Package 'TrendCatcher'

July 18, 2022

Title TrendCatcher: A Versitile R package for analyse longitudinal RNA-seq study

Version 1.0.0

Description Identify dynamic gene from longitudinal/time course RNA-seq dataset.

License GPL (>= 2)

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.0

Depends RColorBrewer,
plyr,
dplyr,
gss,
compiler,
stringr,
gridExtra,
ggplot2,
ggnewscale,
enrichR,
foreach,
circlize,
doSNOW,
reshape2,
pracma,
nlme,
MASS,
sva,
ComplexHeatmap,
DESeq2,
Rcpp,
clusterProfiler,
edgeR,
org.Hs.eg.db,
org.Mm.eg.db

Suggests knitr,
rmarkdown,
pkgdown

VignetteBuilder knitr

Imports biomaRt,
scales

R topics documented:

`cal_p` *Calculate the significance of the dynamic signal for a single gene's single time expression. Compared to baseline NB confidence interval.*

Description

Calculate the significance of the dynamic signal for a single gene's single time expression. Compared to baseline NB confidence interval.

Usage

```
cal_p(obs.val, size, mu)
```

Arguments

obs.val,	estimated value from non-baseline model.
size,	the size of negative binomial model.
mu,	the estimated mean of constant negative binomial model

Value

a numeric p-value for time t.

---

cal_time_p_single_gene

Calculate the significance of the dynamic signal for a single gene's over alll the time points.

Description

Calculate the significance of the dynamic signal for a single gene's over alll the time points.

Usage

```
cal_time_p_single_gene(const.output, spline.output)
```

Arguments

const.output,	list returned from fit_single_gene_const function.
spline.output,	dataframe returned from fit_single_gene_spline function.

Value

p-value for a single point.

Check_CountTable_Format *Check the Input Count Table Format*

Description

This function takes the CSV file path of the count table provided by user. The input count table must be a CSV file, includes integer count table, first column are the GENE SYMBOL or GENE ENSEMBL, and first row are SAMPLE NAME. The count table is a rounded up after normalization and batch correction. Please check TrendCatcher QC functions for normalization and batch correction details. The row names must be in the format of "ProjectName_Time_Rep1" format. ProjectName is a string, can be single letter. Time is a integer. Rep is the replicate ID. This function checked the count table format, and order the sample columns based on its time order. It will return a right formatted count table to run TrendCatcher.

Usage

```
Check_CountTable_Format(count.table.path, min.low.count = 1)
```

Arguments

`count.table.path`,
: string contain the absolute path of the CSV file count table, with first column as GENE SYMBOL or GENE ENSEMBL and first row as SAMPLE NAME (with format composed by project name,time and replicateID, such as "Lung_0_Rep1")

`min.low.count`,
: one numeric variable, the minimal count threshold for filtering low count within each time group. By default it is 1.

Value

A list object, contains "raw.df", original count table ordered by time and replicate ID; "count.table", filtered out low genes for further fitting; "removed.genes", low count genes.

Examples

```
example.file.path<-system.file("extdata", "Brain_DemoCountTable.csv", package = "TrendCatcher")
## Not run:
count<-Check_CountTable_Format(example.file.path, min.low.count = 1)

## End(Not run)
```

combine_p_single_gene *Combine multiple p-value using Fisher's p-value combination method.*

Description

Combine multiple p-value using Fisher's p-value combination method.

Usage

```
combine_p_single_gene(p.arr)
```

Arguments

p.arr,	a vector of p-values.

Value

a numeric combined p-value.

curveFitting.new *CurveFitting function*

Description

To perform Local Polynomial Regression Fitting

Usage

```
curveFitting.new(perm.dat.1, points)
```

Arguments

perm.dat.1,	data to fit the curve.
points,	array of time values.

Value

fitted count over time

`draw_CurveComp` *Draw DDEGs from one biological pathway from two master.list objects and also show LOESS curve fitting for both experimental groups*

Description

For one specific biological pathway, compare its DDEGs from one experimental group to the other one. For example, if group 1 the most dynamic biological pathway from TimeHeatmap is GO term A, and there were 100 DDEGs identified from experimental group 1. We want to see how these DDEGs behave in the other experimental group. Maybe they are also dynamic, but activation/deactivation time may differ. This function will fit LOESS smooth curve fitting for each group and compare the trajectories visually.

Usage

```
draw_CurveComp(
  master.list.1,
  master.list.2,
  ht.1,
  pathway = "",
  group.1.name = "group1",
  group.2.name = "group2"
)
```

Arguments

`master.list.1,`	a list object. The output from run_TrendCatcher function, contains master.table element.
`master.list.2,`	a list object. The output from run_TrendCatcher function, contains master.table element.
`ht.1,`	TimeHeatmap object. The output from draw_TimeHeatmap_GO function, contains GO.df object.
`pathway,`	characters. Must be a biological pathway from GO.df, Description column.
`group.1.name,`	characters. For example, severe group. By default group1.
`group.2.name,`	characters. For example, moderate group. By default group2

Value

A ggplot object and plot.

Examples

```
## Not run:
severe.path<-system.file("extdata", "MasterListSevere.rda", package = "TrendCatcher")
load(severe.path)
moderate.path<-system.file("extdata", "MasterListModerate.rda", package = "TrendCatcher")
load(moderate.path)
ht.path<-system.file("extdata", "htSevere.rda", package = "TrendCatcher")
load(ht.path)
g<-draw_CurveComp(master.list.1 = master.list.severe, master.list.2 = master.list.moderate, ht.1 = ht.severe, pat
print(g)

## End(Not run)
```

---

draw_CurveComp_Perm *Draw DDEGs from one biological pathway from two master.list objects and also show LOESS curve fitting for both experimental groups, and perform permutation test*

Description

For one specific biological pathway, compare its DDEGs from one experimental group to the other one. For example, if group 1 the most dynamic biological pathway from TimeHeatmap is GO term A, and there were 100 DDEGs identified from experimental group 1. We want to see how these DDEGs behave in the other experimental group. Maybe they are also dynamic, but activation/deactivation time may differ. This function will fit LOESS smooth curve fitting for each group and compare the trajectories visually. And run permutation test to see which time interval these two curve is significantlly separated.

Usage

```
draw_CurveComp_Perm(
  master.list.1,
  master.list.2,
  ht.1,
  pathway,
  group.1.name = "group1",
  group.2.name = "group2",
  n.perm = 500,
  parall = F,
  pvalue.threshold = 0.05
)
```

Arguments

`master.list.1,`
: a list object. The output from run_TrendCatcher function, contains master.table element.

`master.list.2`,
: a list object. The output from run_TrendCatcher function, contains master.table element.

`ht.1`,
: TimeHeatmap object. The output from draw_TimeHeatmap_GO function, contains GO.df object.

`pathway`,
: characters. Must be a biological pathway from GO.df, Description column.

`group.1.name`,
: characters. For example, severe group. By default group1.

`group.2.name`,
: characters. For example, moderate group. By default group2

`n.perm`,
: an integer variable, the number of repeated times to run permutation test. By default is 500.

`parall`,
: a logical variable. If users want to run using multiple core, set it to TRUE. By default is FALSE.

`pvalue.threshold`,
: a numeric variable. The adjusted p-value for permutation test. By default is 0.05.

Value

a list object. Contains elements named adjusted.pvalue.area, perm, st, en and plot. adjusted.pvalue.area is the adjusted p-value for each chopped small time interval area compared to the permutation test. perm is the permutations test result for each individual run. st is the start separation time. en is the end of the separation time. plot is the ggplot.

Examples

```
## Not run:
severe.path<-system.file("extdata", "MasterListSevere.rda", package = "TrendCatcher")
load(severe.path)
moderate.path<-system.file("extdata", "MasterListModerate.rda", package = "TrendCatcher")
load(moderate.path)
ht.path<-system.file("extdata", "htSevere.rda", package = "TrendCatcher")
load(ht.path)
perm_output<-draw_CurveComp_Perm(master.list.1 = master.list.severe,
                                 master.list.2 = master.list.moderate,
                                 ht.1 = ht.severe, pathway = "neutrophil activation",
                                 group.1.name = "severe",
                                 group.2.name = "moderate",
                                 n.perm = 100,
                                 parall = FALSE,
                                  pvalue.threshold = 0.05)
print(perm_output$plot)

## End(Not run)
```

draw_GeneTraj *Draw gene(s) trajectory with observed data and fitted data*

Description

This function takes the master.list object output from run_TrendCatcher function, and an array of gene(s). It will draw gene(s) trajectory with observed data and fitted data.

Usage

```
draw_GeneTraj(
  master.list,
  gene.symbol.arr,
  savepdf.path = NA,
  ncol = 5,
  nrow = 3,
  fig.width = 15,
  fig.height = 10
)
```

Arguments

master.list,	a list object. Output from the run_TrendCatcher with ID conversion to add Symbol column to master table.
gene.symbol.arr,	a character array. It must be a subset of row names from the master.list$master.table$Symbol. The Symbol column need get_GeneEnsembl2Symbol function to convert original ensembl ID into gene symbol.
savepdf.path,	an obsolute file path to save the figure as PDF file. By default is NA, it will be printed.
ncol,	an integer variable. If more than one gene need to be plotted, it will layout as grid structure. This represents the number of column of the grid layout.
nrow,	an integer variable. If more than one gene need to be plotted, it will layout as grid structure. This represents the number of row of the grid layout.
fig.width,	a numeric variable. If save figure as PDF file, the width of the PDF file. By default is 15.
fig.height,	a numeric variable. If save figure as PDF file, the height of the PDF file. By default is 10.

Value

"arrangelist" "list" object.

`draw_GOHeatmap` *Draw GOHeatmap containing Terms from TimeHeatmap and included Genes*

Description

This funcitons takes the master.list output from run_TrendCatcher, and merge.df output from draw_TimeHeatmap_GO and draw_TimeHeatmap_enrichR. And showing all the genes used for enrichment analysis and their logFC compared to previous break point.

Usage

```
draw_GOHeatmap(
  master.list,
  time.window = "",
  go.terms = "",
  merge.df = NA,
  logFC.thres = 2,
  figure.title = "",
  save.tiff.path = NA,
  tiff.res = 100,
  tiff.width = 1500,
  tiff.height = 1500
)
```

Arguments

`master.list,`	a list object. The output from run_TrendCatcher function, contains master.table element.
`time.window,`	a character. Must be one of the merge.df$t.name.
`go.terms,`	a character array. Must be an array of go terms from the merge.df$Description.
`merge.df,`	a dataframe. The output dataframe from output list of draw_TimeHeatmap_GO or draw_TimeHeatmap_enrichR. Use $merge.df to obtain it.
`logFC.thres,`	a numeric variable. The logFC threshold compared to each genes previous break point expression level. By default is 2, meaning for each gene, the current time window's expression level is 2-fold compared to previous break point's expression level.
`figure.title,`	character
`save.tiff.path,`	by default is NA
`tiff.res,`	resolution
`tiff.width,`	figure width
`tiff.height,`	figure height

Value

A list object, including elements named GOheatmap and GOheatmapDat. GOheatmap is a ComplexHeatmap object for figure. GOheatmapDat is a data.frame include log2FC value of each gene's expression change compared to the previous break point.

Examples

```
## Not run:
example.file.path<-system.file("extdata", "BrainMasterList.rda", package = "TrendCatcher")
load(example.file.path)
time_heatmap<-draw_TimeHeatmap_GO(master.list = master.list)
merge.df<-time_heatmap$merg.df
time.window<-"0h-6h"
go.terms<-c("regulation of defense response", "leukocyte migration", "myeloid leukocyte migration", "leukocyte che
"granulocyte chemotaxis", "cellular response to chemokine", "chemokine-mediated signaling pathway", "angiogenesis
go.df<-draw_GOHeatmap(master.list = master.list, time.window = "0h-6h", go.terms = go.terms, merge.df = merge.df,

## End(Not run)
```

---

draw_TimeHeatmap_enrichR

Draw TimeHeatmap Using enrichR

Description

This funcitons takes the master.list output from run_TrendCatcher. And apply a time window sliding strategy to capture all the genes increased/decreased compared to its previous break point, and apply enrichR enrichment analysis.

Usage

```
draw_TimeHeatmap_enrichR(
  master.list,
  logFC.thres = 0,
  top.n = 10,
  dyn.gene.p.thres = 0.05,
  dbs = "BioPlanet_2019",
  term.width = 80,
  OrgDb = "org.Mm.eg.db",
  GO.enrich.p = 0.05,
  figure.title = "",
  save.tiff.path = NA,
  tiff.res = 100,
  tiff.width = 1500,
  tiff.height = 1500
)
```

Arguments

master.list,	a list object. The output from run_TrendCatcher function, contains master.table element.
logFC.thres,	a numeric variable. The logFC threshold compared to each genes previous break point expression level. By default is 0, meaning for each gene, the current time window's expression level is 2-fold compared to previous break point's expression level.
top.n,	an integer variable. The top N GO enrichment term need to be shown in the TimeHeatmap for up and down regulated pathway. By default is 10. Top 20 GO terms, 10 from up-regulated pathway and 10 from down-regulated pathway will shown in TimeHeatmap.
dyn.gene.p.thres,	a numeric variable. The DDEGs dynamic p-value threshold. By default is 0.05.
dbs,	must one of the enrichR supported database name. To check the list, run dbs <- listEnrichrDbs() command. By default is "BioPlanet_2019".
term.width,	an integer variable. The character length for each GO term. If one GO term is super long, we can wrap it into term.width of strings into multiple rows. By default if 80.
OrgDb,	must be either "org.Mm.eg.db" or "org.Hs.eg.db". Currently only support mouse and human GO annotation database.
GO.enrich.p,	an numeric variable. The GO enrichment p-value threshold. By default if 0.05.
figure.title,	a character variable. The main title of TimeHeatmap.
save.tiff.path,	a character variable, the file path to save the TIFF figure. If set to NA, it will plot it out. By default is NA.
tiff.res,	a numeric variable, the resolution of the TIFF figure. By default is 100.
tiff.width,	a numeric variable, the width of the TIFF figure. By default is 1500.
tiff.height,	a numeric variable, the height of the TIFF figure. By default is 1500.

Value

A list object, including elements names time.heatmap, merge.df and GO.df. time.heatmap is the ComplexHeatmap object. merge.df includes all the GO enrichment result and their activation/deactivation time window. GO.df includes GO enrichment used for plot TimeHeatmap and all the individual genes within each time window.

Examples

```
## Not run:
example.file.path<-system.file("extdata", "BrainMasterList.rda", package = "TrendCatcher")
load(example.file.path)
gene.symbol.df<-get_GeneEnsembl2Symbol(ensemble.arr = master.list$master.table$Gene)
th.obj<-draw_TimeHeatmap_enrichR(master.list = master.list)
print(th.obj$time.heatmap)
head(th.obj$merge.df)

## End(Not run)
```

`draw_TimeHeatmap_GO` *Draw TimeHeatmap Using Gene Ontology (GO) Enrichment*

Description

This funcitons takes the master.list output from run_TrendCatcher. And apply a time window sliding strategy to capture all the DDEGs increased/decreased compared to its previous break point, and apply GO enrichment analysis.

Usage

```
draw_TimeHeatmap_GO(
  master.list,
  logFC.thres = 0,
  top.n = 10,
  dyn.gene.p.thres = 0.05,
  keyType = "SYMBOL",
  OrgDb = "org.Mm.eg.db",
  ont = "BP",
  term.width = 80,
  GO.enrich.p = 0.05,
  figure.title = "",
  save.tiff.path = NA,
  tiff.res = 100,
  tiff.width = 1500,
  tiff.height = 1500
)
```

Arguments

`master.list,`	a list object. The output from run_TrendCatcher function, contains master.table element.
`logFC.thres,`	a numeric variable. The logFC threshold compared to each gene's previous break point expression level. By default is 0, meaning for each gene, the current time window's expression level is 2-fold compared to previous break point's expression level.
`top.n,`	an integer variable. The top N GO enrichment term need to be shown in the TimeHeatmap for up and down regulated pathway (based on the change of first time interval). By default is 10. Top 20 GO terms, 10 from up-regulated pathway and 10 from down-regulated pathway will shown in TimeHeatmap.
`dyn.gene.p.thres,`	a numeric variable. The DDEGs adjusted dynamic p-value threshold. By default is 0.05.
`keyType,`	must be either ENSEMBL or SYMBOL. The row names of your master.list$master.table.
`OrgDb,`	must be either "org.Mm.eg.db" or "org.Hs.eg.db". Currently only support mouse and human GO annotation database.

ont,	one of "BP", "MF", and "CC" sub ontologies, or "ALL" for all three. By default is "BP".
term.width,	an integer variable. The character length for each GO term. If one GO term is super long, we can wrap it into term.width of strings into multiple rows. By default is 80.
GO.enrich.p,	an numeric variable. The GO enrichment p-value threshold. By default if 0.05.
figure.title,	a character variable. The main title of TimeHeatmap.
save.tiff.path,	a character variable, the file path to save the TIFF figure. If set to NA, it will plot it out. By default is NA.
tiff.res,	a numeric variable, the resolution of the TIFF figure. By default is 100.
tiff.width,	a numeric variable, the width of the TIFF figure. By default is 1500.
tiff.height,	a numeric variable, the height of the TIFF figure. By default is 1500.

Value

A list object, including elements names time.heatmap, merge.df and GO.df. time.heatmap is the ComplexHeatmap object. merge.df includes all the GO enrichment result and their activation/deactivation time window. GO.df includes GO enrichment used for plot TimeHeatmap and all the individual genes within each time window.

Examples

```
## Not run:
example.file.path<-system.file("extdata", "BrainMasterList.rda", package = "TrendCatcher")
load(example.file.path)
gene.symbol.df<-get_GeneEnsembl2Symbol(ensemble.arr = master.list$master.table$Gene)
th.obj<-draw_TimeHeatmap_GO(master.list = master.list)
print(th.obj$time.heatmap)
head(th.obj$merge.df)

## End(Not run)
```

---

draw_TimeHeatmap_selGO

Subset TimeHeatmap by providing a manually selected non-redundant GO terms

---

Description

Some GO terms are redundant. Users can manually select GO terms that are shown in the GO.df element in the TimeHeatmap object and show the TimeHeatmap figure.

Usage

```
draw_TimeHeatmap_selGO(
  time_heatmap,
  sel.go,
  master.list,
  GO.perc.thres = 0,
  nDDEG.thres = 0,
  term.width = 80,
  figure.title = "",
  save.tiff.path = NA,
  tiff.res = 100,
  tiff.width = 1500,
  tiff.height = 1500
)
```

Arguments

time_heatmap,	a list, the output of draw_TimeHeatmap_GO function. A TimeHeatmap object, with GO.df element included.
sel.go,	a character variable. An array of character names of GO terms, that match the GO terms from the Description column of GO.df.
master.list,	a list, the output of run_TrendCatcher function, a master.list object.
GO.perc.thres,	a numeric variable. A threshold to filter out GOs that only a little percentage of the genes are DDEGs. By default is 0.
nDDEG.thres,	an integer variable. A threshold to filter out GOs that only a small number of genes included. By default is 0.
term.width,	an integer variable. The character length for each GO term. If one GO term is super long, we can wrap it into term.width of strings into multiple rows. By default is 80.
figure.title,	a character variable. The main title of TimeHeatmap.
save.tiff.path,	a character variable, the file path to save the TIFF figure. If set to NA, it will plot it out. By default is NA.
tiff.res,	a numeric variable, the resolution of the TIFF figure. By default is 100.
tiff.width,	a numeric variable, the width of the TIFF figure. By default is 1500.
tiff.height,	a numeric variable, the height of the TIFF figure. By default is 1500.

Value

A list object, including elements names time.heatmap, merge.df and GO.df. time.heatmap is the ComplexHeatmap object. merge.df includes all the GO enrichment result and their activation/deactivation time window. GO.df includes GO enrichment used for plot TimeHeatmap and all the individual genes within each time window.

Examples

```
## Not run:
example.file.path<-system.file("extdata", "BrainMasterList.rda", package = "TrendCatcher")
load(example.file.path)
gene.symbol.df<-get_GeneEnsembl2Symbol(ensemble.arr = master.list$master.table$Gene)
time_heatmap<-draw_TimeHeatmap_GO(master.list = master.list)
go.terms<-unique(time_heatmap$GO.df$Description)[1:5]
time_heatmap_selGO<-draw_TimeHeatmap_selGO(time_heatmap = time_heatmap, sel.go = go.terms, master.list = master.l

## End(Not run)
```

`draw_TrajClusterGrid` *Draw Grouped DDEGs Trajectories in Grid Plot*

Description

Group all DDEGs based on their sub-type trajectory patterns and plot their trajectories together, then layout all sub-type trajectory patterns which contains more than N genes in a grid plot. Each individual sub-grid plot is titled with sub-type trajectory pattern and number of genes included. X-axis is the time, Y-aixs is log2 transformed fitted count trajectory.

Usage

```
draw_TrajClusterGrid(
  master.list,
  min.traj.n = 10,
  save.as.PDF = NA,
  pdf.width = 10,
  pdf.height = 10
)
```

Arguments

`master.list,`	a list object. The output from run_TrendCatcher function, contains master.table element.
`min.traj.n,`	an integer variable. The minimum number of genes from the same sub-type trajectory. By default if 10.
`save.as.PDF,`	a string. The absolute file path to save the figure as PDF file if needed. If set to NA, will print the figure instead of saving it as PDF file.
`pdf.width,`	a numeric variable. The PDF file width size. By default is 10.
`pdf.height,`	a numeric variable. The PDF file height size. By default is 10.

draw_TrajClusterPie *Draw Grouped DDEGs Main-type and Sub-type Composition in Hierarchical Pie Chart*

Description

Group all DDEGs based on their main-type and sub-type trajectory patterns and plot their composition in a hierachical pie chart. Inner pie chart represents the main-type trajectory pattern composition. The outer pie chart represents sub-type trajectory pattern composition.

Usage

```
draw_TrajClusterPie(
  master.list,
  fig.title = "",
  inner.radius = 0.7,
  cex.out = 1,
  cex.in = 1
)
```

Arguments

master.list,	a list object. The output from run_TrendCatcher function, contains master.table element.
fig.title,	a string. The main title of the figure. By default if "".
inner.radius,	a numeric variable. The inner pie chart radius size. By default is 0.7.
cex.out,	a numeric variable. The text size of label of outer pie chart. By default is 1.
cex.in,	a numeric variable. The text size of the label of inner pie chart. By default is 1.

Value

plot.

fit_single_gene_const *Fit the baseline count data into a constant negative binomial model.*

Description

Fit the baseline count data into a constant negative binomial model.

Usage

```
fit_single_gene_const(
  count.arr,
  disp.var,
  MAXIT = 1000,
  RELTOL = 10^(-8),
  trace = 10
)
```

Arguments

count.arr,	a data frame returned from transform_single_gene_df, only with the baseline time.
disp.var,	the dispersion value estimated from DESeq2.
MAXIT,	1.
RELTOL,	10*-8
trace,	10

Value

a list contain all the estimated value from NB model.

---

fit_single_gene_spline

Fit the non-baseline count data into a smoothed ANOVA model.

Description

Fit the non-baseline count data into a smoothed ANOVA model.

Usage

```
fit_single_gene_spline(data.trans)
```

Arguments

data.trans,	a data frame returned from transform_single_gene_df, without the baseline time.

Value

a dataframe with all the fitted count from spline model.

get_GeneEnsembl2Symbol *ID convention (from ENSEMBL to SYMBOL)*

Description

This function takes an array of ENSEMBL ID and convert it into GENE SYMBOL.

Usage

```
get_GeneEnsembl2Symbol(ensemble.arr, dataset = "mmusculus_gene_ensembl")
```

Arguments

ensemble.arr,	a character array. The ENSEMBL ID array.
dataset,	must be either "mmusculus_gene_ensembl" or "hsapiens_gene_ensembl".

Value

A data frame contains 3 columns, "Gene", "Symbol" and "description".

get_rep_array *Get the replicate array from count table*

Description

It take the count table, grep the replicate element from column name. The column name of the count table must satisfy "Prj_Time_Rep1" format.

Usage

```
get_rep_array(raw.count.df)
```

Arguments

raw.count.df,	a count table.

Value

a numeric array of each sample's replicate.

get_Symbol2GeneEnsembl *ID convention (from SYMBOL to ENSEMBL)*

Description

This function takes an array of SYMBOL and convert it into GENE ENSEMBL.

Usage

```
get_Symbol2GeneEnsembl(symbol.arr, dataset = "mmusculus_gene_ensembl")
```

Arguments

symbol.arr,	a character array. The SYMBOL array.
dataset,	must be either "mmusculus_gene_ensembl" or "hsapiens_gene_ensembl".

Value

A data frame contains 3 columns, "Gene", original ID, "Symbol" and "description".

get_time_array *Get the time array from count table*

Description

It take the count table, grep the time element from column name. The column name of the count table must satisfy `Prj_Time_Rep1` format.

Usage

```
get_time_array(raw.count.df)
```

Arguments

raw.count.df,	a count table.

Value

a numeric array of each sample's time.

permutation.new *Permutation test*

Description

Function to perform permutation test.

Usage

```
permutation.new(perm.dat, n.perm = 100, points, parall = FALSE)
```

Arguments

perm.dat,	temporal data frame to perform permutation.
n.perm,	time to perm
points,	array of time values
parall,	run using multiple core

Value

permutation test output data

preprocess_TrendCatcher *Preprocessing for TrendCatcher*

Description

This is the preprocessing function to prepare the input count table for run_TrendCatcher function. It takes the CSV file count table and logic variables to check if normalization and batch correction needed for preprocessing. It creates pdf figure report to show before and after of normalization and batch correction to assess quality control (QC).

Usage

```
preprocess_TrendCatcher(
  count.table.path = "",
  need.batch.correction = TRUE,
  need.normalization = TRUE,
  batch.arr = "",
  pdf.file.path = NA,
  pdf.width = 8,
  pdf.height = 10,
  n.low.count = 10
)
```

Arguments

`count.table.path,`
: string contain the absolute path of the CSV file count table, with first column as GENE SYMBOL or GENE ENSEMBL and first row as SAMPLE NAME (with format composed by project name,time and replicateID, such as "Lung_0_Rep1")

`need.batch.correction,`
: logic variable. If batch correction is needed. By default is TRUE.

`need.normalization,`
: logic variable. If normalization is needed. By default is TRUE.

`batch.arr,`
: a numeric vector of batch number. Need to be the same length as the number of samples.

`pdf.file.path,`
: an absolute file path for save the QC report file. If not need, set if to NA. The report will be printed. By default is NA.

`pdf.width,`
: a numeric variable. The PDF file width size. By default is 8.

`pdf.height,`
: a numeric variable. The PDF file height size. By default is 10.

`n.low.count,`
: a numeric variable. The minimal number to filter low count genes. By default is 10.

Value

A matrix array object.

Examples

```
example.file.path<-system.file("extdata", "Brain_DemoRawCountTable.csv", package = "TrendCatcher")
## Not run:
count.table<-preprocess_TrendCatcher(count.table.path = example.file.path,
need.batch.correction = TRUE,
need.normalization = TRUE,
batch.arr<-c(2,2,3,3,3,
             3,3,3,3,0,
             2,3,0,4,
             3,3,2,2,0,4,
             3,3,2,0,
              2,2,0,4),
pdf.file.path = NA,
pdf.width=8, pdf.height=10,
n.low.count = 10)

## End(Not run)
```

run_TrendCatcher *Run TrendCatcher Main Algorithm*

Description

This is the main function to run TrendCatcher to identify Dynamic Differentially Expressed Genes (DDEGs). This function loads a rounded count matrix CSV file after the normalization and batch correction, run the core algorithm and output a list object contains all the genes dynamic information.

Usage

```
run_TrendCatcher(
 count.table.path = "~/Documents/TrendCatcher/inst/extdata/Lung_DemoCountTable.csv",
  baseline.t = 0,
  time.unit = "h",
  min.low.count = 1,
  para.core.n = NA,
  dyn.p.thres = 0.05,
  show.verbose = F
)
```

Arguments

count.table.path,	string contain the absolute path of the CSV file count table, with first column as GENE SYMBOL or GENE ENSEMBL and first row as SAMPLE NAME (with format composed by project name,time and replicateID, such as "Lung_0_Rep1")
baseline.t,	one numeric variable, the baseline time of the longitudinal study. By default it is 0.
time.unit,	one character variable, the time unit of longitudinal study. If choose hour, please transform all sample collecting time into hour.
min.low.count,	one numeric variable, the minimal count threshold for filtering low count within each time group. By default it is 1.
para.core.n,	one numeric variable, number of cores will be used for running TrendCatcher parallel. By default it is NA, which will use N-1 cores from computer.
dyn.p.thres,	one numeric variable, the threshold of adjusted p-value of the dynamic gene. By default 0.05.
show.verbose,	logic variable. If gssanova fitting failed, users can set this to TRUE, it will print out which gene failed the fitting. This process takes only one CPU, so it may be slower than the multi-core version. Normally the fitting failure is caused by low count genes. Users can manually remove it from your count table. By default set to FALSE.

Value

A list object, including "time.unit", "baseline.t", "t.arr", "Project.name", "raw.df", "fitted.count" and "master.table".

Examples

```
example.file.path<-system.file("extdata", "Brain_DemoCountTable.csv", package = "TrendCatcher")
## Not run:
master.list<-run_TrendCatcher(count.table.path = example.file.path,
baseline.t = 0,
time.unit = "h",
min.low.count = 1,
para.core.n = NA,
dyn.p.thres = 0.05,
show.verbose = FALSE)

## End(Not run)
```

---

transform_single_gene_df

Convert a single gene's count row number into data frame with two columns.

Description

Convert a single gene's count row number into data frame with two columns.

Usage

```
transform_single_gene_df(gene.row.info, gene.name, time.arr, rep.arr)
```

Arguments

gene.row.info,	a single row from count table.
gene.name,	the gene name.
time.arr,	the return value from get_time_array function.
rep.arr,	the return value from get_rep_array function.

Value

a dataframe object ordered by time and replicate id.

Index