Package 'ggpicrust2'

April 3, 2025

Type Package

Title Make 'PICRUSt2' Output Analysis and Visualization Easier

Version 2.0.0

Description Provides a convenient way to analyze and visualize 'PICRUSt2' output with predefined plots and functions. Allows for generating statistical plots about microbiome functional predictions and offers customization options. Features a one-click option for creating publication-level plots, saving time and effort in producing professional-grade figures. Streamlines the 'PICRUSt2' analysis and visualization process. For more details, see Yang et al. (2023) <doi:10.1093/bioinformatics/btad470>.

BugReports https://github.com/cafferychen777/ggpicrust2/issues

URL https://github.com/cafferychen777/ggpicrust2, https://cafferyang.com/ggpicrust2/

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Imports aplot, dplyr, ggplot2, grid, ggh4x, readr, tibble, tidyr, ggprism, patchwork, ggplotify, magrittr, progress, stats, methods, grDevices, utils

Depends R (>= 3.5.0)

Suggests Biobase, KEGGREST, ComplexHeatmap, BiocGenerics, knitr, rmarkdown, testthat (>= 3.0.0), ALDEx2, DESeq2, edgeR, GGally, limma, Maaslin2, metagenomeSeq, MicrobiomeStat, SummarizedExperiment, circlize, lefser

Config/testthat/edition 3

biocViews Microbiome, Metagenomics, Software

NeedsCompilation no

Author Chen Yang [aut, cre], Liangliang Zhang [aut]

Maintainer Chen Yang <cafferychen7850@gmail.com>

Repository CRAN

Date/Publication 2025-04-03 16:00:05 UTC

Contents

`compare_daa_results` *Compare the Consistency of Statistically Significant Features*

Description

This function compares the consistency and inconsistency of statistically significant features obtained using different methods in ‘pathway_daa‘ from the ‘ggpicrust2‘ package. It creates a report showing the number of common and different features identified by each method, and the features themselves.

Arguments

`daa_results_list`
: A list of data frames containing statistically significant features obtained using different methods.

`method_names`
: A character vector of names for each method used.

`p_values_threshold`
: A numeric value representing the threshold for the p-values. Features with p-values less than this threshold are considered statistically significant. Default is 0.05.

Value

A data frame with the comparison results. The data frame has the following columns:

- `method`: The name of the method.
- `num_features`: The total number of statistically significant features obtained by the method.
- `num_common_features`: The number of features that are common to other methods.
- `num_diff_features`: The number of features that are different from other methods.
- `diff_features`: The names of the features that are different from other methods.

Examples

```
library(magrittr)
library(ggpicrust2)
library(tibble)
data("metacyc_abundance")
data("metadata")

# Run pathway_daa function for multiple methods
methods <- c("DESeq2", "edgeR","Maaslin2")
daa_results_list <- lapply(methods, function(method) {
pathway_daa(abundance = metacyc_abundance %>% column_to_rownames("pathway"),
metadata = metadata, group = "Environment", daa_method = method)
})

names(daa_results_list) <- methods
# Correct Maaslin2 feature names by replacing dots with hyphens.
# Note: When using Maaslin2 as the differential abundance analysis method,
# it modifies the original feature names by replacing hyphens (-) with dots (.).
# This replacement can cause inconsistencies when trying to compare results from Maaslin2
# with those from other methods that do not modify feature names.
# Therefore, this line of code reverses that replacement, converting the dots back into
# hyphens for accurate and consistent comparisons across different methods.
daa_results_list[["Maaslin2"]]$feature <- gsub("\\.", "-", daa_results_list[["Maaslin2"]]$feature)

# Compare results across different methods
comparison_results <- compare_daa_results(daa_results_list = daa_results_list,
method_names = c("DESeq2", "edgeR", "Maaslin2"))
```

`compare_metagenome_results`

Compare Metagenome Results

Description

Compare Metagenome Results

Arguments

metagenomes	A list of metagenomes matrices with rows as KOs and columns as samples. Each matrix in the list should correspond to a different metagenome.
names	A vector of names for the metagenomes in the same order as in the ‘metagenomes‘ list.
daa_method	A character specifying the method for differential abundance analysis (DAA). Possible choices are: "ALDEx2", "DESeq2", "edgeR", "limma voom", "metagenomeSeq", "LinDA", "Maaslin2", and "Lefse". The default is "ALDEx2".
p.adjust	A character specifying the method for p-value adjustment. Possible choices are: "BH" (Benjamini-Hochberg), "holm", "bonferroni", "hochberg", "fdr", and "none". The default is "BH".
reference	A character specifying the reference group level for DAA. This parameter is used when there are more than two groups. The default is NULL.

Value

A list containing two elements:

- "daa": a list of results from the ‘pathway_daa‘ function. Each result is a data frame containing the differential abundance analysis results with columns for the feature ID, the test statistic, the raw p-value, and the adjusted p-value.
- "correlation": a list with two elements: "cor_matrix" and "p_matrix", which are matrices of Spearman correlation coefficients and their corresponding p-values, respectively, between every pair of metagenomes.

Examples

```
library(dplyr)
library(ComplexHeatmap)
# Generate example data
set.seed(123)
# First metagenome
metagenome1 <- abs(matrix(rnorm(1000), nrow = 100, ncol = 10))
rownames(metagenome1) <- paste0("KO", 1:100)
colnames(metagenome1) <- paste0("sample", 1:10)
# Second metagenome
metagenome2 <- abs(matrix(rnorm(1000), nrow = 100, ncol = 10))
rownames(metagenome2) <- paste0("KO", 1:100)
colnames(metagenome2) <- paste0("sample", 1:10)
# Put the metagenomes into a list
metagenomes <- list(metagenome1, metagenome2)
# Define names
names <- c("metagenome1", "metagenome2")
# Call the function
results <- compare_metagenome_results(metagenomes, names, daa_method = "LinDA")
# Print the correlation matrix
print(results$correlation$cor_matrix)
# Print the p-value matrix
print(results$correlation$p_matrix)
```

daa_annotated_results_df *Differentially Abundant Analysis Results with Annotation*

Description

This is a result dataset after processing 'kegg_abundance' through the 'pathway_daa' with the LinDA method and further annotation with 'pathway_annotation'.

Usage

```
daa_annotated_results_df
```

Format

A data frame with 10 variables:

adj_method Method used for adjusting p-values.

feature Feature being tested.

group1 One group in the comparison.

group2 The other group in the comparison.

method Statistical test used.

p_adjust Adjusted p-value.

p_values P-values from the statistical test.

pathway_class Class of the pathway.

pathway_description Description of the pathway.

pathway_map Map of the pathway.

pathway_name Name of the pathway.

Source

From ggpicrust2 package demonstration.

References

Douglas GM, Maffei VJ, Zaneveld J, Yurgel SN, Brown JR, Taylor CM, Huttenhower C, Langille MGI. PICRUSt2 for prediction of metagenome functions. Nat Biotechnol. 2020.

`daa_results_df` *DAA Results Dataset*

Description

This dataset is the result of processing 'kegg_abundance' through the 'LinDA' method in the 'pathway_daa' function. It includes information about the feature, groups compared, p values, and method used.

Usage

```
daa_results_df
```

Format

A data frame with columns:

adj_method Method used for p-value adjustment.

feature The feature (pathway) being compared.

group1 The first group in the comparison.

group2 The second group in the comparison.

method The method used for the comparison.

p_adjust The adjusted p-value from the comparison.

p_values The raw p-value from the comparison.

Source

From ggpicrust2 package demonstration.

References

Douglas GM, Maffei VJ, Zaneveld J, Yurgel SN, Brown JR, Taylor CM, Huttenhower C, Langille MGI. PICRUSt2 for prediction of metagenome functions. Nat Biotechnol. 2020.

`ggpicrust2` *This function integrates pathway name/description annotations, ten of the most advanced differential abundance (DA) methods, and visualization of DA results.*

Description

This function integrates pathway name/description annotations, ten of the most advanced differential abundance (DA) methods, and visualization of DA results.

Usage

```
ggpicrust2(
  file = NULL,
  data = NULL,
  metadata,
  group,
  pathway,
  daa_method = "ALDEx2",
  ko_to_kegg = FALSE,
  p.adjust = "BH",
  order = "group",
  p_values_bar = TRUE,
  x_lab = NULL,
  select = NULL,
  reference = NULL,
  colors = NULL
)
```

Arguments

file	A character string representing the file path of the input file containing KO abundance data in picrust2 export format. The input file should have KO identifiers in the first column and sample identifiers in the first row. The remaining cells should contain the abundance values for each KO-sample pair.
data	An optional data.frame containing KO abundance data in the same format as the input file. If provided, the function will use this data instead of reading from the file. By default, this parameter is set to NULL.
metadata	A tibble, consisting of sample information
group	A character, name of the group
pathway	A character, consisting of "EC", "KO", "MetaCyc"
daa_method	a character specifying the method for differential abundance analysis, default is "ALDEx2", choices are: - "ALDEx2": ANOVA-Like Differential Expression tool for high throughput sequencing data - "DESeq2": Differential expression analysis based on the negative binomial distribution using DESeq2 - "edgeR": Exact test for differences between two groups of negative-binomially distributed counts using edgeR - "limma voom": Limma-voom framework for the analysis of RNA-seq data - "metagenomeSeq": Fit logistic regression models to test for differential abundance between groups using metagenomeSeq - "LinDA": Linear models for differential abundance analysis of microbiome compositional data - "Maaslin2": Multivariate Association with Linear Models (MaAsLin2) for differential abundance analysis
ko_to_kegg	A character to control the conversion of KO abundance to KEGG abundance
p.adjust	a character specifying the method for p-value adjustment, default is "BH", choices are: - "BH": Benjamini-Hochberg correction - "holm": Holm's correction - "bonferroni": Bonferroni correction - "hochberg": Hochberg's correction - "fdr": False discovery rate correction - "none": No p-value adjustment.

order	A character to control the order of the main plot rows
p_values_bar	A character to control if the main plot has the p_values bar
x_lab	A character to control the x-axis label name, you can choose from "feature","pathway_name" and "description"
select	A vector consisting of pathway names to be selected
reference	A character, a reference group level for several DA methods
colors	A vector consisting of colors number

Value

daa.results.df, a dataframe of DA results A list of sub-lists, each containing a ggplot2 plot (‘plot‘) and a dataframe of differential abundance results (‘results‘) for a specific DA method. Each plot visualizes the differential abundance results of a specific DA method, and the corresponding dataframe contains the results used to create the plot.

Examples

```
## Not run:
# Load necessary data: abundance data and metadata
abundance_file <- "path/to/your/abundance_file.tsv"
metadata <- read.csv("path/to/your/metadata.csv")

# Run ggpicrust2 with input file path
results_file_input <- ggpicrust2(file = abundance_file,
                                 metadata = metadata,
                                 group = "your_group_column",
                                 pathway = "KO",
                                 daa_method = "LinDA",
                                 ko_to_kegg = "TRUE",
                                 order = "pathway_class",
                                 p_values_bar = TRUE,
                                 x_lab = "pathway_name")

# Run ggpicrust2 with imported data.frame
abundance_data <- read_delim(abundance_file, delim="\t", col_names=TRUE, trim_ws=TRUE)

# Run ggpicrust2 with input data
results_data_input <- ggpicrust2(data = abundance_data,
                                 metadata = metadata,
                                 group = "your_group_column",
                                 pathway = "KO",
                                 daa_method = "LinDA",
                                 ko_to_kegg = "TRUE",
                                 order = "pathway_class",
                                 p_values_bar = TRUE,
                                 x_lab = "pathway_name")

# Access the plot and results dataframe for the first DA method
example_plot <- results_file_input[[1]]$plot
example_results <- results_file_input[[1]]$results
```

```
# Use the example data in ggpicrust2 package
data(ko_abundance)
data(metadata)
results_file_input <- ggpicrust2(data = ko_abundance,
                                 metadata = metadata,
                                 group = "Environment",
                                 pathway = "KO",
                                 daa_method = "LinDA",
                                 ko_to_kegg = TRUE,
                                 order = "pathway_class",
                                 p_values_bar = TRUE,
                                 x_lab = "pathway_name")
# Analyze the EC or MetaCyc pathway
data(metacyc_abundance)
results_file_input <- ggpicrust2(data = metacyc_abundance,
                                 metadata = metadata,
                                 group = "Environment",
                                 pathway = "MetaCyc",
                                 daa_method = "LinDA",
                                 ko_to_kegg = FALSE,
                                 order = "group",
                                 p_values_bar = TRUE,
                                 x_lab = "description")

## End(Not run)
```

`import_MicrobiomeAnalyst_daa_results` *Import Differential Abundance Analysis (DAA) results from MicrobiomeAnalyst*

Description

This function imports DAA results from an external platform such as MicrobiomeAnalyst. It can be used to compare the results obtained from different platforms.

Arguments

`file_path`	a character string specifying the path to the CSV file containing the DAA results from MicrobiomeAnalyst. If this parameter is NULL and no data frame is provided, an error will be thrown. Default is NULL.
`data`	a data frame containing the DAA results from MicrobiomeAnalyst. If this parameter is NULL and no file path is provided, an error will be thrown. Default is NULL.
`method`	a character string specifying the method used for the DAA. This will be added as a new column in the returned data frame. Default is "MicrobiomeAnalyst".
`group_levels`	a character vector specifying the group levels for the DAA. This will be added as new columns in the returned data frame. Default is c("control", "treatment").

Value

a data frame containing the DAA results from MicrobiomeAnalyst with additional columns for the method and group levels.

Examples

```
## Not run:
# Assuming you have a CSV file named "DAA_results.csv" in your current directory
daa_results <- import_MicrobiomeAnalyst_daa_results(file_path = "DAA_results.csv")

## End(Not run)
```

---

kegg_abundance *KEGG Abundance Dataset*

Description

A dataset derived from 'ko_abundance' by the function 'ko2kegg_abundance' in the ggpicrust2 package. Each row corresponds to a KEGG pathway, and each column corresponds to a sample.

Usage

```
kegg_abundance
```

Format

A data frame where rownames are KEGG pathways and column names are individual sample names, including: "SRR11393730", "SRR11393731", "SRR11393732", "SRR11393733", "SRR11393734", "SRR11393735", "SRR11393736", "SRR11393737", "SRR11393738", "SRR11393739", "SRR11393740", "SRR11393741", "SRR11393742", "SRR11393743", "SRR11393744", "SRR11393745", "SRR11393746", "SRR11393747", "SRR11393748", "SRR11393749", "SRR11393750", "SRR11393751", "SRR11393752", "SRR11393753", "SRR11393754", "SRR11393755", "SRR11393756", "SRR11393757", "SRR11393758", "SRR11393759", "SRR11393760", "SRR11393761", "SRR11393762", "SRR11393763", "SRR11393764", "SRR11393765", "SRR11393766", "SRR11393767", "SRR11393768", "SRR11393769", "SRR11393770", "SRR11393771", "SRR11393772", "SRR11393773", "SRR11393774", "SRR11393775", "SRR11393776", "SRR11393777", "SRR11393778", "SRR11393779"

Source

From ggpicrust2 package demonstration.

References

Douglas GM, Maffei VJ, Zaneveld J, Yurgel SN, Brown JR, Taylor CM, Huttenhower C, Langille MGI. PICRUSt2 for prediction of metagenome functions. Nat Biotechnol. 2020.

ko2kegg_abundance *Convert KO abundance in picrust2 export files to KEGG pathway abundance*

Description

This function takes a file containing KO (KEGG Orthology) abundance data in picrust2 export format and converts it to KEGG pathway abundance data. The input file should be in .tsv, .txt, or .csv format.

Usage

```
ko2kegg_abundance(file = NULL, data = NULL)
```

Arguments

file	A character string representing the file path of the input file containing KO abundance data in picrust2 export format. The input file should have KO identifiers in the first column and sample identifiers in the first row. The remaining cells should contain the abundance values for each KO-sample pair.
data	An optional data.frame containing KO abundance data in the same format as the input file. If provided, the function will use this data instead of reading from the file. By default, this parameter is set to NULL.

Value

A data frame with KEGG pathway abundance values. Rows represent KEGG pathways, identified by their KEGG pathway IDs. Columns represent samples, identified by their sample IDs from the input file. Each cell contains the abundance of a specific KEGG pathway in a given sample, calculated by summing the abundances of the corresponding KOs in the input file.

Examples

```
## Not run:
library(ggpicrust2)
library(readr)

# Example 1: Demonstration with a hypothetical input file

# Prepare an input file path
input_file <- "path/to/your/picrust2/results/pred_metagenome_unstrat.tsv"

# Run ko2kegg_abundance function
kegg_abundance <- ko2kegg_abundance(file = input_file)

# Alternatively, read the data from a file and use the data argument
file_path <- "path/to/your/picrust2/results/pred_metagenome_unstrat.tsv"
ko_abundance <- read_delim(file_path, delim = "\t")
```

```
kegg_abundance <- ko2kegg_abundance(data = ko_abundance)

# Example 2: Working with real data
# In this case, we're using an existing dataset from the ggpicrust2 package.

# Load the data
data(ko_abundance)

# Apply the ko2kegg_abundance function to our real dataset
kegg_abundance <- ko2kegg_abundance(data = ko_abundance)

## End(Not run)
```

---

ko_abundance *KO Abundance Dataset*

Description

This is a demonstration dataset from the ggpicrust2 package, representing the output of PICRUSt2. Each row represents a KO (KEGG Orthology) group, and each column corresponds to a sample.

Usage

```
ko_abundance
```

Format

A data frame where rownames are KO groups and column names include #NAME and individual sample names, such as: "#NAME", "SRR11393730", "SRR11393731", "SRR11393732", "SRR11393733", "SRR11393734", "SRR11393735", "SRR11393736", "SRR11393737", "SRR11393738", "SRR11393739", "SRR11393740", "SRR11393741", "SRR11393742", "SRR11393743", "SRR11393744", "SRR11393745", "SRR11393746", "SRR11393747", "SRR11393748", "SRR11393749", "SRR11393750", "SRR11393751", "SRR11393752", "SRR11393753", "SRR11393754", "SRR11393755", "SRR11393756", "SRR11393757", "SRR11393758", "SRR11393759", "SRR11393760", "SRR11393761", "SRR11393762", "SRR11393763", "SRR11393764", "SRR11393765", "SRR11393766", "SRR11393767", "SRR11393768", "SRR11393769", "SRR11393770", "SRR11393771", "SRR11393772", "SRR11393773", "SRR11393774", "SRR11393775", "SRR11393776", "SRR11393777", "SRR11393778", "SRR11393779"

Source

From ggpicrust2 package demonstration.

References

Douglas GM, Maffei VJ, Zaneveld J, Yurgel SN, Brown JR, Taylor CM, Huttenhower C, Langille MGI. PICRUSt2 for prediction of metagenome functions. Nat Biotechnol. 2020.

metacyc_abundance *MetaCyc Abundance Dataset*

Description

This is a demonstration dataset from the ggpicrust2 package, representing the output of PICRUSt2. Each row represents a MetaCyc pathway, and each column corresponds to a sample.

Usage

```
metacyc_abundance
```

Format

A data frame where rownames are MetaCyc pathways and column names include "pathway" and individual sample names, such as: "pathway", "SRR11393730", "SRR11393731", "SRR11393732", "SRR11393733", "SRR11393734", "SRR11393735", "SRR11393736", "SRR11393737", "SRR11393738", "SRR11393739", "SRR11393740", "SRR11393741", "SRR11393742", "SRR11393743", "SRR11393744", "SRR11393745", "SRR11393746", "SRR11393747", "SRR11393748", "SRR11393749", "SRR11393750", "SRR11393751", "SRR11393752", "SRR11393753", "SRR11393754", "SRR11393755", "SRR11393756", "SRR11393757", "SRR11393758", "SRR11393759", "SRR11393760", "SRR11393761", "SRR11393762", "SRR11393763", "SRR11393764", "SRR11393765", "SRR11393766", "SRR11393767", "SRR11393768", "SRR11393769", "SRR11393770", "SRR11393771", "SRR11393772", "SRR11393773", "SRR11393774", "SRR11393775", "SRR11393776", "SRR11393777", "SRR11393778", "SRR11393779"

Source

From ggpicrust2 package demonstration.

References

Douglas GM, Maffei VJ, Zaneveld J, Yurgel SN, Brown JR, Taylor CM, Huttenhower C, Langille MGI. PICRUSt2 for prediction of metagenome functions. Nat Biotechnol. 2020.

metadata *Metadata for ggpicrust2 Demonstration*

Description

This is a demonstration dataset from the ggpicrust2 package. It provides the metadata required for the demonstration functions in the package. The dataset includes environmental information for each sample.

Usage

```
metadata
```

Format

A tibble with each row representing metadata for a sample.

Sample1 Metadata for Sample1, including Environment

Sample2 Metadata for Sample2, including Environment

... ...

Source

ggpicrust2 package demonstration.

References

Douglas GM, Maffei VJ, Zaneveld J, Yurgel SN, Brown JR, Taylor CM, Huttenhower C, Langille MGI. PICRUSt2 for prediction of metagenome functions. Nat Biotechnol. 2020.

---

pathway_annotation *Pathway information annotation of "EC", "KO", "MetaCyc" pathway*

Description

This function has two primary use cases: 1. Annotating pathway information using the output file from PICRUSt2. 2. Annotating pathway information from the output of ‘pathway_daa‘ function, and converting KO abundance to KEGG pathway abundance when ‘ko_to_kegg‘ is set to TRUE.

Usage

```
pathway_annotation(
  file = NULL,
  pathway = NULL,
  daa_results_df = NULL,
  ko_to_kegg = FALSE
)
```

Arguments

file	A character, address to store PICRUSt2 export files. Provide this parameter when using the function for the first use case.
pathway	A character, consisting of "EC", "KO", "MetaCyc"
daa_results_df	A data frame, output of pathway_daa. Provide this parameter when using the function for the second use case.
ko_to_kegg	A logical, decide if convert KO abundance to KEGG pathway abundance. Default is FALSE. Set to TRUE when using the function for the second use case.

Value

A data frame containing pathway annotation information. The data frame has the following columns:

- `feature`: The feature ID of the pathway (e.g., KO, EC, or MetaCyc ID).
- `description`: The description or name of the pathway.
- Other columns depending on the input parameters and type of pathway.

If `ko_to_kegg` is set to TRUE, the output data frame will also include the following columns:

- `pathway_name`: The name of the KEGG pathway.
- `pathway_description`: The description of the KEGG pathway.
- `pathway_class`: The class of the KEGG pathway.
- `pathway_map`: The KEGG pathway map ID.

Examples

```
## Not run:
# Prepare the required input files and data frames
# Then, you can use the pathway_annotation function as follows:

# Use case 1: Annotating pathway information using the output file from PICRUSt2
result1 <- pathway_annotation(file = "path/to/picrust2/export/file.txt",
                              pathway = "KO",
                              daa_results_df = NULL,
                              ko_to_kegg = FALSE)

# Use case 2: Annotating pathway information from the output of pathway_daa function
# and converting KO abundance to KEGG pathway abundance
# This use case will be demonstrated using both a hypothetical example, and a real dataset.

## Hypothetical example
hypothetical_daa_results_df <- data.frame() # Replace this with your actual data frame
result2 <- pathway_annotation(file = NULL,
                              pathway = "KO",
                              daa_results_df = hypothetical_daa_results_df,
                              ko_to_kegg = TRUE)

## Real dataset example
# Load the real dataset
data(daa_results_df)
result3 <- pathway_annotation(file = NULL,
                              pathway = "KO",
                              daa_results_df = daa_results_df,
                              ko_to_kegg = TRUE)

## End(Not run)
```

pathway_daa *Differential Abundance Analysis for Predicted Functional Pathways*

Description

Performs differential abundance analysis on predicted functional pathway data using various statistical methods. This function supports multiple methods for analyzing differences in pathway abundance between groups, including popular approaches like ALDEx2, DESeq2, edgeR, and others.

Usage

```
pathway_daa(
  abundance,
  metadata,
  group,
  daa_method = "ALDEx2",
  select = NULL,
  p.adjust = "BH",
  reference = NULL,
  ...
)
```

Arguments

abundance	A data frame or matrix containing predicted functional pathway abundance, with pathways/features as rows and samples as columns. The column names should match the sample names in metadata. Values should be counts or abundance measurements.
metadata	A data frame or tibble containing sample information. Must include a 'sample' column with sample identifiers matching the column names in abundance data.
group	Character string specifying the column name in metadata that contains group information for differential abundance analysis.
daa_method	Character string specifying the method for differential abundance analysis. Available choices are: • "ALDEx2": ANOVA-Like Differential Expression tool • "DESeq2": Differential expression analysis based on negative binomial distribution • "edgeR": Exact test for differences between groups using negative binomial model • "limma voom": Limma-voom framework for RNA-seq analysis • "metagenomeSeq": Zero-inflated Gaussian mixture model • "LinDA": Linear models for differential abundance analysis • "Maaslin2": Multivariate Association with Linear Models • "Lefser": Linear discriminant analysis effect size

Default is "ALDEx2".

`select` Vector of sample names to include in the analysis. If NULL (default), all samples are included.

`p.adjust` Character string specifying the method for p-value adjustment. Choices are:

- `"BH"`: Benjamini-Hochberg procedure (default)
- `"holm"`: Holm's step-down method
- `"bonferroni"`: Bonferroni correction
- `"hochberg"`: Hochberg's step-up method
- `"fdr"`: False Discovery Rate
- `"none"`: No adjustment

`reference` Character string specifying the reference level for the group comparison. If NULL (default), the first level is used as reference.

`...` Additional arguments passed to the specific DAA method

Value

A data frame containing the differential abundance analysis results

References

- ALDEx2: Fernandes et al. (2014) Unifying the analysis of high-throughput sequencing datasets: characterizing RNA-seq, 16S rRNA gene sequencing and selective growth experiments by compositional data analysis. Microbiome.
- DESeq2: Love et al. (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biology.
- edgeR: Robinson et al. (2010) edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics.
- limma-voom: Law et al. (2014) voom: precision weights unlock linear model analysis tools for RNA-seq read counts. Genome Biology.
- metagenomeSeq: Paulson et al. (2013) Differential abundance analysis for microbial marker-gene surveys. Nature Methods.
- Maaslin2: Mallick et al. (2021) Multivariable Association Discovery in Population-scale Meta-omics Studies.

Examples

```
# Load example data
data(ko_abundance)
data(metadata)

# Prepare abundance data
abundance_data <- as.data.frame(ko_abundance)
rownames(abundance_data) <- abundance_data[, "#NAME"]
abundance_data <- abundance_data[, -1]

# Run differential abundance analysis using ALDEx2
```

```
results <- pathway_daa(
  abundance = abundance_data,
  metadata = metadata,
  group = "Environment"
)

# Using a different method (DESeq2)
deseq_results <- pathway_daa(
  abundance = abundance_data,
  metadata = metadata,
  group = "Environment",
  daa_method = "DESeq2"
)

# Create example data with more samples
abundance <- data.frame(
  sample1 = c(10, 20, 30),
  sample2 = c(20, 30, 40),
  sample3 = c(30, 40, 50),
  sample4 = c(40, 50, 60),
  sample5 = c(50, 60, 70),
  row.names = c("pathway1", "pathway2", "pathway3")
)

metadata <- data.frame(
  sample = c("sample1", "sample2", "sample3", "sample4", "sample5"),
  group = c("control", "control", "treatment", "treatment", "treatment")
)

# Run differential abundance analysis using ALDEx2
results <- pathway_daa(abundance, metadata, "group")

# Using a different method (limma voom instead of DESeq2 for this small example)
limma_results <- pathway_daa(abundance, metadata, "group",
                             daa_method = "limma voom")

# Analyze specific samples only
subset_results <- pathway_daa(abundance, metadata, "group",
                              select = c("sample1", "sample2", "sample3", "sample4"))
```

---

pathway_errorbar *The function pathway_errorbar() is used to visualize the results of functional pathway differential abundance analysis as error bar plots.*

Description

The function pathway_errorbar() is used to visualize the results of functional pathway differential abundance analysis as error bar plots.

Arguments

abundance	A data frame with row names representing pathways and column names representing samples. Each element represents the relative abundance of the corresponding pathway in the corresponding sample.
daa_results_df	A data frame containing the results of the differential abundance analysis of the pathways, generated by the pathway_daa function. x_lab should be a column name of daa_results_df.
Group	A data frame or a vector that assigns each sample to a group. The groups are used to color the samples in the figure.
ko_to_kegg	A logical parameter indicating whether there was a convertion that convert ko abundance to kegg abundance.
p_values_threshold	A numeric parameter specifying the threshold for statistical significance of differential abundance. Pathways with p-values below this threshold will be considered significant.
order	A parameter controlling the ordering of the rows in the figure. The options are: "p_values" (order by p-values), "name" (order by pathway name), "group" (order by the group with the highest mean relative abundance), or "pathway_class" (order by the pathway category).
select	A vector of pathway names to be included in the figure. This can be used to limit the number of pathways displayed. If NULL, all pathways will be displayed.
p_value_bar	A logical parameter indicating whether to display a bar showing the p-value threshold for significance. If TRUE, the bar will be displayed.
colors	A vector of colors to be used to represent the groups in the figure. Each color corresponds to a group.
x_lab	A character string to be used as the x-axis label in the figure. The default value is "description" for KOs'descriptions and "pathway_name" for KEGG pathway names.

Value

A ggplot2 plot showing the error bar plot of the differential abundance analysis results for the functional pathways. The plot visualizes the differential abundance results of a specific differential abundance analysis method. The corresponding dataframe contains the results used to create the plot.

Examples

```
## Not run:
# Example 1: Analyzing KEGG pathway abundance
metadata <- read_delim(
  "path/to/your/metadata.txt",
  delim = "\t",
  escape_double = FALSE,
  trim_ws = TRUE
)
```

```
# data(metadata)

kegg_abundance <- ko2kegg_abundance(
  "path/to/your/pred_metagenome_unstrat.tsv"
)

# data(kegg_abundance)

# Please change group to "your_group_column" if you are not using example dataset
group <- "Environment"

daa_results_df <- pathway_daa(
  abundance = kegg_abundance,
  metadata = metadata,
  group = group,
  daa_method = "ALDEx2",
  select = NULL,
  reference = NULL
)

# Please check the unique(daa_results_df$method) and choose one
daa_sub_method_results_df <- daa_results_df[daa_results_df$method
== "ALDEx2_Welch's t test", ]

daa_annotated_sub_method_results_df <- pathway_annotation(
  pathway = "KO",
  daa_results_df = daa_sub_method_results_df,
  ko_to_kegg = TRUE
)

# Please change Group to metadata$your_group_column if you are not using example dataset
Group <- metadata$Environment

p <- pathway_errorbar(
  abundance = kegg_abundance,
  daa_results_df = daa_annotated_sub_method_results_df,
  Group = Group,
  p_values_threshold = 0.05,
  order = "pathway_class",
  select = daa_annotated_sub_method_results_df %>%
  arrange(p_adjust) %>%
  slice(1:20) %>%
  select("feature") %>% pull(),
  ko_to_kegg = TRUE,
  p_value_bar = TRUE,
  colors = NULL,
  x_lab = "pathway_name"
)

# Example 2: Analyzing EC, MetaCyc, KO without conversions
metadata <- read_delim(
  "path/to/your/metadata.txt",
```

```
  delim = "\t",
  escape_double = FALSE,
  trim_ws = TRUE
)
# data(metadata)

metacyc_abundance <- read.delim("path/to/your/metacyc_abundance.tsv")

# data(metacyc_abundance)

group <- "Environment"

daa_results_df <- pathway_daa(
  abundance = metacyc_abundance %>% column_to_rownames("pathway"),
  metadata = metadata,
  group = group,
  daa_method = "LinDA",
  select = NULL,
  reference = NULL
)


daa_annotated_results_df <- pathway_annotation(
  pathway = "MetaCyc",
  daa_results_df = daa_results_df,
  ko_to_kegg = FALSE
)

Group <- metadata$Environment

p <- pathway_errorbar(
  abundance = metacyc_abundance %>% column_to_rownames("pathway"),
  daa_results_df = daa_annotated_results_df,
  Group = Group,
  p_values_threshold = 0.05,
  order = "group",
  select = NULL,
  ko_to_kegg = FALSE,
  p_value_bar = TRUE,
  colors = NULL,
  x_lab = "description"
)

## End(Not run)
```

---

pathway_heatmap *Create pathway heatmap*

Description

This function creates a heatmap of the predicted functional pathway abundance data. The function first makes the abundance data relative, then converts the abundance data to a long format and orders the samples based on the environment information. The heatmap is then created using the 'ggplot2' library.

Arguments

abundance	A matrix or data frame of pathway abundance data, where columns correspond to samples and rows correspond to pathways. Must contain at least two samples.
metadata	A data frame of metadata, where each row corresponds to a sample and each column corresponds to a metadata variable.
group	A character string specifying the column name in the metadata data frame that contains the group variable. Must contain at least two groups.
colors	A vector of colors used for the background of the facet labels in the heatmap. If NULL or not provided, a default color set is used for the facet strips.
font_size	A numeric value specifying the font size for the heatmap.
show_row_names	A logical value indicating whether to show row names in the heatmap.
show_legend	A logical value indicating whether to show the legend in the heatmap.
custom_theme	A custom theme for the heatmap.

Value

A ggplot heatmap object representing the heatmap of the predicted functional pathway abundance data.

Examples

```
library(ggpicrust2)
library(ggh4x)
library(dplyr)
library(tidyr)
library(tibble)
library(magrittr)

# Create example functional pathway abundance data
kegg_abundance_example <- matrix(rnorm(30), nrow = 3, ncol = 10)
colnames(kegg_abundance_example) <- paste0("Sample", 1:10)
rownames(kegg_abundance_example) <- c("PathwayA", "PathwayB", "PathwayC")

# Create example metadata
metadata_example <- data.frame(
  sample_name = colnames(kegg_abundance_example),
  group = factor(rep(c("Control", "Treatment"), each = 5))
)

# Custom colors for facet strips
custom_colors <- c("skyblue", "salmon")
```

```
# Create a heatmap using custom colors for facet strips
pathway_heatmap(kegg_abundance_example, metadata_example, "group", colors = custom_colors)

# Use real dataset
data("metacyc_abundance")
data("metadata")
metacyc_daa_results_df <- pathway_daa(
  abundance = metacyc_abundance %>% column_to_rownames("pathway"),
  metadata = metadata,
  group = "Environment",
  daa_method = "LinDA"
)
annotated_metacyc_daa_results_df <- pathway_annotation(
  pathway = "MetaCyc",
  daa_results_df = metacyc_daa_results_df,
  ko_to_kegg = FALSE
)
feature_with_p_0.05 <- metacyc_daa_results_df %>% filter(p_adjust < 0.05)
pathway_heatmap(
  abundance = metacyc_abundance %>%
    right_join(
      annotated_metacyc_daa_results_df %>%
      select(all_of(c("feature","description"))),
      by = c("pathway" = "feature")
    ) %>%
    filter(pathway %in% feature_with_p_0.05$feature) %>%
    select(-"pathway") %>%
    column_to_rownames("description"),
  metadata = metadata,
  group = "Environment",
  colors = custom_colors
)
```

---

pathway_pca *Perform Principal Component Analysis (PCA) on functional pathway abundance data*

Description

This function performs PCA analysis on pathway abundance data and creates an informative visualization that includes a scatter plot of the first two principal components (PC1 vs PC2) with density plots for both PCs. The plot helps to visualize the clustering patterns and distribution of samples across different groups.

Usage

```
pathway_pca(abundance, metadata, group, colors = NULL)
```

Arguments

abundance	A numeric matrix or data frame containing pathway abundance data. Rows represent pathways, columns represent samples. Column names must match the sample names in metadata. Values must be numeric and cannot contain missing values (NA).
metadata	A data frame containing sample information. Must include: • A column named "sample_name" matching the column names in abundance • A column for grouping samples (specified by the 'group' parameter)
group	A character string specifying the column name in metadata that contains group information for samples (e.g., "treatment", "condition", "group").
colors	Optional. A character vector of colors for different groups. Length must match the number of unique groups. If NULL, default colors will be used.

Details

The function performs several validations on input data:

- Abundance matrix must have at least 2 pathways and 3 samples
- All values in abundance matrix must be numeric
- Sample names must match between abundance and metadata
- Group column must exist in metadata
- If custom colors are provided, they must be valid color names or codes

Value

A ggplot object showing:

- Center: PCA scatter plot with confidence ellipses (95
- Top: Density plot for PC1
- Right: Density plot for PC2

Examples

```
# Create example abundance data
abundance_data <- matrix(rnorm(30), nrow = 3, ncol = 10)
colnames(abundance_data) <- paste0("Sample", 1:10)
rownames(abundance_data) <- c("PathwayA", "PathwayB", "PathwayC")

# Create example metadata
metadata <- data.frame(
  sample_name = paste0("Sample", 1:10),
  group = factor(rep(c("Control", "Treatment"), each = 5))
)

# Basic PCA plot with default colors
pca_plot <- pathway_pca(abundance_data, metadata, "group")
```

```
# PCA plot with custom colors
pca_plot <- pathway_pca(
  abundance_data,
  metadata,
  "group",
  colors = c("blue", "red")  # One color per group
)


# Example with real data
data("metacyc_abundance")  # Load example pathway abundance data
data("metadata")          # Load example metadata

# Generate PCA plot
# Prepare abundance data
abundance_data <- as.data.frame(metacyc_abundance)
rownames(abundance_data) <- abundance_data$pathway
abundance_data <- abundance_data[, -which(names(abundance_data) == "pathway")]

# Create PCA plot
pathway_pca(
  abundance_data,
  metadata,
  "Environment",
  colors = c("green", "purple")
)
```

`safe_extract` *Safely Extract Elements from a List*

Description

Safely extracts elements from a list, returning NA if the extraction fails

Usage

```
safe_extract(list, field, index = 1)
```

Arguments

`list`	A list object from which to extract elements
`field`	The name of the field to extract from the list
`index`	The index position to extract from the field. Default is 1

Value

The extracted element if successful, NA if extraction fails

Examples

```
# Create a sample list
my_list <- list(
  a = list(x = 1:3),
  b = list(y = 4:6)
)

# Extract existing element
safe_extract(my_list, "a", 1)

# Extract non-existing element (returns NA)
safe_extract(my_list, "c", 1)
```

Index