

Package: ReporterScore (via r-universe)

September 15, 2024

Type Package

Title Generalized Reporter Score-Based Enrichment Analysis for Omics Data

Version 0.1.8

Description Inspired by the classic 'RSA', we developed the improved 'Generalized Reporter Score-based Analysis (GRSA)' method, implemented in the R package 'ReporterScore', along with comprehensive visualization methods and pathway databases. 'GRSA' is a threshold-free method that works well with all types of biomedical features, such as genes, chemical compounds, and microbial species. Importantly, the 'GRSA' supports multi-group and longitudinal experimental designs, because of the included multi-group-compatible statistical methods.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.3

Imports magrittr, dplyr, stats, ggplot2 ($\geq 3.2.0$), pcutils ($\geq 0.2.5$), utils, scales, ggnewscale, ggrepel, reshape2, stringr, foreach

Suggests knitr, rmarkdown, plyr, e1071, factoextra, snow, doSNOW, pheatmap, readr, R.utils, KEGGREST, clusterProfiler, enrichplot, pathview, GSA, vegan, MetaNet, igraph, ggraph, PADOG, safe, rSEA, GSA

Depends R ($\geq 4.2.0$)

VignetteBuilder knitr

BugReports <https://github.com/Asa12138/ReporterScore/issues>

URL <https://github.com/Asa12138/ReporterScore>

Repository <https://asa12138.r-universe.dev>

RemoteUrl <https://github.com/asa12138/reporterscore>

RemoteRef HEAD

RemoteSha 2cf66ab9d0716891182a9b53de8531148d7aaa96

Contents

| | |
|--------------------------------------|----|
| cm_test_k | 3 |
| combine_rs_res | 4 |
| Compound_htable | 5 |
| CPDlist | 5 |
| custom_modulelist | 6 |
| custom_modulelist_from_org | 7 |
| export_report_table | 8 |
| gene2ko | 8 |
| genedf | 9 |
| get_features | 9 |
| get_reporter_score | 10 |
| Golist | 11 |
| hsa_kegg_pathway | 11 |
| ko.test | 12 |
| KOlist | 13 |
| KO_abundance | 13 |
| KO_enrich | 14 |
| KO_fisher | 15 |
| KO_gsa | 16 |
| KO_gsea | 17 |
| KO_gsva | 18 |
| KO_htable | 19 |
| KO_padog | 19 |
| KO_safe | 20 |
| KO_sea | 21 |
| load_CARDinfo | 22 |
| load_Golist | 23 |
| load_htable | 23 |
| mmu_kegg_pathway | 25 |
| modify_description | 25 |
| Module_htable | 26 |
| Pathway_htable | 26 |
| plot.cm_res | 26 |
| plot_enrich_res | 27 |
| plot_features_box | 28 |
| plot_features_distribution | 30 |
| plot_features_heatmap | 31 |
| plot_features_in_pathway | 32 |
| plot_features_network | 33 |
| plot_htable | 34 |
| plot_KEGG_map | 35 |
| plot_report | 36 |
| plot_report_circle_packing | 37 |
| plot_significance | 38 |
| print.reporter_score | 39 |
| print.rs_by_cm | 39 |

| | |
|------------------------------|-----------|
| <code>cm_test_k</code> | 3 |
| pvalue2zs | 40 |
| reporter_score | 42 |
| reporter_score_res | 44 |
| RSA_by_cm | 45 |
| update_CARDinfo | 47 |
| update_GOlist | 47 |
| update_KEGG | 48 |
| up_level_KO | 49 |
| Index | 50 |

| | |
|------------------------|---|
| <code>cm_test_k</code> | <i>Test the proper clusters k for c_means</i> |
|------------------------|---|

Description

Test the proper clusters k for `c_means`
C-means cluster

Usage

```
cm_test_k(otu_group, filter_var, fast = TRUE)

c_means(otu_group, k_num, filter_var)
```

Arguments

| | |
|-------------------------|--------------------------|
| <code>otu_group</code> | standardize data |
| <code>filter_var</code> | filter the highest var |
| <code>fast</code> | whether do the gap_stat? |
| <code>k_num</code> | cluster number |

Value

ggplot
ggplot

See Also

Other C_means: [RSA_by_cm\(\)](#)

Examples

```

if (requireNamespace("e1071") && requireNamespace("factoextra")) {
  data(otutab, package = "pcutils")
  pcutils::hebing(otutab, metadata$Group) -> otu_group
  cm_test_k(otu_group, filter_var = 0.7)
  cm_res <- c_means(otu_group, k_num = 3, filter_var = 0.7)
  plot(cm_res, 0.8)
}

```

 combine_rs_res

Combine the results of 'step by step GRSA'

Description

Combine the results of 'step by step GRSA'

Usage

```
combine_rs_res(kodf, group, metadata, ko_stat, reporter_s, modulelist = NULL)
```

Arguments

| | |
|------------|--|
| kodf | KO_abundance table, rowname are feature ids (e.g. K00001 if feature="ko"; PEX111A if feature="gene"; C00024 if feature="compound"), colnames are samples. |
| group | The comparison groups (at least two categories) in your data, one column name of metadata when metadata exist or a vector whose length equal to columns number of kodf. And you can use factor levels to change order. |
| metadata | sample information data.frame contains group |
| ko_stat | result of pvalue2zs |
| reporter_s | result of get_reporter_score |
| modulelist | NULL or customized modulelist dataframe, must contain 'id', 'K_num', 'KOs', 'Description' columns. Take the 'KOList' as example, use custom_modulelist . |

Value

reporter_score object

See Also

Other GRSA: [get_reporter_score\(\)](#), [ko.test\(\)](#), [pvalue2zs\(\)](#), [reporter_score\(\)](#)

Examples

```

data("KO_abundance_test")
ko_pvalue <- ko.test(KO_abundance, "Group", metadata)
ko_stat <- pvalue2zs(ko_pvalue, mode = "directed")
reporter_s1 <- get_reporter_score(ko_stat, perm = 499)
reporter_res <- combine_rs_res(KO_abundance, "Group", metadata, ko_stat, reporter_s1)

```

| | |
|-----------------|------------------------------------|
| Compound_htable | <i>Compound htable from 'KEGG'</i> |
|-----------------|------------------------------------|

Description

Compound htable from 'KEGG'

See Also

Other data: [CPDlist](#), [Golist](#), [KO_htable](#), [Kolist](#), [Module_htable](#), [Pathway_htable](#), [hsa_kegg_pathway](#), [mmu_kegg_pathway](#)

| | |
|---------|---|
| CPDlist | <i>The CPDlist used for enrichment.</i> |
|---------|---|

Description

an list contains two data.frame named pathway and module.

Format

four columns in each data.frame.

id "map0010" or "M00001"

K_num contains how many Compounds in this pathway or module

KOs Compounds name

Description the description of this pathway or module

See Also

Other data: [Compound_htable](#), [Golist](#), [KO_htable](#), [Kolist](#), [Module_htable](#), [Pathway_htable](#), [hsa_kegg_pathway](#), [mmu_kegg_pathway](#)

custom_modulelist *Build a custom modulelist*

Description

Build a custom modulelist

Transform a modulelist to a list

Usage

```
custom_modulelist(pathway2ko, pathway2desc = NULL, verbose = TRUE)
```

```
transform_modulelist(mymodulelist, mode = 1)
```

Arguments

pathway2ko user input annotation of Pathway to KO mapping, a data.frame of 2 column with pathway and ko.

pathway2desc user input of Pathway TO Description mapping, a data.frame of 2 column with pathway and description.

verbose verbose

mymodulelist mymodulelist

mode 1~2

Value

a custom modulelist

modulelist

See Also

Other modulelist: [custom_modulelist_from_org\(\)](#), [get_features\(\)](#)

Other modulelist: [custom_modulelist_from_org\(\)](#), [get_features\(\)](#)

Examples

```
mydat <- data.frame(pathway = paste0("PATHWAY", rep(seq_len(2), each = 5)), ko = paste0("K", 1:10))
mymodulelist <- custom_modulelist(mydat)
print(mymodulelist)
transform_modulelist(mymodulelist)
```

`custom_modulelist_from_org`*Custom modulelist from a specific organism*

Description

Custom modulelist from a specific organism

Usage

```
custom_modulelist_from_org(  
  org = "hsa",  
  feature = "ko",  
  gene = "symbol",  
  verbose = TRUE  
)
```

Arguments

| | |
|----------------------|---|
| <code>org</code> | kegg organism, listed in https://www.genome.jp/kegg/catalog/org_list.html , default, "hsa" |
| <code>feature</code> | one of "ko", "gene", "compound" |
| <code>gene</code> | one of "symbol", "id" |
| <code>verbose</code> | logical |

Value

modulelist

See Also

Other modulelist: [custom_modulelist\(\)](#), [get_features\(\)](#)

Examples

```
hsa_pathway <- custom_modulelist_from_org(org = "hsa", feature = "gene")
```

export_report_table *Export report score result tables*

Description

Export report score result tables

Usage

```
export_report_table(reporter_res, dir_name, overwrite = FALSE)
```

Arguments

| | |
|--------------|---|
| reporter_res | a reporter_score object or rs_by_cm object |
| dir_name | the directory to save the report tables |
| overwrite | overwrite the existed files or not, default is FALSE. |

Value

No return value

gene2ko *Transfer gene symbol table to KO table*

Description

You can use 'clusterProfiler::bitr()' to transfer your table from other gene_id to gene_symbol.

Usage

```
gene2ko(genedf, org = "hsa")
```

Arguments

| | |
|--------|---|
| genedf | ,rowname is gene symbol (e.g. PFKM), colnames is samples |
| org | kegg organism, listed in 'https://www.genome.jp/kegg/catalog/org_list.html', default, 'hsa' |

Value

kodf

Examples

```
data("genedf")
K0df <- gene2ko(genedf, org = "hsa")
```

| | |
|--------|-------------------------|
| genedf | <i>human gene table</i> |
|--------|-------------------------|

Description

human gene table

See Also

Other test_data: [KO_abundance](#), [reporter_score_res](#)

| | |
|--------------|-------------------------------------|
| get_features | <i>get features in a modulelist</i> |
|--------------|-------------------------------------|

Description

get features in a modulelist

Usage

```
get_features(map_id = "map00010", ko_stat = NULL, modulelist = NULL)
```

Arguments

| | |
|------------|--|
| map_id | map_id in modulelist |
| ko_stat | NULL or ko_stat result from pvalue2zs |
| modulelist | NULL or customized modulelist dataframe, must contain 'id', 'K_num', 'KOs', 'Description' columns. Take the 'KOlist' as example, use custom_modulelist . |

Value

KOids, or data.frame with these KOids.

See Also

Other modulelist: [custom_modulelist_from_org\(\)](#), [custom_modulelist\(\)](#)

Examples

```
get_features(map_id = "map00010")
```

get_reporter_score *Calculate reporter score*

Description

Calculate reporter score

Usage

```
get_reporter_score(
  ko_stat,
  type = c("pathway", "module")[1],
  feature = "ko",
  threads = 1,
  modulelist = NULL,
  perm = 4999,
  verbose = TRUE,
  p.adjust.method2 = "BH",
  min_exist_KO = 3,
  max_exist_KO = 600
)
```

Arguments

| | |
|------------------|--|
| ko_stat | ko_stat result from pvalue2zs |
| type | 'pathway' or 'module' for default KOList for microbiome, 'CC', 'MF', 'BP', 'ALL' for default GOList for homo sapiens. And org in listed in 'https://www.genome.jp/kegg/catalog/org_...' such as 'hsa' (if your kodf is come from a specific organism, you should specify type here). |
| feature | one of 'ko', 'gene', 'compound' |
| threads | default 1 |
| modulelist | NULL or customized modulelist dataframe, must contain 'id', 'K_num', 'KOs', 'Description' columns. Take the 'KOList' as example, use custom_modulelist . |
| perm | permutation number, default: 4999. |
| verbose | logical |
| p.adjust.method2 | p.adjust.method for the correction of ReporterScore, see p.adjust |
| min_exist_KO | min exist KO number in a pathway (default, 3, when a pathway contains KOs less than 3, there will be no RS) |
| max_exist_KO | max exist KO number in a pathway (default, 600, when a pathway contains KOs more than 600, there will be no RS) |

Value

reporter_res data.frame

See Also

Other GRSA: [combine_rs_res\(\)](#), [ko.test\(\)](#), [pvalue2zs\(\)](#), [reporter_score\(\)](#)

Examples

```
data("KO_abundance_test")
ko_pvalue <- ko.test(KO_abundance, "Group", metadata)
ko_stat <- pvalue2zs(ko_pvalue, mode = "directed")
reporter_s1 <- get_reporter_score(ko_stat, perm = 499)
```

Golist

The Golist used for enrichment.

Description

an list contains three data.frame named BP, CC, MF.

Format

four columns in each data.frame.

id "map0010" or "M00001"

K_num contains how many Genes in this GO term

KOs Genes name

Description the description of this GO term

See Also

Other data: [CPDlist](#), [Compound_htable](#), [KO_htable](#), [Kolist](#), [Module_htable](#), [Pathway_htable](#), [hsa_kegg_pathway](#), [mmu_kegg_pathway](#)

hsa_kegg_pathway

pathway information for "hsa"

Description

pathway information for "hsa"

See Also

Other data: [CPDlist](#), [Compound_htable](#), [Golist](#), [KO_htable](#), [Kolist](#), [Module_htable](#), [Pathway_htable](#), [mmu_kegg_pathway](#)

 ko.test

Differential analysis or Correlation analysis for KO-abundance table

Description

Differential analysis or Correlation analysis for KO-abundance table

Usage

```
ko.test(
  kofd,
  group,
  metadata = NULL,
  method = "wilcox.test",
  pattern = NULL,
  p.adjust.method1 = "none",
  threads = 1,
  verbose = TRUE
)
```

Arguments

| | |
|------------------|--|
| kofd | KO_abundance table, rowname are feature ids (e.g. K00001 if feature="ko"; PEX111A if feature="gene"; C00024 if feature="compound"), colnames are samples. |
| group | The comparison groups (at least two categories) in your data, one column name of metadata when metadata exist or a vector whose length equal to columns number of kofd. And you can use factor levels to change order. |
| metadata | sample information data.frame contains group |
| method | the type of test. Default is 'wilcox.test'. Allowed values include: <ul style="list-style-type: none"> • t.test (parametric) and wilcox.test (non-parametric). Perform comparison between two groups of samples. If the grouping variable contains more than two levels, then a pairwise comparison is performed. • anova (parametric) and kruskal.test (non-parametric). Perform one-way ANOVA test comparing multiple groups. • 'pearson', 'kendall', or 'spearman' (correlation), see cor. |
| pattern | a named vector matching the group, e.g. c('G1'=1,'G2'=3,'G3'=2), use the correlation analysis with specific pattern to calculate p-value. |
| p.adjust.method1 | p.adjust.method for 'ko.test', see p.adjust |
| threads | default 1 |
| verbose | logical |

Value

ko_pvalue data.frame

See Also

Other GRSA: [combine_rs_res\(\)](#), [get_reporter_score\(\)](#), [pvalue2zs\(\)](#), [reporter_score\(\)](#)

Examples

```
data("KO_abundance_test")
ko_pvalue <- ko.test(KO_abundance, "Group", metadata)
```

KOlis

The KOlis used for enrichment.

Description

an list contains two data.frame named pathway and module.

Format

four columns in each data.frame.

id "map0010" or "M00001"

K_num contains how many KOs in this pathway or module

KOs KOs name

Description the description of this pathway or module

See Also

Other data: [CPDlist](#), [Compound_htable](#), [Golist](#), [KO_htable](#), [Module_htable](#), [Pathway_htable](#), [hsa_kegg_pathway](#), [mmu_kegg_pathway](#)

KO_abundance

The KOs abundance table and group table.

Description

The KOs abundance table and group table.

The KOs abundance table and group table.

See Also

Other test_data: [genedf](#), [reporter_score_res](#)

KO_enrich

*Perform enrichment analysis***Description**

This function performs KO enrichment analysis using the ‘clusterProfiler’ package.

Usage

```
KO_enrich(
  ko_stat,
  padj_threshold = 0.05,
  logFC_threshold = NULL,
  add_mini = NULL,
  p.adjust.method = "BH",
  type = c("pathway", "module")[1],
  feature = "ko",
  modulelist = NULL,
  verbose = TRUE
)

as.enrich_res(gsea_res)
```

Arguments

| | |
|-----------------|--|
| ko_stat | ko_stat dataframe from ko.test . |
| padj_threshold | p.adjust threshold to determine whether a feature significant or not. p.adjust < padj_threshold, default: 0.05 |
| logFC_threshold | logFC threshold to determine whether a feature significant or not. abs(logFC)>logFC_threshold, default: NULL |
| add_mini | add_mini when calculate the logFC. e.g (10+0.1)/(0+0.1), default 0.05*min(avg_abundance) |
| p.adjust.method | The method used for p-value adjustment (default: "BH"). |
| type | "pathway" or "module" for default Kolist_file. |
| feature | one of "ko", "gene", "compound" |
| modulelist | NULL or customized modulelist dataframe, must contain "id", "K_num", "KOs", "Description" columns. Take the ‘Kolist’ as example, use custom_modulelist . |
| verbose | logical |
| gsea_res | gsea_res from KO_gsea |

Value

A data frame containing the enrichment results.

enrich_res object

See Also

Other common_enrich: [KO_fisher\(\)](#), [KO_gsa\(\)](#), [KO_gsea\(\)](#), [KO_gsva\(\)](#), [KO_padog\(\)](#), [KO_safe\(\)](#), [KO_sea\(\)](#), [plot_enrich_res\(\)](#)

 KO_fisher

Perform fisher's exact enrichment analysis

Description

Perform fisher's exact enrichment analysis

Usage

```
KO_fisher(
  ko_stat,
  padj_threshold = 0.05,
  logFC_threshold = NULL,
  add_mini = NULL,
  p.adjust.method = "BH",
  type = c("pathway", "module")[1],
  feature = "ko",
  modulelist = NULL,
  verbose = TRUE
)
```

Arguments

| | |
|-----------------|--|
| ko_stat | ko_stat dataframe from ko.test . |
| padj_threshold | p.adjust threshold to determine whether a feature significant or not. p.adjust < padj_threshold, default: 0.05 |
| logFC_threshold | logFC threshold to determine whether a feature significant or not. abs(logFC)>logFC_threshold, default: NULL |
| add_mini | add_mini when calculate the logFC. e.g (10+0.1)/(0+0.1), default 0.05*min(avg_abundance) |
| p.adjust.method | The method used for p-value adjustment (default: "BH"). |
| type | "pathway" or "module" for default Kolist_file. |
| feature | one of "ko", "gene", "compound" |
| modulelist | NULL or customized modulelist dataframe, must contain "id", "K_num", "KOs", "Description" columns. Take the 'Kolist' as example, use custom_modulelist . |
| verbose | logical |

Value

data.frame

See Also

Other common_enrich: [KO_enrich\(\)](#), [KO_gsa\(\)](#), [KO_gsea\(\)](#), [KO_gsva\(\)](#), [KO_padog\(\)](#), [KO_safe\(\)](#), [KO_sea\(\)](#), [plot_enrich_res\(\)](#)

Examples

```
## use `fisher.test` from the `stats` package.
data("reporter_score_res")
fisher_res <- KO_fisher(reporter_score_res)
```

| | |
|--------|----------------------------------|
| KO_gsa | <i>Perform gene set analysis</i> |
|--------|----------------------------------|

Description

Perform gene set analysis

Usage

```
KO_gsa(
  reporter_res,
  method = "Two class unpaired",
  p.adjust.method = "BH",
  verbose = TRUE,
  perm = 1000,
  ...
)
```

Arguments

| | |
|-----------------|---|
| reporter_res | reporter_res |
| method | Problem type: "quantitative" for a continuous parameter; "Two class unpaired" ; "Survival" for censored survival outcome; "Multiclass" : more than 2 groups, coded 1,2,3...; "Two class paired" for paired outcomes, coded -1,1 (first pair), -2,2 (second pair), etc |
| p.adjust.method | "BH" |
| verbose | TRUE |
| perm | 1000 |
| ... | additional parameters to GSA |

Value

enrich_res object

See Also

Other common_enrich: [KO_enrich\(\)](#), [KO_fisher\(\)](#), [KO_gsea\(\)](#), [KO_gsva\(\)](#), [KO_padog\(\)](#), [KO_safe\(\)](#), [KO_sea\(\)](#), [plot_enrich_res\(\)](#)

Examples

```
## use `GSA` from the `GSA` package.
if (requireNamespace("GSA")) {
  data("reporter_score_res")
  gsa_res <- KO_gsa(reporter_score_res, p.adjust.method = "none", perm = 200)
  plot(gsa_res)
}
```

 KO_gsea

Perform gene set enrichment analysis

Description

Perform gene set enrichment analysis

Usage

```
KO_gsea(
  ko_stat,
  weight = "logFC",
  add_mini = NULL,
  p.adjust.method = "BH",
  type = c("pathway", "module")[1],
  feature = "ko",
  modulelist = NULL,
  verbose = TRUE
)
```

Arguments

| | |
|-----------------|--|
| ko_stat | ko_stat dataframe from ko.test . |
| weight | the metric used for ranking, default: logFC |
| add_mini | add_mini when calculate the logFC. e.g (10+0.1)/(0+0.1), default 0.05*min(avg_abundance) |
| p.adjust.method | The method used for p-value adjustment (default: "BH"). |
| type | "pathway" or "module" for default KOList_file. |
| feature | one of "ko", "gene", "compound" |
| modulelist | NULL or customized modulelist dataframe, must contain "id", "K_num", "KOs", "Description" columns. Take the 'KOList' as example, use custom_modulelist . |
| verbose | logical |

Value

DOSE object

See Also

Other common_enrich: [KO_enrich\(\)](#), [KO_fisher\(\)](#), [KO_gsa\(\)](#), [KO_gsva\(\)](#), [KO_padog\(\)](#), [KO_safe\(\)](#), [KO_sea\(\)](#), [plot_enrich_res\(\)](#)

Examples

```
message("The following example require some time to run:")

## use `GSEA` from the `clusterProfiler` package.
if (requireNamespace("clusterProfiler")) {
  data("reporter_score_res")
  gsea_res <- KO_gsea(reporter_score_res, p.adjust.method = "none")
  enrichplot::gseaplot(gsea_res, geneSetID = data.frame(gsea_res)$ID[1])
  gsea_res_df <- as.enrich_res(gsea_res)
  plot(gsea_res_df)
}
```

KO_gsva

Perform Gene Set Variation Analysis

Description

Perform Gene Set Variation Analysis

Usage

```
KO_gsva(
  reporter_res,
  verbose = TRUE,
  method = "wilcox.test",
  p.adjust.method = "BH",
  ...
)
```

Arguments

| | |
|-----------------|---|
| reporter_res | reporter_res |
| verbose | verbose |
| method | see ko.test |
| p.adjust.method | p.adjust.method |
| ... | additional parameters to gsva |

Value

enrich_res

See Also

Other common_enrich: [KO_enrich\(\)](#), [KO_fisher\(\)](#), [KO_gsa\(\)](#), [KO_gsea\(\)](#), [KO_padog\(\)](#), [KO_safe\(\)](#), [KO_sea\(\)](#), [plot_enrich_res\(\)](#)

Examples

```
## use `gsva` from the `GSVA` package.
if (requireNamespace("GSVA")) {
  data("reporter_score_res")
  gsva_res <- KO_gsva(reporter_score_res, p.adjust.method = "none")
}
```

| | |
|-----------|------------------------------|
| KO_htable | <i>KO htable from 'KEGG'</i> |
|-----------|------------------------------|

Description

KO htable from 'KEGG'

See Also

Other data: [CPDlist](#), [Compound_htable](#), [GOList](#), [KOList](#), [Module_htable](#), [Pathway_htable](#), [hsa_kegg_pathway](#), [mmu_kegg_pathway](#)

| | |
|----------|--|
| KO_padog | <i>Perform Pathway Analysis with Down-weighting of Overlapping Genes (PADOG)</i> |
|----------|--|

Description

Perform Pathway Analysis with Down-weighting of Overlapping Genes (PADOG)

Usage

```
KO_padog(
  reporter_res,
  verbose = TRUE,
  perm = 1000,
  p.adjust.method = "BH",
  ...
)
```

Arguments

reporter_res The input reporter result.
 verbose If TRUE, print verbose messages. Default is TRUE.
 perm The number of permutations. Default is 1000.
 p.adjust.method Method for p-value adjustment. Default is "BH".
 ... Additional parameters to be passed to [padog](#) function.

Value

A data frame containing PADOG results for KO enrichment.

A data frame with columns "ID," "Description," "K_num," "Exist_K_num," "p.value," and "p.adjust."

See Also

Other common_enrich: [KO_enrich\(\)](#), [KO_fisher\(\)](#), [KO_gsa\(\)](#), [KO_gsea\(\)](#), [KO_gsva\(\)](#), [KO_safe\(\)](#), [KO_sea\(\)](#), [plot_enrich_res\(\)](#)

Examples

```
## use `PADOG` from the `PADOG` package.
if (requireNamespace("PADOG")) {
  data("reporter_score_res")
  padog_res <- KO_padog(reporter_score_res,
    verbose = TRUE,
    perm = 200, p.adjust.method = "none"
  )
}
```

 KO_safe

Perform Significance Analysis of Function and Expression

Description

Perform Significance Analysis of Function and Expression

Usage

```
KO_safe(
  reporter_res,
  verbose = TRUE,
  perm = 1000,
  C.matrix = NULL,
  p.adjust.method = "BH",
  ...
)
```

Arguments

| | |
|-----------------|--|
| reporter_res | The input reporter result. |
| verbose | If TRUE, print verbose messages. Default is TRUE. |
| perm | The number of permutations. Default is 1000. |
| C.matrix | The contrast matrix. Default is NULL, and it will be generated from the module list. |
| p.adjust.method | Method for p-value adjustment. Default is "BH". |
| ... | Additional parameters to be passed to safe function. |

Value

A data frame containing SAFE results for KO enrichment.

See Also

Other common_enrich: [KO_enrich\(\)](#), [KO_fisher\(\)](#), [KO_gsa\(\)](#), [KO_gsea\(\)](#), [KO_gsva\(\)](#), [KO_padog\(\)](#), [KO_sea\(\)](#), [plot_enrich_res\(\)](#)

Examples

```
## use `safe` from the `safe` package.
if (requireNamespace("safe")) {
  data("reporter_score_res")
  safe_res <- KO_safe(reporter_score_res,
    verbose = TRUE,
    perm = 200, p.adjust.method = "none"
  )
}
```

 KO_sea

Perform Simultaneous Enrichment Analysis

Description

Perform Simultaneous Enrichment Analysis

Usage

```
KO_sea(reporter_res, verbose = TRUE, ...)
```

Arguments

| | |
|--------------|---|
| reporter_res | The input reporter result. |
| verbose | If TRUE, print verbose messages. Default is TRUE. |
| ... | Additional parameters to be passed to SEA function. |

Value

enrich_res

See Also

Other common_enrich: [KO_enrich\(\)](#), [KO_fisher\(\)](#), [KO_gsa\(\)](#), [KO_gsea\(\)](#), [KO_gsva\(\)](#), [KO_padog\(\)](#), [KO_safe\(\)](#), [plot_enrich_res\(\)](#)

Examples

```
## use `SEA` from the `rSEA` package.
if (requireNamespace("rSEA")) {
  data("reporter_score_res")
  sea_res <- KO_sea(reporter_score_res, verbose = TRUE)
}
```

load_CARDinfo

Load the CARDinfo (from CARD database)

Description

Load the CARDinfo (from CARD database)

Usage

```
load_CARDinfo(verbose = TRUE)
```

Arguments

verbose logical

Value

CARDinfo

| | |
|-------------|---|
| load_GOlist | <i>Load the GOlist (from 'GO' database)</i> |
|-------------|---|

Description

Load the GOlist (from 'GO' database)

Load the GOinfo (from GO)

Usage

```
load_GOlist(verbose = TRUE)
```

```
load_GOinfo(verbose = TRUE)
```

Arguments

verbose logical

Value

GOlist

GOinfo

| | |
|-------------|--|
| load_htable | <i>Load the specific table (from 'KEGG')</i> |
|-------------|--|

Description

Load the specific table (from 'KEGG')

Load the KOList (from 'KEGG')

Load the CPDlist (from 'KEGG')

Load the KO description (from 'KEGG')

Load the KO_htable (from 'KEGG')

Load the Pathway_htable (from 'KEGG')

Load the Module_htable (from 'KEGG')

Load the Compound_htable (from 'KEGG')

Load the pathway information for an organism (from 'KEGG')

Usage

```
load_htable(type, verbose = TRUE)

load_KOlist(verbose = TRUE)

load_CPDlist(verbose = TRUE)

load_KO_desc(verbose = TRUE)

load_KO_htable(verbose = TRUE)

load_Pathway_htable(verbose = TRUE)

load_Module_htable(verbose = TRUE)

load_Compound_htable(verbose = TRUE)

load_org_pathway(org = "hsa", verbose = TRUE)
```

Arguments

| | |
|---------|---|
| type | "ko", "module", "pathway", "compound" ... |
| verbose | logical |
| org | kegg organism, listed in https://www.genome.jp/kegg/catalog/org_list.html , default, "hsa" |

Value

```
KO_htable
KOlist
CPDlist
KO description
KO_htable
Pathway_htable
Module_htable
Compound_htable
KOlist
```

Examples

```
Pathway_htable <- load_htable("pathway")
head(Pathway_htable)
```

| | |
|------------------|--------------------------------------|
| mmu_kegg_pathway | <i>pathway information for "mmu"</i> |
|------------------|--------------------------------------|

Description

pathway information for "mmu"

See Also

Other data: [CPDlist](#), [Compound_htable](#), [Golist](#), [KO_htable](#), [Kolist](#), [Module_htable](#), [Pathway_htable](#), [hsa_kegg_pathway](#)

| | |
|--------------------|---|
| modify_description | <i>Modify the pathway description before plotting</i> |
|--------------------|---|

Description

Modify the pathway description before plotting

Usage

```
modify_description(  
  reporter_res,  
  pattern = " - Homo sapiens (human)",  
  replacement = ""  
)
```

Arguments

| | |
|--------------|-------------------------------------|
| reporter_res | reporter_res |
| pattern | str, like " - Homo sapiens (human)" |
| replacement | str, like "" |

Value

reporter_res

Examples

```
data("reporter_score_res")  
modify_description(reporter_score_res, pattern = " - Homo sapiens (human)")
```

| | |
|----------------|----------------------------------|
| Module_hhtable | <i>Module htable from 'KEGG'</i> |
|----------------|----------------------------------|

Description

Module htable from 'KEGG'

See Also

Other data: [CPDlist](#), [Compound_hhtable](#), [Golist](#), [KO_hhtable](#), [Kolist](#), [Pathway_hhtable](#), [hsa_kegg_pathway](#), [mmu_kegg_pathway](#)

| | |
|-----------------|-----------------------------------|
| Pathway_hhtable | <i>Pathway htable from 'KEGG'</i> |
|-----------------|-----------------------------------|

Description

Pathway htable from 'KEGG'

See Also

Other data: [CPDlist](#), [Compound_hhtable](#), [Golist](#), [KO_hhtable](#), [Kolist](#), [Module_hhtable](#), [hsa_kegg_pathway](#), [mmu_kegg_pathway](#)

| | |
|-------------|----------------------------|
| plot.cm_res | <i>Plot c_means result</i> |
|-------------|----------------------------|

Description

Plot c_means result

Usage

```
## S3 method for class 'cm_res'  
plot(  
  x,  
  filter_membership,  
  mode = 1,  
  show.clust.cent = TRUE,  
  show_num = TRUE,  
  ...  
)
```

Arguments

x a cm_res object
 filter_membership filter membership
 mode 1~2
 show.clust.cent show cluster center?
 show_num show number of each cluster?
 ... additional

Value

ggplot

| | |
|-----------------|------------------------|
| plot_enrich_res | <i>Plot enrich_res</i> |
|-----------------|------------------------|

Description

Plot enrich_res

Plot enrich_res

Usage

```

plot_enrich_res(
  enrich_res,
  mode = 1,
  padj_threshold = 0.05,
  show_ID = FALSE,
  Pathway_description = TRUE,
  facet_level = FALSE,
  facet_anno = NULL,
  str_width = 50,
  facet_str_width = 15,
  ...
)

## S3 method for class 'enrich_res'
plot(
  x,
  mode = 1,
  padj_threshold = 0.05,
  show_ID = FALSE,
  Pathway_description = TRUE,
  facet_level = FALSE,

```

```

    facet_anno = NULL,
    str_width = 50,
    facet_str_width = 15,
    ...
  )

```

Arguments

| | |
|---------------------|--|
| enrich_res | enrich_res object |
| mode | plot style: 1~2 |
| padj_threshold | p.adjust threshold |
| show_ID | show pathway id |
| Pathway_description | show KO description rather than KO id. |
| facet_level | facet plot if the type is "pathway" or "module" |
| facet_anno | annotation table for facet, two columns, first is level summary, second is pathway id. |
| str_width | default: 50 |
| facet_str_width | str width for facet label |
| ... | add |
| x | enrich_res object |

Value

ggplot
ggplot

See Also

Other common_enrich: [KO_enrich\(\)](#), [KO_fisher\(\)](#), [KO_gsa\(\)](#), [KO_gsea\(\)](#), [KO_gsva\(\)](#), [KO_padog\(\)](#), [KO_safe\(\)](#), [KO_sea\(\)](#)

plot_features_box *Plot features boxplot*

Description

Plot features boxplot

Usage

```
plot_features_box(
  kofd,
  group = NULL,
  metadata = NULL,
  map_id = "map00780",
  select_ko = NULL,
  only_sig = FALSE,
  box_param = NULL,
  modulelist = NULL,
  KO_description = FALSE,
  str_width = 50
)
```

Arguments

| | |
|----------------|--|
| kofd | KO_abundance table, rowname is ko id (e.g. K00001), colnames is samples. or result of 'get_reporter_score' |
| group | The compare group (two category) in your data, one column name of metadata when metadata exist or a vector whose length equal to columns number of kofd. |
| metadata | metadata |
| map_id | the pathway or module id |
| select_ko | select which ko |
| only_sig | only show the significant features |
| box_param | parameters pass to group_box |
| modulelist | NULL or customized modulelist dataframe, must contain "id", "K_num", "KOs", "Description" columns. Take the 'KOlist' as example, use custom_modulelist . |
| KO_description | show KO description rather than KO id. |
| str_width | str_width to wrap |

Value

ggplot

Examples

```
data("reporter_score_res")
plot_features_box(reporter_score_res,
  select_ko = c("K00059", "K00208", "K00647", "K00652", "K00833", "K01012"),
  box_param = list(p_value1 = FALSE, trend_line = TRUE)
)
plot_features_box(reporter_score_res,
  select_ko = "K00059", KO_description = TRUE,
  box_param = list(p_value1 = FALSE, trend_line = TRUE)
)
```

`plot_features_distribution`*plot the Z-score of features distribution*

Description

plot the Z-score of features distribution

Usage

```
plot_features_distribution(  
  reporter_res,  
  map_id,  
  text_size = 4,  
  text_position = NULL,  
  rug_length = 0.04  
)
```

Arguments

| | |
|----------------------------|---|
| <code>reporter_res</code> | result of 'reporter_score' |
| <code>map_id</code> | the pathway or module id |
| <code>text_size</code> | <code>text_size=4</code> |
| <code>text_position</code> | text_position, e.g. <code>c(x=3,y=0.4)</code> |
| <code>rug_length</code> | <code>rug_length=0.04</code> |

Value

ggplot

Examples

```
data("reporter_score_res")  
plot_features_distribution(reporter_score_res, map_id = c("map05230", "map03010"))
```

plot_features_heatmap *Plot features heatmap*

Description

Plot features heatmap

Usage

```
plot_features_heatmap(  
  kodf,  
  group = NULL,  
  metadata = NULL,  
  map_id = "map00780",  
  select_ko = NULL,  
  only_sig = FALSE,  
  columns = NULL,  
  modulelist = NULL,  
  KO_description = FALSE,  
  str_width = 50,  
  heatmap_param = list()  
)
```

Arguments

| | |
|----------------|--|
| kodf | KO_abundance table, rowname is ko id (e.g. K00001), colnames is samples. or result of 'get_reporter_score' |
| group | The compare group (two category) in your data, one column name of metadata when metadata exist or a vector whose length equal to columns number of kodf. |
| metadata | metadata |
| map_id | the pathway or module id |
| select_ko | select which ko |
| only_sig | only show the significant KOs |
| columns | change columns |
| modulelist | NULL or customized modulelist dataframe, must contain "id", "K_num", "KOs", "Description" columns. Take the 'KOlist' as example, use custom_modulelist . |
| KO_description | show KO description rather than KO id. |
| str_width | str_width to wrap |
| heatmap_param | parameters pass to pheatmap |

Value

ggplot

Examples

```
if (requireNamespace("pheatmap")) {
  data("reporter_score_res")
  plot_features_heatmap(reporter_score_res, map_id = "map00780")
}
```

plot_features_in_pathway

Plot features trend in one pathway or module

Description

Plot features trend in one pathway or module

Usage

```
plot_features_in_pathway(
  ko_stat,
  map_id = "map00780",
  modulelist = NULL,
  select_ko = NULL,
  box_color = reporter_color,
  show_number = TRUE,
  scale = FALSE,
  feature_type = "KOs",
  line_color = c(Depleted = "seagreen", Enriched = "orange", None = "grey", Significant =
    "red2")
)
```

Arguments

| | |
|--------------|--|
| ko_stat | ko_stat result from pvalue2zs or result of ‘get_reporter_score’ |
| map_id | the pathway or module id |
| modulelist | NULL or customized modulelist dataframe, must contain "id", "K_num", "KOs", "Description" columns. Take the ‘KOList’ as example, use custom_modulelist . |
| select_ko | select which ko |
| box_color | box and point color, default: c("#e31a1c", "#1f78b4") |
| show_number | show the numbers. |
| scale | scale the data by row. |
| feature_type | show in the title ,default: KOs |
| line_color | line color, default: c("Depleted"="seagreen", "Enriched"="orange", "None"="grey") |

Value

ggplot

Examples

```
data("reporter_score_res")
plot_features_in_pathway(ko_stat = reporter_score_res, map_id = "map00860")
```

plot_features_network *Plot features network*

Description

Plot features network

Usage

```
plot_features_network(
  ko_stat,
  map_id = "map00780",
  near_pathway = FALSE,
  modulelist = NULL,
  kos_color = c(Depleted = "seagreen", Enriched = "orange", None = "grey", Significant =
    "red2", Pathway = "#80b1d3"),
  pathway_label = TRUE,
  kos_label = TRUE,
  pathway_description = FALSE,
  kos_description = FALSE,
  str_width = 50,
  mark_module = FALSE,
  mark_color = NULL,
  return_net = FALSE,
  ...
)
```

Arguments

| | |
|---------------------|---|
| ko_stat | ko_stat result from pvalue2zs or result of 'get_reporter_score' |
| map_id | the pathway or module id |
| near_pathway | show the near_pathway if any features exist. |
| modulelist | NULL or customized modulelist dataframe, must contain "id", "K_num", "KOs", "Description" columns. Take the 'KOlis' as example, use custom_modulelist . |
| kos_color | default, c("Depleted"="seagreen", "Enriched"="orange", "None"="grey", "Significant"="red2") |
| pathway_label | show pathway_label? |
| kos_label | show kos_label? |
| pathway_description | show the pathway description? |

```

kos_description      show the kos description?
str_width            str width
mark_module          mark the modules?
mark_color           mark colors, default, c("Depleted"="seagreen", "Enriched"="orange", "None"="grey", "Significant"="red2
return_net           return the network
...                  additional arguments for c_net_plot

```

Value

network plot

Examples

```

if (requireNamespace("MetaNet")) {
  data("reporter_score_res")
  plot_features_network(reporter_score_res, map_id = "map05230")
  plot_features_network(reporter_score_res, map_id = "map00780", near_pathway = TRUE)
}

```

plot_htable

Plot htable levels

Description

Plot htable levels

Usage

```
plot_htable(type = "ko", select = NULL, htable = NULL)
```

Arguments

```

type          "ko", "module", "pathway", "compound"
select        select ids
htable        custom a htable

```

Value

ggplot

Examples

```

data("KO_abundance_test")
plot_htable(select = rownames(KO_abundance))

```

| | |
|---------------|----------------------|
| plot_KEGG_map | <i>plot_KEGG_map</i> |
|---------------|----------------------|

Description

plot_KEGG_map

Usage

```
plot_KEGG_map(
  ko_stat,
  map_id = "map00780",
  modulelist = NULL,
  type = "pathway",
  feature = "ko",
  color_var = "Z_score",
  save_dir,
  color = c("seagreen", "grey", "orange")
)
```

Arguments

| | |
|------------|---|
| ko_stat | ko_stat result from pvalue2zs or result of 'get_reporter_score' |
| map_id | the pathway or module id |
| modulelist | NULL or customized modulelist dataframe, must contain "id", "K_num", "KOs", "Description" columns. Take the 'KOl原因' as example, use custom_modulelist . |
| type | "pathway" or "module" for default KOl原因 for microbiome, "CC", "MF", "BP", "ALL" for default GOlist for homo sapiens. And org in listed in 'https://www.genome.jp/kegg/catalog/org' such as "hsa" (if your kodf is come from a specific organism, you should specify type here). |
| feature | one of "ko", "gene", "compound" |
| color_var | use which variable to color |
| save_dir | where to save the png files |
| color | color |

Value

png files

References

<https://zhuanlan.zhihu.com/p/357687076>

Examples

```
message("The following example will download some files, run yourself:")

if (requireNamespace("pathview")) {
  output_dir <- tempdir()
  data("reporter_score_res")
  plot_KEGG_map(reporter_score_res$ko_stat,
    map_id = "map00780", type = "pathway",
    feature = "ko", color_var = "Z_score", save_dir = output_dir
  )
}
```

| | |
|-------------|------------------------------|
| plot_report | <i>Plot the reporter_res</i> |
|-------------|------------------------------|

Description

Plot the reporter_res

Usage

```
plot_report(
  reporter_res,
  rs_threshold = 1.64,
  mode = 1,
  y_text_size = 13,
  str_width = 100,
  show_ID = FALSE,
  Pathway_description = TRUE,
  facet_level = FALSE,
  facet_anno = NULL,
  facet_str_width = 15
)
```

Arguments

| | |
|---------------------|--|
| reporter_res | result of ‘get_reporter_score’ or ‘reporter_score’ |
| rs_threshold | plot threshold vector, default:1.64 |
| mode | 1~2 plot style. |
| y_text_size | y_text_size |
| str_width | str_width to wrap |
| show_ID | show pathway id |
| Pathway_description | show KO description rather than KO id. |

| | |
|-----------------|--|
| facet_level | facet plot if the type is "pathway" or "module" |
| facet_anno | annotation table for facet, two columns, first is level summary, second is pathway id. |
| facet_str_width | str width for facet label |

Value

ggplot

Examples

```
data("reporter_score_res")
plot_report(reporter_score_res, rs_threshold = c(2.5, -2.5), y_text_size = 10, str_width = 40)
```

plot_report_circle_packing

Plot the reporter_res as circle_packing

Description

Plot the reporter_res as circle_packing

Usage

```
plot_report_circle_packing(
  reporter_res,
  rs_threshold = 1.64,
  mode = 2,
  facet_anno = NULL,
  show_ID = FALSE,
  Pathway_description = TRUE,
  str_width = 10,
  show_level_name = "all",
  show_tip_label = TRUE
)
```

Arguments

| | |
|--------------|--|
| reporter_res | result of 'get_reporter_score' |
| rs_threshold | plot threshold vector, default:1.64 |
| mode | 1~2 plot style. |
| facet_anno | annotation table for facet, more two columns, last is pathway name, last second is pathway id. |
| show_ID | show pathway id |

```

Pathway_description      show KO description rather than KO id.
str_width                str_width to wrap
show_level_name         show the level name?
show_tip_label          show the tip label?

```

Value

```
ggplot
```

Examples

```

data("reporter_score_res")
if (requireNamespace("igraph") && requireNamespace("ggraph")) {
  plot_report_circle_packing(reporter_score_res, rs_threshold = c(2, -2), str_width = 40)
}

```

```
plot_significance      Plot the significance of pathway
```

Description

Plot the significance of pathway

Usage

```
plot_significance(reporter_res, map_id)
```

Arguments

```

reporter_res    result of 'get_reporter_score' or 'reporter_score'
map_id          the pathway or module id

```

Value

```
ggplot
```

Examples

```

data("reporter_score_res")
plot_significance(reporter_score_res, map_id = c("map05230", "map03010"))

```

print.reporter_score *Print reporter_score*

Description

Print reporter_score

Usage

```
## S3 method for class 'reporter_score'  
print(x, ...)
```

Arguments

| | |
|-----|----------------|
| x | reporter_score |
| ... | add |

Value

No value

print.rs_by_cm *Print rs_by_cm*

Description

Print rs_by_cm

Usage

```
## S3 method for class 'rs_by_cm'  
print(x, ...)
```

Arguments

| | |
|-----|----------|
| x | rs_by_cm |
| ... | add |

Value

No value

pvalue2zs

*Transfer p-value of KOs to Z-score***Description**

Transfer p-value of KOs to Z-score

Usage

```
pvalue2zs(
  ko_pvalue,
  mode = c("directed", "mixed")[1],
  p.adjust.method1 = "none"
)
```

Arguments

`ko_pvalue` data.frame from [ko.test](#), 'KO_id' and 'p.value' columns are required.

`mode` 'mixed' or 'directed' (default, only for two groups differential analysis or multi-groups correlation analysis.), see details in [pvalue2zs](#).

`p.adjust.method1` p.adjust.method for 'ko.test', see [p.adjust](#)

Details

'**mixed**' mode is the original reporter-score method from Patil, K. R. et al. PNAS 2005. In this mode, the reporter score is **undirected**, and the larger the reporter score, the more significant the enrichment, but it cannot indicate the up-and-down regulation information of the pathway! (Liu, L. et al. iMeta 2023.)

steps:

1. Use the Wilcoxon rank sum test to obtain the P value of the significance of each KO difference between the two groups (ie P_{koi} , i represents a certain KO);
2. Using an inverse normal distribution, convert the P value of each KO into a Z value (Z_{koi}), the formula:

$$Z_{koi} = \theta^{-1}(1 - P_{koi})$$

3. 'Upgrade' KO to pathway: Z_{koi} , calculate the Z value of the pathway, the formula:

$$Z_{pathway} = \frac{1}{\sqrt{k}} \sum Z_{koi}$$

where k means A total of k KOs were annotated to the corresponding pathway;

4. Evaluate the degree of significance: permutation (permutation) 1000 times, get the random distribution of $Z_{pathway}$, the formula:

$$Z_{adjustedpathway} = (Z_{pathway} - \mu_k) / \sigma_k$$

μ_k is The mean of the random distribution, σ_k is the standard deviation of the random distribution.

Instead, '**directed**' mode is a derived version of 'mixed', referenced from <https://github.com/wangpeng407/ReporterSc>

This approach is based on the same assumption of many differential analysis methods: the expression of most genes has no significant change.

steps:

1. Use the Wilcoxon rank sum test to obtain the P value of the significance of each KO difference between the two groups (ie P_{koi} , i represents a certain KO), and then divide the P value by 2, that is, the range of (0,1] becomes (0,0.5], $P_{koi} = P_{koi}/2$;

2. Using an inverse normal distribution, convert the P value of each KO into a Z value (Z_{koi}), the formula:

$$Z_{koi} = \theta^{-1}(1 - P_{koi})$$

since the above P value is less than 0.5, all Z values will be greater than 0;

3. Considering whether each KO is up-regulated or down-regulated, calculate *diff_KO*,

$$Z_{koi} = -Z_{koi} \quad (diff_KO < 0),$$

so Z_{koi} is greater than 0 Up-regulation, Z_{koi} less than 0 is down-regulation;

4. 'Upgrade' KO to pathway: Z_{koi} , calculate the Z value of the pathway, the formula:

$$Z_{pathway} = \frac{1}{\sqrt{k}} \sum Z_{koi}$$

where k means A total of k KOs were annotated to the corresponding pathway;

5. Evaluate the degree of significance: permutation (permutation) 1000 times, get the random distribution of $Z_{pathway}$, the formula:

$$Z_{adjustedpathway} = (Z_{pathway} - \mu_k) / \sigma_k$$

μ_k is The mean of the random distribution, σ_k is the standard deviation of the random distribution.

The finally obtained $Z_{adjustedpathway}$ is the Reporter score value enriched for each pathway. In this mode, the Reporter score is directed, and a larger positive value represents a significant up-regulation enrichment, and a smaller negative values represent significant down-regulation enrichment.

However, the disadvantage of this mode is that when a pathway contains about the same number of significantly up-regulates KOs and significantly down-regulates KOs, the final absolute value of Reporter score may approach 0, becoming a pathway that has not been significantly enriched.

Value

ko_stat data.frame

References

- Patil, K. R. & Nielsen, J. Uncovering transcriptional regulation of metabolism by using metabolic network topology. *Proc Natl Acad Sci U S A* 102, 2685–2689 (2005).
- Liu, L., Zhu, R. & Wu, D. Misuse of reporter score in microbial enrichment analysis. *iMeta* n/a, e95.
- <https://github.com/wangpeng407/Reporter>

See Also

Other GRSA: [combine_rs_res\(\)](#), [get_reporter_score\(\)](#), [ko.test\(\)](#), [reporter_score\(\)](#)

Examples

```
data("KO_abundance_test")
ko_pvalue <- ko.test(KO_abundance, "Group", metadata)
ko_stat <- pvalue2zs(ko_pvalue, mode = "directed")
```

reporter_score *One step to get the reporter score of your KO abundance table.*

Description

One step to get the reporter score of your KO abundance table.

Usage

```
reporter_score(
  kodf,
  group,
  metadata = NULL,
  method = "wilcox.test",
  pattern = NULL,
  p.adjust.method1 = "none",
  mode = c("directed", "mixed")[1],
  verbose = TRUE,
  feature = "ko",
  type = c("pathway", "module")[1],
  p.adjust.method2 = "BH",
  modulelist = NULL,
  threads = 1,
  perm = 4999,
  min_exist_KO = 3,
  max_exist_KO = 600
)
```

Arguments

| | |
|----------|--|
| kodf | KO_abundance table, rowname are feature ids (e.g. K00001 if feature="ko"; PEX11A if feature="gene"; C00024 if feature="compound"), colnames are samples. |
| group | The comparison groups (at least two categories) in your data, one column name of metadata when metadata exist or a vector whose length equal to columns number of kodf. And you can use factor levels to change order. |
| metadata | sample information data.frame contains group |
| method | the type of test. Default is 'wilcox.test'. Allowed values include: |

- `t.test` (parametric) and `wilcox.test` (non-parametric). Perform comparison between two groups of samples. If the grouping variable contains more than two levels, then a pairwise comparison is performed.
- `anova` (parametric) and `kruskal.test` (non-parametric). Perform one-way ANOVA test comparing multiple groups.
- 'pearson', 'kendall', or 'spearman' (correlation), see `cor`.

`pattern` a named vector matching the group, e.g. `c('G1'=1,'G2'=3,'G3'=2)`, use the correlation analysis with specific pattern to calculate p-value.

`p.adjust.method1` `p.adjust.method` for 'ko.test', see `p.adjust`

`mode` 'mixed' or 'directed' (default, only for two groups differential analysis or multi-groups correlation analysis.), see details in `pvalue2zs`.

`verbose` logical

`feature` one of 'ko', 'gene', 'compound'

`type` 'pathway' or 'module' for default KOList for microbiome, 'CC', 'MF', 'BP', 'ALL' for default GOList for homo sapiens. And org in listed in 'https://www.genome.jp/kegg/catalog/org_ such as 'hsa' (if your kodf is come from a specific organism, you should specify type here).

`p.adjust.method2` `p.adjust.method` for the correction of ReporterScore, see `p.adjust`

`modulelist` NULL or customized modulelist dataframe, must contain 'id', 'K_num', 'KOs', 'Description' columns. Take the 'KOList' as example, use `custom_modulelist`.

`threads` default 1

`perm` permutation number, default: 4999.

`min_exist_KO` min exist KO number in a pathway (default, 3, when a pathway contains KOs less than 3, there will be no RS)

`max_exist_KO` max exist KO number in a pathway (default, 600, when a pathway contains KOs more than 600, there will be no RS)

Value

`reporter_score` object:

| | |
|-------------------------|---|
| <code>kodf</code> | your input KO_abundance table |
| <code>ko_stat</code> | ko statistics result contains p.value and z_score |
| <code>reporter_s</code> | the reporter score in each pathway |
| <code>modulelist</code> | default KOList or customized modulelist dataframe |
| <code>group</code> | The comparison groups in your data |
| <code>metadata</code> | sample information dataframe contains group |

for the 'reporter_s' in result, whose columns represent:

| | |
|-------------|---------------------|
| ID | pathway id |
| Description | pathway description |

| | |
|-------------------|--|
| K_num | total number of KOs/genes in the pathway |
| Exist_K_num | number of KOs/genes in your inputdata that exist in the pathway |
| Significant_K_num | number of kos/genes in your inputdata that are significant in the pathway |
| Z_score | $Z_{pathway} = \frac{1}{\sqrt{k}} \sum Z_{koi}$ |
| BG_Mean | Background mean, μ_k |
| BG_Sd | Background standard deviation, σ_k |
| ReporterScore | ReporterScore of the pathway, $ReporterScore = (Z_{pathway} - \mu_k) / \sigma_k$ |
| p.value | p.value of the ReporterScore |
| p.adjust | adjusted p.value by p.adjust.method2 |

See Also

Other GRSA: [combine_rs_res\(\)](#), [get_reporter_score\(\)](#), [ko.test\(\)](#), [pvalue2zs\(\)](#)

Examples

```
message("The following example require some time to run:")

data("KO_abundance_test")
reporter_score_res <- reporter_score(KO_abundance, "Group", metadata,
  mode = "directed", perm = 499
)
head(reporter_score_res$reporter_s)
reporter_score_res2 <- reporter_score(KO_abundance, "Group2", metadata,
  mode = "mixed",
  method = "kruskal.test", p.adjust.method1 = "none", perm = 499
)
reporter_score_res3 <- reporter_score(KO_abundance, "Group2", metadata,
  mode = "directed",
  method = "pearson", pattern = c("G1" = 1, "G2" = 3, "G3" = 2), perm = 499
)
```

reporter_score_res *'reporter_score()' result from KO_abundance_test*

Description

'reporter_score()' result from KO_abundance_test

'reporter_score()' result from KO_abundance_test

Format

a list contain 7 elements.

kodf your input KO_abundance table

ko_stat ko statistics result contains p.value and z_score

reporter_s the reporter score in each pathway

modulelist default Kolist or customized modulelist dataframe

group The compare group (two category) in your data

metadata sample information dataframe contains group

See Also

Other test_data: [KO_abundance](#), [genedf](#)

RSA_by_cm

Reporter score analysis after C-means clustering

Description

Reporter score analysis after C-means clustering

Extract one cluster from rs_by_cm object

Plot c_means result

Usage

```
RSA_by_cm(  
  kodf,  
  group,  
  metadata = NULL,  
  k_num = NULL,  
  filter_var = 0.7,  
  verbose = TRUE,  
  method = "pearson",  
  ...  
)  
  
extract_cluster(rsa_cm_res, cluster = 1)  
  
plot_c_means(  
  rsa_cm_res,  
  filter_membership,  
  mode = 1,  
  show.clust.cent = TRUE,  
  show_num = TRUE,  
  ...  
)
```

Arguments

| | |
|-------------------|--|
| kodf | KO_abundance table, rowname is ko id (e.g. K00001), colnames is samples. |
| group | The comparison groups (at least two categories) in your data, one column name of metadata when metadata exist or a vector whose length equal to columns number of kodf. And you can use factor levels to change order. |
| metadata | sample information data.frame contains group |
| k_num | if NULL, perform the cm_test_k, else an integer |
| filter_var | see c_means |
| verbose | verbose |
| method | method from reporter_score |
| ... | additional |
| rsa_cm_res | a cm_res object |
| cluster | integer |
| filter_membership | filter membership 0~1. |
| mode | 1~2 |
| show.clust.cent | show cluster center? |
| show_num | show number of each cluster? |

Value

rs_by_cm
 reporter_score object
 ggplot

See Also

Other C_means: [cm_test_k\(\)](#)

Examples

```
message("The following example require some time to run:")

if (requireNamespace("e1071") && requireNamespace("factoextra")) {
  data("KO_abundance_test")
  rsa_cm_res <- RSA_by_cm(KO_abundance, "Group2", metadata,
    k_num = 3,
    filter_var = 0.7, method = "pearson", perm = 199
  )
  extract_cluster(rsa_cm_res, cluster = 1)
}
```

| | |
|-----------------|--|
| update_CARDinfo | <i>update CARDinfo from (from 'CARD' database)</i> |
|-----------------|--|

Description

update CARDinfo from (from 'CARD' database)

Usage

```
update_CARDinfo(download_dir = NULL, card_data = NULL)
```

Arguments

| | |
|--------------|--|
| download_dir | download_dir |
| card_data | card_data from https://card.mcmaster.ca/download/0/broadstreet-v3.2.8.tar.bz2 |

Value

No value

| | |
|---------------|--|
| update_GOlist | <i>Update the GO2gene files (from 'GO' database)</i> |
|---------------|--|

Description

Download links: <http://geneontology.org/docs/download-ontology/> <https://asa12138.github.io/FileList/GO>

Usage

```
update_GOlist(download_dir = NULL, GO_file = NULL)
```

```
update_GOinfo(download_dir = NULL, obo_file = NULL)
```

Arguments

| | |
|--------------|---|
| download_dir | download_dir |
| GO_file | GO_file |
| obo_file | obo_file from http://current.geneontology.org/ontology/go.obo |

Value

No value

 update_KEGG

Update files from 'KEGG'

Description

Download links:

<https://rest.kegg.jp/list/pathway> <https://rest.kegg.jp/link/pathway/ko> <https://rest.kegg.jp/link/pathway/hsa>

<https://rest.kegg.jp/list/module> <https://rest.kegg.jp/link/module/ko> <https://rest.kegg.jp/link/module/hsa>

Usage

```
update_KEGG(download_dir)
```

```
update_KO_file(download_dir, RDSfile = NULL)
```

```
update_htable(type, keg_file = NULL, download = FALSE, download_dir = NULL)
```

```
update_org_pathway(
  org = "hsa",
  RDS_file = NULL,
  download = TRUE,
  download_dir = NULL
)
```

Arguments

| | |
|--------------|---|
| download_dir | where to save the .keg file? |
| RDSfile | saved KO_files.RDS file |
| type | "ko", "module", "pathway", "compound" ... |
| keg_file | path of a .keg file, such as ko00001.keg from https://www.genome.jp/kegg-bin/download_htext?htext=ko00001&format=htext . |
| download | save the .keg file? |
| org | kegg organism, listed in https://www.genome.jp/kegg/catalog/org_list.html , default, "hsa" |
| RDS_file | path of a org.RDS file if you saved before. |

Value

No value

| | |
|-------------|-----------------------------|
| up_level_KO | <i>Upgrade the KO level</i> |
|-------------|-----------------------------|

Description

Upgrade the KO level

Usage

```
up_level_KO(  
  KO_abundance,  
  level = "pathway",  
  show_name = FALSE,  
  modulelist = NULL,  
  verbose = TRUE  
)
```

Arguments

| | |
|--------------|--|
| KO_abundance | KO_abundance |
| level | one of 'pathway', 'module', 'level1', 'level2', 'level3', 'module1', 'module2', 'module3'. |
| show_name | logical |
| modulelist | NULL or customized modulelist dataframe, must contain 'id', 'K_num', 'KOs', 'Description' columns. Take the 'KOlist' as example, use custom_modulelist . |
| verbose | logical |

Value

data.frame

Examples

```
data("KO_abundance_test")  
KO_level1 <- up_level_KO(KO_abundance, level = "level1", show_name = TRUE)
```

Index

- * **C_means**
 - cm_test_k, 3
 - RSA_by_cm, 45
- * **GRSA**
 - combine_rs_res, 4
 - get_reporter_score, 10
 - ko.test, 12
 - pvalue2zs, 40
 - reporter_score, 42
- * **common_enrich**
 - KO_enrich, 14
 - KO_fisher, 15
 - KO_gsa, 16
 - KO_gsea, 17
 - KO_gsva, 18
 - KO_padog, 19
 - KO_safe, 20
 - KO_sea, 21
 - plot_enrich_res, 27
- * **data**
 - Compound_htable, 5
 - CPDlist, 5
 - GOList, 11
 - hsa_kegg_pathway, 11
 - KO_htable, 19
 - KOList, 13
 - mmu_kegg_pathway, 25
 - Module_htable, 26
 - Pathway_htable, 26
- * **modulelist**
 - custom_modulelist, 6
 - custom_modulelist_from_org, 7
 - get_features, 9
- * **test_data**
 - genedf, 9
 - KO_abundance, 13
 - reporter_score_res, 44
- anova, 12, 43
- as.enrich_res (KO_enrich), 14
- c_means (cm_test_k), 3
- c_net_plot, 34
- cm_test_k, 3, 46
- combine_rs_res, 4, 11, 13, 41, 44
- Compound_htable, 5, 5, 11, 13, 19, 25, 26
- cor, 12, 43
- CPDlist, 5, 5, 11, 13, 19, 25, 26
- custom_modulelist, 4, 6, 7, 9, 10, 14, 15, 17, 29, 31–33, 35, 43, 49
- custom_modulelist_from_org, 6, 7, 9
- download_org_pathway (update_KEGG), 48
- export_report_table, 8
- extract_cluster (RSA_by_cm), 45
- gene2ko, 8
- genedf, 9, 13, 45
- get_features, 6, 7, 9
- get_KOs (get_features), 9
- get_org_pathway (update_KEGG), 48
- get_reporter_score, 4, 10, 13, 41, 44
- GOList, 5, 11, 11, 13, 19, 25, 26
- group_box, 29
- GRSA (reporter_score), 42
- GRSA_by_cm (RSA_by_cm), 45
- GSA, 16
- gsva, 18
- hsa_kegg_pathway, 5, 11, 11, 13, 19, 25, 26
- ko.test, 4, 11, 12, 14, 15, 17, 18, 40, 41, 44
- KO_abundance, 9, 13, 45
- KO_enrich, 14, 16–22, 28
- KO_fisher, 15, 15, 17–22, 28
- KO_gsa, 15, 16, 16, 18–22, 28
- KO_gsea, 15–17, 17, 19–22, 28
- KO_gsva, 15–18, 18, 20–22, 28
- KO_htable, 5, 11, 13, 19, 25, 26
- KO_padog, 15–19, 19, 21, 22, 28
- KO_safe, 15–20, 20, 22, 28

- KO_sea, [15–21](#), [21](#), [28](#)
- KOlist, [5](#), [11](#), [13](#), [19](#), [25](#), [26](#)
- kruskal.test, [12](#), [43](#)

- load_CARDinfo, [22](#)
- load_Compound_htable (load_htable), [23](#)
- load_CPDlist (load_htable), [23](#)
- load_GOinfo (load_GOlist), [23](#)
- load_GOlist, [23](#)
- load_htable, [23](#)
- load_KO_desc (load_htable), [23](#)
- load_KO_htable (load_htable), [23](#)
- load_KOlist (load_htable), [23](#)
- load_Module_htable (load_htable), [23](#)
- load_org_pathway (load_htable), [23](#)
- load_Pathway_htable (load_htable), [23](#)

- metadata (KO_abundance), [13](#)
- mmu_kegg_pathway, [5](#), [11](#), [13](#), [19](#), [25](#), [26](#)
- modify_description, [25](#)
- Module_htable, [5](#), [11](#), [13](#), [19](#), [25](#), [26](#), [26](#)

- p.adjust, [10](#), [12](#), [40](#), [43](#)
- padog, [20](#)
- Pathway_htable, [5](#), [11](#), [13](#), [19](#), [25](#), [26](#), [26](#)
- pheatmap, [31](#)
- plot.cm_res, [26](#)
- plot.enrich_res (plot_enrich_res), [27](#)
- plot_c_means (RSA_by_cm), [45](#)
- plot_enrich_res, [15–22](#), [27](#)
- plot_features_box, [28](#)
- plot_features_distribution, [30](#)
- plot_features_heatmap, [31](#)
- plot_features_in_pathway, [32](#)
- plot_features_network, [33](#)
- plot_htable, [34](#)
- plot_KEGG_map, [35](#)
- plot_KOs_box (plot_features_box), [28](#)
- plot_KOs_distribution
 (plot_features_distribution),
 [30](#)
- plot_KOs_heatmap
 (plot_features_heatmap), [31](#)
- plot_KOs_in_pathway
 (plot_features_in_pathway), [32](#)
- plot_KOs_network
 (plot_features_network), [33](#)
- plot_report, [36](#)
- plot_report_bar (plot_report), [36](#)

- plot_report_circle_packing, [37](#)
- plot_significance, [38](#)
- print.reporter_score, [39](#)
- print.rs_by_cm, [39](#)
- pvalue2zs, [4](#), [9–11](#), [13](#), [32](#), [33](#), [35](#), [40](#), [40](#), [43](#),
 [44](#)

- reporter_score, [4](#), [11](#), [13](#), [41](#), [42](#), [46](#)
- reporter_score_res, [9](#), [13](#), [44](#)
- reporter_score_res2
 (reporter_score_res), [44](#)
- RSA (reporter_score), [42](#)
- RSA_by_cm, [3](#), [45](#)

- safe, [21](#)
- SEA, [21](#)

- t.test, [12](#), [43](#)
- transform_modulelist
 (custom_modulelist), [6](#)

- up_level_KO, [49](#)
- update_CARDinfo, [47](#)
- update_GOinfo (update_GOlist), [47](#)
- update_GOlist, [47](#)
- update_htable (update_KEGG), [48](#)
- update_KEGG, [48](#)
- update_KO_file (update_KEGG), [48](#)
- update_org_pathway (update_KEGG), [48](#)

- wilcox.test, [12](#), [43](#)