Package: richCluster (via r-universe)

October 31, 2026

Type Package

Title Fast, Robust Clustering Algorithms for Gene Enrichment Data

Version 2.0.1

Date 2026-09-18

Maintainer Junguk Hur <hurlabshared@gmail.com>

Description Clusters functionally related biological terms from gene set enrichment results. Terms are compared by the overlap of their gene sets using Cohen's kappa, the Jaccard index, or the Dice coefficient, and the resulting similarity matrix is grouped either by agglomerative hierarchical clustering with single, complete, average, or Ward linkage, or by the seed-and-merge procedure of the 'DAVID' functional classification tool. The distance and clustering routines are written in 'C++' for speed. The methods are described in Huang et al. (2007) <doi:10.1186/gb-2007-8-9-r183>, Ward (1963) <doi:10.1080/01621459.1963.10500845>, Cohen (1960) <doi:10.1177/001316446002000104>, and Jaccard (1912) <doi:10.1111/j.1469-8137.1912.tb05611.x>.

License GPL-3

Depends R (>= 3.5.0)

Imports dplyr, fields, heatmaply, igraph, magrittr, networkD3, plotly, Rcpp (>= 1.0.14), stats, tidyr, viridis

Suggests devtools, knitr, rmarkdown, roxygen2, testthat

LinkingTo Rcpp

VignetteBuilder knitr

Encoding UTF-8

URL https://github.com/hurlab/richCluster

BugReports https://github.com/hurlab/richCluster/issues

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Junguk Hur [aut, cre] (ORCID:
<https://orcid.org/0000-0002-0736-2149>), Sarah Hong [aut],
Jane Kim [aut]

Config/pak/sysreqs
cmake libglpk-dev make libmagick++-dev gsfonts libicu-dev libuv1-dev libxml2-dev libssl-dev

Repository https://hurlab.r-universe.dev

Date/Publication 2026-09-30 22:21:18 UTC

RemoteUrl https://github.com/hurlab/richcluster

RemoteRef HEAD

RemoteSha a0d1f630fbe64089ab38516116b78962672e6349

Contents

cluster . . . 2
cluster_bar . . . 5
cluster_correlation_hmap . . . 6
cluster_dot . . . 7
cluster_hmap . . . 7
cluster_network . . . 9
compare_network_graphs_plotly . . . 9
david_cluster . . . 10
export_df . . . 11
filter_clusters . . . 12
format_colnames . . . 13
full_network . . . 13
merge_enrichment_results . . . 14
plot_network_graph . . . 15
runRichCluster . . . 16
term_bar . . . 17
term_dot . . . 18
term_hmap . . . 19

Index **21**

`cluster` *Cluster Terms from Enrichment Results*

Description

This function performs clustering on enrichment results by integrating gene similarity scores and various clustering strategies.

Usage

```
cluster(
  enrichment_results,
  df_names = NULL,
  min_terms = 5,
  min_value = 0.1,
  distance_metric = "kappa",
  distance_cutoff = 0.5,
  linkage_method = "average",
  linkage_cutoff = 0.5,
  filter_on = "Padj",
  gene_delim = ",",
  verbose = FALSE
)
```

Arguments

enrichment_results	A list of dataframes, each containing enrichment results. Each dataframe should include at least the columns 'Term', 'GeneID', and 'Padj'.
df_names	Optional, a character vector of names for the enrichment result dataframes. Must match the length of `enrichment_results`. Default is `NULL`.
min_terms	Minimum number of terms each final cluster must include
min_value	Upper bound on the significance value: a term is kept when its `filter_on` column is strictly LESS than `min_value`. Despite the name this is a maximum, not a minimum; the name is retained from 1.0.2 so that existing calls keep working. Default is `0.1`.
distance_metric	A string specifying the distance metric to use. Supported options are "kappa", "jaccard", and "dice".
distance_cutoff	A numeric value for the distance cutoff (0 < cutoff <= 1). The comparison is STRICT: two terms are linked when their similarity is greater than `distance_cutoff`, not when it equals it.
linkage_method	A string specifying the linkage method to use (e.g., "average"). Supported options are "single", "complete", "average", and "ward".
linkage_cutoff	A numeric value between 0 and 1 for the membership cutoff. The comparison is STRICT: clusters merge when their linkage score is greater than `linkage_cutoff`, not when it equals it.
filter_on	Name of the column terms are selected on, compared against `min_value`. Default is `"Padj"`; `"Pvalue"` reproduces 1.0.2 behaviour. When the named column is absent but 'Pvalue' is present, `cluster()` warns (condition class `richCluster_filter_on_fallback`) and falls back to 'Pvalue' rather than erroring.
gene_delim	A single string separating gene identifiers within the 'GeneID' column. Default is `","`, which reproduces 1.0.2 behaviour exactly. Set it when your gene lists use another separator (for example `";"`); it is applied both to the per-term gene split and to the distinct-gene universe count, which must agree.

verbose — Logical; print the C++ core's progress narration to the console. Default FALSE — the core runs silent. Before 2.0.0 this narration was unconditional and could not be switched off, and it scales with the data (one line per merge iteration), so it is now opt-in.

Value

A named list of eight elements, in this order:

distance_matrix A numeric n x n matrix of pairwise SIMILARITY scores between the n terms that survived the min_value filter. Values lie on [0, 1] and LARGER means MORE similar; the diagonal is 1. For "kappa", a negative value is set to 0, so the matrix holds a non-negative kappa similarity rather than an unmodified Cohen's kappa. Row and column names are the terms, in merged_df row order. Despite the element name it holds similarities, not distances; the name is retained so that existing code keeps working.

all_clusters A data frame of every merged cluster, BEFORE the min_terms filter, with three columns: Cluster (integer cluster ID), TermNames (the cluster's term names, comma-separated) and TermIndices (the same terms as ZERO-based row indices into merged_df, comma-separated).

df_list The enrichment_results argument, unmodified.

merged_df The merged and filtered data frame the clustering ran on: one row per surviving term, carrying Term, the per-dataset columns suffixed _1, _2, ... , and the pooled GeneID, Pvalue, Padj and DatasetCount columns. Its row order is the index basis of TermIndices.

cluster_options A named list of the parameters clustering ran with: min_terms, min_value, distance_metric, distance_cutoff, linkage_method, linkage_cutoff and filter_on.

df_names A character vector naming the input data frames. Always present: when df_names is NULL or does not match the length of enrichment_results it is replaced with "1", "2", ...

final_clusters all_clusters with clusters of fewer than min_terms terms dropped and Cluster renumbered 1..k. TWO columns only — Cluster and TermIndices; TermNames is not carried over.

cluster_df The main output, and the frame the plotting functions (cluster_hmap(), cluster_dot(), cluster_bar(), cluster_network()) consume: one row per (cluster, term) pair, being a Cluster column followed by that term's row from merged_df.

Determinism

Clustering results depend only on the terms and gene sets supplied, not on the order in which rows are given: cluster() and runRichCluster() canonicalise row order internally, so re-sorting the input cannot change cluster membership. Note that the merge stage is a greedy agglomeration — clusters are built by repeatedly merging the best-scoring available pair — so results reflect that greedy strategy rather than a global optimum, and small changes to linkage_cutoff can change membership substantially.

Examples

```
# A small two-dataset input.  The shipped data is far larger, and an example
# that runs during R CMD check should stay quick.
terms <- sprintf("TERM_%02d", 1:12)
```

```
genes <- vapply(1:12, function(i) paste0("G", i:(i + 4), collapse = ","),
                character(1))
d1 <- data.frame(Term = terms, GeneID = genes,
                 Pvalue = 1e-5, Padj = 1e-4, stringsAsFactors = FALSE)
d2 <- d1
res <- cluster(list(d1, d2), min_terms = 2, distance_metric = "kappa")
head(res$cluster_df[, c("Cluster", "Term")])
```

`cluster_bar` *Cluster-level Bar Plot of Enrichment Significance*

Description

Generates a horizontal bar plot showing average enrichment significance for each cluster, across one or more enrichment datasets.

Usage

```
cluster_bar(cluster_result, clusters = NULL, value_type = "Padj", title = NULL)
```

Arguments

`cluster_result`	A result list returned by `cluster`.
`clusters`	Optional numeric vector of cluster IDs to include. Defaults to all clusters.
`value_type`	The column name to use for enrichment significance ("Padj" or "Pvalue").
`title`	Optional plot title. If NULL, a default will be generated.

Value

A `plotly` object representing the bar plot.

Examples

```
# Load example data
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
cbar <- cluster_bar(cluster_result)
cbar
```

`cluster_correlation_hmap` *Create a Correlation Heatmap for a Specific Cluster*

Description

This function generates a correlation heatmap for a specific cluster based on the provided distance matrix.

Usage

```
cluster_correlation_hmap(
  final_clusters,
  distance_matrix,
  cluster_number,
  merged_df
)
```

Arguments

`final_clusters`	A dataframe containing the final cluster data.
`distance_matrix`	A matrix representing the distances between terms.
`cluster_number`	An integer cluster id, as found in `final_clusters$Cluster`.
`merged_df`	A dataframe with all terms used to map term indices to names.

Value

An interactive heatmaply heatmap.

Examples

```
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
chmap <- cluster_correlation_hmap(cluster_result$final_clusters,
                                  cluster_result$distance_matrix,
                                  cluster_number = 1,
                                  cluster_result$merged_df)
chmap
```

cluster_dot *Cluster-level Dot Plot of Enrichment Significance*

Description

Creates a dot plot summarizing cluster-level enrichment across datasets. Each point represents a cluster, with its size proportional to the number of terms and its x-position reflecting average significance (e.g., Padj or Pvalue).

Usage

```
cluster_dot(cluster_result, clusters = NULL, value_type = "Padj", title = NULL)
```

Arguments

cluster_result	A result list returned from cluster.
clusters	Optional numeric vector of cluster IDs to include. Defaults to all clusters.
value_type	The name of the value column to visualize (e.g., "Padj" or "Pvalue").
title	Optional title for the plot. If NULL, a default title is generated.

Value

A plotly object representing the dot plot.

Examples

```
# Load example data
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
cdot <- cluster_dot(cluster_result)
cdot
```

cluster_hmap *Create a Heatmap of Clustered Enrichment Results*

Description

Generates an interactive heatmap from the given clustering results, visualizing -log10(Padj) values for each cluster. The function aggregates values per cluster and assigns representative terms as row names.

Usage

```
cluster_hmap(
  cluster_result,
  clusters = NULL,
  value_type = "Padj",
  aggr_type = mean
)
```

Arguments

cluster_result	A list containing a data frame (cluster_df) with clustering results. The data frame must contain at least the columns Cluster, Term, and value_type_* values.
clusters	Optional. A numeric or character vector specifying the clusters to include. If NULL (default), all clusters are included.
value_type	A character string specifying the column name prefix for values to display in hmap cells. Defaults to "Padj".
aggr_type	A function used to aggregate values across clusters (e.g., mean or median). Defaults to mean.

Details

The function processes the given cluster data frame (cluster_df), aggregating the value_type_* values per cluster using the specified aggr_type function. The -log10 transformation is applied, and infinite values are replaced with 0.

Representative terms are selected by choosing the term with the lowest value_type in each cluster. Where two clusters share a representative term, both rows are labelled <term> (cluster <id>) so every row label is unique.

The final heatmap is generated using heatmaply::heatmaply(), with an interactive plotly visualization.

Value

An interactive heatmap object (plotly), displaying the -log10(Padj) values across clusters, with representative terms as row labels.

Examples

```
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
chmap <- cluster_hmap(cluster_result)
chmap
```

cluster_network *Create a Network Graph for a Specific Cluster*

Description

This function generates a network graph for a specific cluster based on the provided distance matrix. The opacity and length of the edges correspond to the given distance_metric (eg, kappa) score similarity between terms, which is based on shared gene content.

Usage

```
cluster_network(final_clusters, distance_matrix, cluster_number, merged_df)
```

Arguments

final_clusters	A dataframe containing the final cluster data.
distance_matrix	A matrix representing the distances between terms.
cluster_number	An integer cluster id, as found in final_clusters$Cluster.
merged_df	A dataframe with all terms used to map term indices to names.

Value

An interactive networkD3 network graph.

Examples

```
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
net <- cluster_network(cluster_result$final_clusters,
                       cluster_result$distance_matrix,
                       cluster_number = 1,
                       cluster_result$merged_df)
net
```

compare_network_graphs_plotly *Compare Network Graphs using Plotly*

Description

This function creates a side-by-side comparison of network graphs for a single cluster using different p-value types.

Usage

```
compare_network_graphs_plotly(cluster_result, cluster_num, pval_names)
```

Arguments

cluster_result	The result from the clustering function.
cluster_num	The cluster number to plot, as found in cluster_result$cluster_df$Cluster.
pval_names	A list of p-value names to compare.

Value

A plotly object.

Examples

```
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
g <- compare_network_graphs_plotly(cluster_result, cluster_num = 1,
                                   c("Padj_1", "Padj_2"))
g
```

---

david_cluster *Cluster Terms using DAVID's method*

Description

This function performs clustering on enrichment results using an algorithm inspired by DAVID's functional clustering method.

Usage

```
david_cluster(
  enrichment_results,
  df_names = NULL,
  similarity_threshold = 0.5,
  initial_group_membership = 3,
  final_group_membership = 3,
  multiple_linkage_threshold = 0.5,
  verbose = FALSE
)
```

Arguments

enrichment_results	A list of dataframes, each containing enrichment results. Each dataframe should include at least the columns 'Term', 'GeneID', and 'Padj'.
df_names	Optional, a character vector of names for the enrichment result dataframes. Must match the length of enrichment_results. Default is NULL.
similarity_threshold	A numeric value for the kappa score cutoff (0 < cutoff <= 1). The comparison is STRICT: a pair counts as similar when its kappa is greater than the threshold, not when it equals it.
initial_group_membership	Minimum number of terms to form an initial seed group.
final_group_membership	Minimum number of terms for a final cluster.
multiple_linkage_threshold	A numeric value for the merging threshold. The comparison is STRICT: groups merge when their shared fraction is greater than the threshold, not when it equals it.
verbose	Logical; print the C++ core's progress narration to the console. Default FALSE — the core runs silent. Before 2.0.0 this narration was unconditional and could not be switched off, and it scales with the data (one line per merge iteration), so it is now opt-in.

Value

A named list containing the clustering results.

Examples

```
# A small two-dataset input.  The shipped data is far larger, and an example
# that runs during R CMD check should stay quick.
terms <- sprintf("TERM_%02d", 1:12)
genes <- vapply(1:12, function(i) paste0("G", i:(i + 4), collapse = ","),
                character(1))
d1 <- data.frame(Term = terms, GeneID = genes,
                 Pvalue = 1e-5, Padj = 1e-4, stringsAsFactors = FALSE)
d2 <- d1
res <- david_cluster(list(d1, d2), similarity_threshold = 0.5)
names(res)
```

---

export_df *Export Cluster Result as Dataframe*

Description

Returns a comprehensive dataframe containing all the different terms in all clusters.

Usage

```
export_df(cluster_result)
```

Arguments

cluster_result	The cluster_result object from cluster()

Value

A data.frame view of the clustering

Examples

```
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
# A flat, sharable table: one row per term, cluster ids resolved to the
# most significant term in each cluster.
flat <- export_df(cluster_result)
dim(flat)
head(flat[, 1:3])
```

---

filter_clusters *Filter Clusters by Number of Terms*

Description

Filters the full list of clusters by keeping only those with greater than or equal to min_terms # of terms.

Usage

```
filter_clusters(all_clusters, min_terms)
```

Arguments

all_clusters	A dataframe containing the merged seeds with column named ClusterIndices.
min_terms	An integer specifying the minimum number of terms required in a cluster.

Value

The filtered data frame with clusters filtered to include only those with at least min_terms terms.

Examples

```
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
# Keep only clusters carrying at least 10 terms.
kept <- filter_clusters(cluster_result$all_clusters, min_terms = 10)
nrow(cluster_result$all_clusters)
nrow(kept)
```

---

format_colnames *Format Column Names for Merging*

Description

This function maps a vector of column names to standardized names for "GeneID", "Pvalue", and "Padj" based on known variations.

Usage

```
format_colnames(colnames)
```

Arguments

colnames	A character vector of column names to be standardized.

Value

A character vector of standardized column names.

---

full_network *Create a Network Graph for the Entire Distance Matrix*

Description

This function generates a network graph for the entire distance matrix.

Usage

```
full_network(cluster_result)
```

Arguments

cluster_result	Cluster result named list from richCluster::cluster()

Value

An interactive networkD3 network graph.

Examples

```
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
fnet <- full_network(cluster_result)
fnet
```

`merge_enrichment_results` *Merge List of Enrichment Results*

Description

This function merges multiple enrichment results ('enrichment_results') into a single dataframe by combining unique GeneID elements across each geneset, and averaging Pvalue / Padj values for each term across all enrichment_results.

Usage

```
merge_enrichment_results(enrichment_results, gene_delim = ",")
```

Arguments

`enrichment_results`	A list of geneset dataframes containing columns c('Term', 'GeneID', 'Pvalue', 'Padj'). A list of length 1 is supported and returns the same merged columns as a longer list, row-ordered by 'Term'.
`gene_delim`	A single non-empty string separating gene identifiers within the 'GeneID' column. Must match the delimiter the caller's data actually uses: the merge splits on it and re-joins on it, so a mismatch fabricates identifiers that appear in no input. Defaults to ",", which reproduces the pre-parameter behaviour exactly.

Details

Each gene identifier appears at most once per term, ordered by first appearance across the contributing genesets.

A missing (`NA`) or empty gene list is treated as the EMPTY SET: the term is similar to nothing, so it joins no cluster and `min_terms` drops it. No row is removed from the caller's data and no count changes. A classed warning (`richCluster_empty_gene_list`) names how many terms were affected, so the exclusion is never silent.

Value

A single merged geneset dataframe with all original columns suffixed with the index of the geneset, with new columns 'GeneID', 'Pvalue', 'Padj' containing the merged values, and a trailing integer column 'DatasetCount' giving the number of input datasets that reported each term.

Examples

```
d1 <- utils::read.delim(system.file("extdata", "HF36wk_vs_HF12wk.txt",
                                    package = "richCluster"))
d2 <- utils::read.delim(system.file("extdata", "HF36wk_vs_WT12wk.txt",
                                    package = "richCluster"))
merged <- merge_enrichment_results(list(d1, d2))
head(merged[, c("Term", "Pvalue", "Padj", "DatasetCount")])
```

---

plot_network_graph *Plot Network Graph for a Cluster*

Description

This function visualizes a single cluster as a network graph.

Usage

```
plot_network_graph(
  cluster_result,
  cluster_num,
  distance_matrix,
  valuetype_list
)
```

Arguments

`cluster_result`	The result from the clustering function.
`cluster_num`	The cluster number to plot, as found in `cluster_result$cluster_df$Cluster`.
`distance_matrix`	The distance matrix used for clustering.
`valuetype_list`	A list of value types (e.g., "Pvalue_1", "Padj_1") to use for node coloring.

Value

`NULL`, invisibly. Called for its side effect: the network graph and its colour legend are drawn on the current graphics device.

Examples

```
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
# valuetype_list names the per-dataset significance columns to colour by.
g <- plot_network_graph(cluster_result, cluster_num = 1,
                        cluster_result$distance_matrix,
                        c("Padj_1", "Padj_2"))
g
```

`runRichCluster` *Run clustering in C++ backend*

Description

Run clustering in C++ backend

Usage

```
runRichCluster(
  terms,
  geneIDs,
  distanceMetric,
  distanceCutoff,
  linkageMethod,
  linkageCutoff,
  geneDelim = ",",
  verbose = FALSE
)
```

Arguments

`terms`	Character vector of term names
`geneIDs`	Character vector of geneIDs
`distanceMetric`	e.g. "kappa"
`distanceCutoff`	numeric between 0 and 1
`linkageMethod`	e.g. "average"
`linkageCutoff`	numeric between 0 and 1
`geneDelim`	single string separating gene identifiers within `geneIDs`. Default `","`.
`verbose`	Logical; print the C++ core's progress narration to the console. Default `FALSE` — the core runs silent. Before 2.0.0 this narration was unconditional and could not be switched off, and it scales with the data (one line per merge iteration), so it is now opt-in.

Details

A missing (`NA`) or empty gene list is treated as the EMPTY SET: the term is similar to nothing, so it joins no cluster and `min_terms` drops it. No row is removed from the caller's data and no count changes. A classed warning (`richCluster_empty_gene_list`) names how many terms were affected, so the exclusion is never silent.

Value

A list of two elements:

distance_matrix A numeric n x n matrix of pairwise SIMILARITY scores between the n input terms. Values lie on [0, 1] and LARGER means MORE similar; the diagonal is 1, the self-similarity every supported metric agrees on. For "kappa", a negative value is set to 0, so the matrix holds a non-negative kappa similarity rather than an unmodified Cohen's kappa. Row and column names are terms, in the caller's input order. Despite the element name it holds similarities, not distances; the name is retained so that existing code keeps working. Linkage converts internally with d = 1 - similarity.

all_clusters A data frame of every merged cluster, with three columns: Cluster (integer cluster ID), TermNames (the cluster's term names, comma-separated) and TermIndices (the same terms as ZERO-based indices into terms, comma-separated).

Determinism

Clustering results depend only on the terms and gene sets supplied, not on the order in which rows are given: cluster() and runRichCluster() canonicalise row order internally, so re-sorting the input cannot change cluster membership. Note that the merge stage is a greedy agglomeration — clusters are built by repeatedly merging the best-scoring available pair — so results reflect that greedy strategy rather than a global optimum, and small changes to linkage_cutoff can change membership substantially.

Examples

```
res <- runRichCluster(
  terms = c("T1", "T2", "T3"),
  geneIDs = c("a,b,c", "b,c,d", "x,y,z"),
  distanceMetric = "kappa", distanceCutoff = 0.5,
  linkageMethod = "average", linkageCutoff = 0.5)
res$distance_matrix
```

---

term_bar *Term-level Bar Plot for a Specific Cluster*

Description

Creates a horizontal bar plot showing enrichment values for individual terms in a selected cluster.

Usage

```
term_bar(cluster_result, cluster = 1, value_type = "Padj", title = NULL)
```

Arguments

cluster_result	A result list returned by cluster.
cluster	A single cluster ID (numeric) or term name (character) to visualize. A numeric ID must appear in cluster_df$Cluster. A term name must belong to exactly one cluster; a term shared by several clusters is ambiguous here and is rejected, naming the clusters it belongs to so one can be passed instead.
value_type	The column name to use for enrichment significance ("Padj" or "Pvalue").
title	Optional plot title. If NULL, a default will be generated.

Value

A plotly object representing the bar plot.

Examples

```
# Load example data
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
tbar <- term_bar(cluster_result, cluster = 1)
tbar
```

---

term_dot *Term-level Dot Plot for a Specific Cluster*

Description

Creates a dot plot of individual terms within a specified cluster, showing their significance and number of genes.

Usage

```
term_dot(cluster_result, cluster = 1, value_type = "Padj", title = NULL)
```

Arguments

cluster_result	A result list returned from cluster.
cluster	A single cluster ID (numeric) or term name (character) to plot. A numeric ID must appear in cluster_df$Cluster. A term name must belong to exactly one cluster; a term shared by several clusters is ambiguous here and is rejected, naming the clusters it belongs to so one can be passed instead.
value_type	The name of the value column to visualize (e.g., "Padj" or "Pvalue").
title	Optional title for the plot. If NULL, a default title is generated using the representative term.

Value

A plotly object representing the dot plot of terms.

Examples

```
# Load example data
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
tdot <- term_dot(cluster_result, cluster = 1)
tdot
```

---

term_hmap *Generate a Heatmap of Enrichment Results for Specific Clusters and Terms*

Description

Creates an interactive heatmap displaying -log10(Padj) values for selected clusters and terms. Users can specify clusters numerically or select them by providing term names. The function ensures that the final heatmap includes all terms from the selected clusters as well as any explicitly provided terms.

Usage

```
term_hmap(
  cluster_result,
  clusters = NULL,
  terms = NULL,
  value_type = "Padj",
  aggr_type = mean,
  title = NULL
)
```

Arguments

cluster_result	A list containing a data frame (cluster_df) with clustering results. The data frame must include at least the columns Cluster, Term, and Padj_* values.
clusters	Optional. A numeric vector specifying the cluster numbers to display, or a character vector specifying terms whose clusters should be included. Defaults to NULL, which includes all clusters. Numeric ids must appear in cluster_df$Cluster.
terms	Optional. A character vector specifying additional terms to include in the heatmap. Defaults to NULL.
value_type	A character string specifying the column name prefix for adjusted p-values. Defaults to "Padj".
aggr_type	A function used to aggregate values across clusters (e.g., mean or median). Defaults to mean.
title	An optional parameter to title the plot something else.

Details

The function processes the given `cluster_df`, identifying the clusters and terms to be visualized. If `clusters` is specified as a numeric vector, the function directly filters based on cluster numbers. If `clusters` is given as a character vector, it identifies the clusters associated with those terms and retrieves all terms from the selected clusters.

The `Padj_*` values are transformed using `-log10()`, and infinite values are replaced with `0`. The resulting heatmap is generated using `heatmaply::heatmaply()` with fixed row ordering (no hierarchical clustering).

Value

An interactive heatmap object (`plotly`), displaying the -log10(Padj) values across clusters, with representative terms as row labels and color-coded cluster annotations.

Examples

```
cluster_result <- readRDS(system.file("extdata", "cluster_result.rds",
                                      package = "richCluster"))
# All arguments after cluster_result have defaults; passing clusters
# restricts the heatmap to those cluster ids.
thmap <- term_hmap(cluster_result, clusters = c(1, 2))
thmap
```

Index

cluster, 2, *5*, *7*, *18*
cluster_bar, 5
cluster_correlation_hmap, 6
cluster_dot, 7
cluster_hmap, 7
cluster_network, 9
compare_network_graphs_plotly, 9

david_cluster, 10

export_df, 11

filter_clusters, 12
format_colnames, 13
full_network, 13

merge_enrichment_results, 14

plot_network_graph, 15

runRichCluster, 16

term_bar, 17
term_dot, 18
term_hmap, 19