; otherwise the genomic feature will be dropped.
samplePassfilter	numberic value indicating the least number of samples/libraries a genomic feature with counts (raw or normalized) more than filter.value for all conditions if such genomic feature will be kept.
...	additional arguments passed to glmFit from edgeR package.

Details

The differetial event is detected by using the generalized linear model (GLM) methods (McCarthy et al, 2012). This function fits the read counts of each genes to a negative binomial glms by using glmFit function from edgeR. To further test the significance of changes, see DBresult, TopDBresult

Value

A TCA object

Author(s)

Mengjun Wu, Lei Gu

References

McCarthy,D.J.,Chen, Y., & Smyth, G. K.(2012). Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. Nucleic acids research 40, 4288-4297.

See Also

DBresult, TopDBresult

Examples

```
data(tca_ATAC)
tca_ATAC <- DBanalysis(tca_ATAC)
```

DBresult *This function performs differetial analysis by fitting read Perform likelihood ratio tests and extract the differential analysis results*

Description

This function performs likelihood ratio tests for given coefficinets contrasts after fitting read counts to GLM by DBanalysis. DBresult extracts the diffential analysis results of given contrasts for all genomic features or genomic features with significant differential events. DBresult.cluster returns similar results while the results only contain genomic features belong to a given cluster.

Usage

```
DBresult(object, group1 = NULL, group2 = NULL, contrasts = NULL,
  p.adjust = "fdr", top.sig = FALSE, pvalue = "paj",
  pvalue.threshold = 0.05, abs.fold = 2, direction = "both",
  result.type = "GRangesList")

DBresult.cluster(object, group1 = NULL, group2 = NULL, contrasts = NULL,
  p.adjust = "fdr", top.sig = FALSE, pvalue = "paj",
  pvalue.threshold = 0.05, abs.fold = 2, direction = "both", cluster,
  cmthreshold = NULL, result.type = "GRangesList")
```

Arguments

object	a TCA object, for DBresult, DBanalysis should already be called on the object; for DBresult.cluster, both DBanalysis and timeclust should be already called.
group1	character string giving the level to be compared, that is the denominator in the fold changes.
group2	a character vetor giving other levels to compared with group1. that are numerator in the fold changes.
contrasts	a character vector, each charcter string in the vector gives a contrast of two groups with the format group2vsgroup1', group1 is the denominator level in the fold changes and group2 is the numerator level in the fold changes.
p.adjust	character string specifying a correction method for p-values. Options are 'holm', hochberg', 'hommel', 'bonferroni', BH', 'BY', 'fdr', 'none'.
top.sig	logical if TRUE, only genomic regions with significant differential events will are returned. Significant differential events are defined by log2-fold changes,p-values or adjusted p-values.
pvalue	character string specify the type of p-values used to define significant differential events('PValue' or adjusted p-value 'paj')
pvalue.threshold	a numeric value giving threshold of selected p-value, Significant differential events have lower (ajusted) p-values than the threshold.
abs.fold	a numeric value, the least absolute log2-fold changes

direction	character string specify the direction of fold changes ('up' (positive fold changes), down' (negative fold changes), both'(both positive and negative fold changes)). Significant events have log2-fold changes exceeding abs.fold in defined directions.
result.type	character string giving the data type of return value. Options are "GRangesList" and "list".
cluster	an integer, the result tables of genomic features belong to the cluster are extracted.
cmthreshold	a numeric value, this argument is applicable only if cmeans' clustering method is selected when calling timeclust function. if not NULL, the result table of genomic features that belong to the defined cluster and the membership values to this cluster exceed cmthreshold are extracted.

Details

This function uses glmLRT from edgeR which perform likelihood ratio tests for testing significance of changes. For more deatils, see glmLRT

Value

A list or a GRangesList. If result.type is "GRangesList", a GRangesList is returned containing the differential analysis results for all provided contrasts. Each GRanges object of the list is one contrast, the analysis results are contained in 4 metadata columns:

logFC log2-fold changes of differential event between two tested.

PValue p-values.

paj adjusted p-values

id genomic feature name

If result.type is "list", a List of data frames is returned. Each data frame is one contrast and contains the following columns:

logFC log2-fold changes of differential event between two tested.

PValue p-values.

paj adjusted p-values

chr name of the chromosomes

start starting position of the feature in the chromosome

end ending postition of the feature in the chromosome

id genomic feature name

Note

If not NULL group1, group2 and contrasts, result tables are extracted from comparisons in constrasts.

Author(s)

Mengjun Wu, Lei Gu

See Also

glmLRT

Examples

```
data(tca_ATAC)
tca_ATAC <- DBanalysis(tca_ATAC)
### extract differntial analysis of 24h, 72h to 0h
# set the contrasts using the 'group1' and 'group2' paramters
res1 <- DBresult(tca_ATAC, group1 = '0h', group2 = c('24h', '72h'))
# one can get the same result by setting the contrasts using hte 'contrasts' parameter
res2 <- DBresult(tca_ATAC, contrasts = c('24hvs0h', '72hvs0h'))
# extract significant diffential events
res.sig <- DBresult(tca_ATAC, contrasts = c('24hvs0h', '72hvs0h'),
                    top.sig = TRUE)

# extract differntial analysis of 24h, 72h to 0h of a given cluster
tca_ATAC <- timecourseTable(tca_ATAC, filter = TRUE)
tca_ATAC <- timeclust(tca_ATAC, algo = 'cm', k = 6)
res_cluster1 <- DBresult.cluster(tca_ATAC, group1 = '0h',
                                 group2 = c('24h', '72h'),
                                 cluster = 1)
```

`experiment` *An example experiment design without BAM file infomration*

Description

A dataset of exemplary experiment design without BAM file infomration

Usage

```
data(experiment)
```

Format

A data frame containing experiment design information for 12 samples/libraries.

Value

A data frame

Examples

```
data(experiment)
```

experiment_BAMfile *An example experiment design with BAM file infomration*

Description

A dataset of exemplary experiment design with BAM file infomration

Usage

```
data(experiment_BAMfile)
```

Format

A data frame containing experiment design information for 12 samples/libraries.

Value

A data frame

Examples

```
data(experiment_BAMfile)
```

genomicIntervals *An example reference genomic regions*

Description

A dataset of exemplary genomic regions

Usage

```
data(genomicIntervals)
```

Format

A data frame containing 2751 genomic regions.

Value

A data frame

Examples

```
data(genomicIntervals)
```

peakreference *combine and merge multiple BED files*

Description

This function merges genomic coordinates of a given data frame or reads in BED files (e.g. generated from a peak caller) under given directory and merge genomic regions that have overlapping genomic intervals into a single feature. The single feature represents the widest genomic interval that covers all merged regions.

Usage

```
peakreference(data = NULL, dir = NULL, pattern = NULL, merge = TRUE,
  overlap = 1, ratio = NULL)
```

Arguments

data	a data frame containg coordinates information of peaks to be merged. Columns of the data frame should be consistent with the BED format where the first column is the name of the chromosome, the second column is the starting position and the third column is the ending position.
dir	character string giving the directory where BED files are stored. If data is not given, the function will reads in the BED files under code.
pattern	an regular expression, only files that have names match the regular expression will be read in.
merge	logical indicating whether to merge overlapped regions or not. If False, regions are simply combined.
overlap	a numberic value giving the least number of base(s) two regions should overlap when merging them.
ratio	a numberic value giving the thresold of overlapping ratio between two regions to merge them. See 'Details' below for the definition of the overlapping ratio.

Details

The overlapping ratio (OR) is defined as:

$$OR = \frac{n}{\min(length(a), length(b)}$$

a, b are two genomic regions, n is the number of overlapping bases between region a and region b.

Value

a data frame with four columns: chr, start, stop, id

Author(s)

Mengjun Wu, Lei Gu

Examples

```
peaks <- data.frame(chr = c(rep('chr1',4),rep('chr2', 3), rep('chr3',2)),
                    start = c(100,148,230,300,330,480,1000,700,801),
                    end = c(150,220,500,450,600,900,1050,760,900))

merged_peaks <- peakreference(data = peaks, merge = TRUE, overlap = 1)
```

TCA-class *TCA class and constructor*

Description

TCA is a S4 class for storing input data, results of differential binding and clustering analysis. A TCA object can be created by the constructor function from a table of sample information, a table genomic coordinates of features, read counts(optional).

Usage

```
TCA(design, counts = matrix(0L, 0L, 0L), genomicFeature, zero.based = TRUE)

TCAFromSummarizedExperiment(se, genomicFeature = NULL)
```

Arguments

design	a data frame containing information about samples/libraries, For time course analysis, design should contain at least three columns (case insensitive): sampleid, timepoint and group providing time point and group information of each sample/library. If counts is not provided when creating TCA object, the column BAMfile can be included in the design data frame, providing corresponding BAM filename of each sample/library, this information can be used for generating count table by using countReads function.
counts	an integer matrix containing read counts. Rows correspond to genomic features and columns to samples/libraries.
genomicFeature	a data frame or a GRanges object containing genomic coordinates of features of interest (e.g. genes in RNA-seq, binding regions in ChIP-seq). If genomicFeature is a data frame, four columns are required in genomicFeature: id, chr, start, end; if genomicFeature is a Granges object, the metadata column "id" is required. For TCAFromSummarizedExperiment, genomicFeature must be provided if se is a SummarizedExperiment object.
zero.based	Logical. If TRUE, the start positions of the genomic ranges in the returned TCA object are *0-based*, if FALSE, the start positions will be *1-based*.
se	A SummarizedExperiment or a RangedSummarizedExperiment object. The object might contain multiple assays (count table) in the assay list, only the first one will be taken to construct TCA object. For SummarizedExperiment object, genomicFeature must be provided while for RangedSummarizedExperiment object, the genomic features will be extracted directly from the object.

Details

A TCA object can be created without providing read counts, read counts can be provided by counts or generated by countReads, the number of rows should equal to that in genomicFeature and the number of columns should equal to number of rows in design. Input data and analysis results in a TCA object can be accessed by using corresponding accessors and functions. The TCA objects also have a show method printing a compact summary of their contents see counts, TCA.accessors, DBresult, tcTable, timeclust. clust

Value

A TCA object

Author(s)

Mengjun Wu

Mengjun Wu

See Also

counts, TCA.accessors, DBresult, timeclust, clust

Examples

```
#create data frame of experiment design: 4 time points and 2 replicates for each time point.
d <- data.frame(sampleID = 1:8, group = rep(c(1, 2, 3, 4), 2),
                timepoint = rep(c('0h', '24h', '48h', '72h'), 2))


#create data frame of genomic intervals of interest
gf <- data.frame(chr = c(rep('chr1', 3), rep('chr2', 2), rep('chr4', 2)),
                 start = seq(100, 2000, by = 300),
                 end = seq(100, 2000, by = 300) + 150,
                 id = paste0('peak', 1:7))
tca <- TCA(design = d, genomicFeature = gf)
genomicFeature(tca)

#if count table is available
c <- matrix(sample(1000, 56), nrow = 7, dimnames = list(paste0('peak', 1:7), 1:8))
tca <- TCA(design = d, counts = c, genomicFeature = gf)
# replace the count table of a \code{TCA} object
c2 <- matrix(sample(500, 56), nrow = 7, dimnames = list(paste0('peak', 1:7), 1:8))
counts(tca) <- c2
```

TCA.accessors *Accessors to extract slots of a TCA class.*

Description

Accessors are provided to extract design, genomicFeature, tcTable, clustResults slots of a TCA class. The design slot stores experimental information of samples/libraries, the genomicFeature slot stores genomic coordinates of features, the tcTable slot stores time couse data as a matrix, where rows are genomic features and columns time points. The clustResults slot stores results of clustering analysis as a clust object.

Usage

```
## S4 method for signature 'TCA'
design(object)

genomicFeature(object)

tcTable(object)

## S4 method for signature 'TCA'
tcTable(object)

clustResults(object)

## S4 method for signature 'TCA'
clustResults(object)
```

Arguments

object	TCA object object

Value

design returns a data frame. genomicFeature returns a data frame. tcTable returns a numeric matrix. clustResults returns a clust object, see clust for details.

Author(s)

Mengjun Wu

See Also

clust

Examples

```
data(tca_ATAC)
genomicFeature(tca_ATAC)
tcTable(tca_ATAC)
```

tca_ATAC *An example TCA object*

Description

A TCA object storing exemplary ATAC-seq time course data, including the experiment design, counts table, reference genomic regions.

Usage

```
data(tca_ATAC)
```

Format

A TCA object of exemplary ATAC-seq time course data

Value

A TCA object

Examples

```
data(tca_ATAC)
```

timeclust *time couse data clustering*

Description

This function performs clustering analysis of time course data.

Usage

```
timeclust(x, algo, k, dist = "euclidean", centers = NULL,
  standardize = TRUE, ...)
```

Arguments

x	a TCA object returned from timecourseTable or a matrix
algo	character string giving a clustering method. Options are 'km' (kmeans), 'pam' (partitioning around medoids), 'hc' (hierachical clustering), 'cm' (cmeans).
k	numeric value between 1 and $n-1$ (n is the number of data points to be clustered).
dist	character string specifying method for distance(dissimilarity) calculation. It should be one of 'correlation' or one of the distance measure method in dist function (for example 'euclidean', 'manhattan')
centers	a numeric matrix giving intial centers for kmeams, pam or cmeans. If given, Number of rows of centers must be equal to k.
standardize	logical, if TRUE, z-score transformation will performed on the data before clustering. See 'Details' below.
...	additional arguments passing to kmeans, pam, hclust, cmeans

Details

two types of clustering methods are provided: hard clustering (kmeans, pam, hclust) and soft clustering(cmeans). In Hard clustering, a data point can only be allocated to exactly one cluster (for hclust, cutree is used to cut a tree into clusters), while in soft clustering (also known as fuzzy clustering), a data point can be assigned to multiple clusters, membership values are used to indicate to what degree a data point belongs to each cluster. For more details, see the help() page of each function.

To avoid the influence of expression level to the clustering analysis, z-score transformation can be applied to covert the expression values to z-scores by performing the following formula:

$$z = \frac{x - \mu}{\sigma}$$

x is value to be converted (e.g., a expression value of a genomic feature in one condition), μ is the population mean (e.g., average expression value of a genomic feature in different conditions), σ is the standard deviation (e.g., standard deviation of expression of a genomic feature in different conditions).

Value

If x is a TCA object, a TCA object will be returned. If x is a matrix, a clust object will be returned

Author(s)

Mengjun Wu

See Also

clust, kmeans, pam, hclust, cutree

Examples

```
x <- matrix(rnorm(1600,sd=0.3), nrow = 200,
            dimnames = list(paste0('peak', 1:200), 1:8))
clust_res <- timeclust(x, algo = 'cm', k = 4) # return a clust object
```

---

timeclustplot *Plot clustering results for time course data.*

Description

This function plots the clusters generated from timeclust. For fuzzy cmeans clustering, data points are color-coded according to membership values, the color palettes can be customized.

Usage

```
timeclustplot(object = NULL, categories = "timepoint",
  value = "expression", cols = NULL, cl.color = "gray50",
  membership.color = rainbow(30, s = 3/4, v = 1, start = 1/6),
  title.size = 18, axis.line.size = 0.6, axis.title.size = 18,
  axis.text.size = 16, legend.title.size = 14, legend.text.size = 14)
```

Arguments

object	a TCA object or a clust object
categories	character string giving the x-axis label
value	character string giving the y-axis label
cols	integer value specifying number of columns in the final layout.
cl.color	character string specifying a color for hard clustering.
membership.color	color palettes, a character vector of n colors
title.size	numeric value specifying the font size of title of each plot in the layout
axis.line.size	numeric value specifying the size of both axis lines
axis.title.size	numeric value specifying the font size of titles of both axis
axis.text.size	numeric value specifying the font size of labels of both axis
legend.title.size	numeric value specifying the font size of legend title
legend.text.size	numeric value specifying the font size of legend text

Value

Plot all clusters in one plot and return a list of ggplot objects, each object is for one cluster. The ggplot object can be drawed by calling print.ggplot

Author(s)

Mengjun Wu

Examples

```
x <- matrix(sample(500, 1600, replace = TRUE), nrow = 200,
            dimnames = list(paste0('peak', 1:200), 1:8))
clust_res <- timeclust(x, algo = 'cm', k = 4, standardize = TRUE)
p <- timeclustplot(clust_res, cols =2)
# to plot a individual cluster
print (p[[2]]) # plot cluster 2
print (p[[3]]) # plot cluster 3
```

---

timecourseTable *constructs time course table for clustering analysis*

Description

This fuction constructs a time course table of which rows corrsponding to genomic features and columns the timepoint. values can be mean normalized read counts or log2-fold changes compared to the first timepoint. The time course table is used for clustering analysis.

Usage

```
timecourseTable(object, value = "expression", lib.norm = TRUE,
  norm.method = "rpkm", subset = NULL, filter = FALSE, pvalue = "fdr",
  pvalue.threshold = 0.05, abs.fold = 2, direction = "both", ...)
```

Arguments

object	a TCA object returned by DBanalysis.
value	character string, either 'expression' or 'FC'. 'expression' is the mean normalized read counts of replicates, FC' is the log2-fold changes compared to the first time point.
lib.norm	logical indicating whether or not use effective library size (see 'Details' in counts).
norm.method	character string specifying the normalization method if value is 'expression'
subset	optinal character vector giving a subset of genomic features, if not NULL, time course table is generated for only this subset of genomic features.
filter	logical, whether to drop the genomic features shows no significant changes (defined by pvalue, pvalue.threshold,abs.fold and direction) between any two time points.
pvalue	character string specify the type of p-values ('PValue' or adjusted p-value 'paj')
pvalue.threshold	a numeric value giving threshold of selected p-value, only features with higher (ajusted) p-values than the threshold are kept.
abs.fold	a numeric value, the least absolute log2-fold changes
direction	character string specify the direction of fold changes ('up' (positive fold changes), down' (negative fold changes), both'(both positive and negative fold changes)), features changes more than abs.fold in the defined direction are kept.
...	additional arguments passing to rpkm, cpm

Value

A TCA object

Note

If 'expression' in value is chosen, for replicates , the normalized expression value is first calculated for each replicate, then mean value is taken to represent the normalized expression value.

Author(s)

Mengjun Wu

Examples

```
data(tca_ATAC)
tca_ATAC <- DBanalysis(tca_ATAC)
tca_ATAC <- timecourseTable(tca_ATAC, value = 'expression',
                            lib.norm = TRUE, norm.method = 'rpkm')
```

Index