

# Package ‘animalcules’

December 19, 2025

**Title** Interactive microbiome analysis toolkit

**Version** 1.27.0

**Description** animalcules is an R package for utilizing up-to-date data analytics, visualization methods, and machine learning models to provide users an easy-to-use interactive microbiome analysis framework. It can be used as a standalone software package or users can explore their data with the accompanying interactive R Shiny application. Traditional microbiome analysis such as alpha/beta diversity and differential abundance analysis are enhanced, while new methods like biomarker identification are introduced by animalcules. Powerful interactive and dynamic figures generated by animalcules enable users to understand their data better and discover new insights.

**License** Artistic-2.0

**URL** <https://github.com/wejlab/animalcules>

**BugReports** <https://github.com/wejlab/animalcules/issues>

**Depends** R (>= 4.3.0)

**Imports** ape, assertthat, caret, covr, DESeq2, dplyr, DT, forcats, ggforce, ggplot2, GUniFrac, lattice, limma, magrittr, Matrix, methods, MultiAssayExperiment, plotly, rentrez, reshape2, ROCit, S4Vectors (>= 0.23.19), scales, shiny, shinyjs, stats, SummarizedExperiment, tibble, tidyr, tsne, umap, utils, vegan, XML

**Suggests** BiocStyle, biomformat, devtools, glmnet, knitr, rmarkdown, testthat, usethis

**VignetteBuilder** knitr

**biocViews** Microbiome, Metagenomics, Coverage, Visualization

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 7.3.0

**git\_url** <https://git.bioconductor.org/packages/animalcules>

**git\_branch** devel

**git\_last\_commit** 30597c8

**git\_last\_commit\_date** 2025-10-29

**Repository** Bioconductor 3.23

**Date/Publication** 2025-12-19

**Author** Jessica McClintock [cre],

Yue Zhao [aut] (ORCID: <<https://orcid.org/0000-0001-5257-5103>>),

Anthony Federico [aut] (ORCID: <<https://orcid.org/0000-0002-9200-1689>>),

W. Evan Johnson [aut] (ORCID: <<https://orcid.org/0000-0002-6247-6595>>)

**Maintainer** Jessica McClintock <jessica.mcclintock@rutgers.edu>

## Contents

|                                      |    |
|--------------------------------------|----|
| animalcules-package . . . . .        | 3  |
| alpha_div_boxplot . . . . .          | 4  |
| alpha_div_test . . . . .             | 5  |
| counts_to_logcpm . . . . .           | 5  |
| counts_to_relabu . . . . .           | 6  |
| df_char_to_factor . . . . .          | 6  |
| differential_abundance . . . . .     | 7  |
| dimred_pca . . . . .                 | 8  |
| dimred_pcoa . . . . .                | 9  |
| dimred_tsne . . . . .                | 10 |
| dimred_umap . . . . .                | 11 |
| diversities . . . . .                | 12 |
| diversities_help . . . . .           | 13 |
| diversity_beta_boxplot . . . . .     | 14 |
| diversity_beta_heatmap . . . . .     | 15 |
| diversity_beta_NMDS . . . . .        | 16 |
| diversity_beta_test . . . . .        | 16 |
| do_alpha_div_test . . . . .          | 17 |
| filter_categorize . . . . .          | 18 |
| filter_summary_bar_density . . . . . | 19 |
| filter_summary_pie_box . . . . .     | 20 |
| find_biomarker . . . . .             | 21 |
| find_taxonomy . . . . .              | 22 |
| find_taxonomy_300 . . . . .          | 23 |
| find_taxon_mat . . . . .             | 23 |
| gini_simpson . . . . .               | 24 |
| grep_tid . . . . .                   | 24 |
| inverse_simpson . . . . .            | 25 |
| is_categorical . . . . .             | 25 |
| is_integer0 . . . . .                | 26 |
| is_integer1 . . . . .                | 26 |
| mae_pick_organisms . . . . .         | 27 |
| mae_pick_samples . . . . .           | 28 |
| pct2str . . . . .                    | 28 |

|                                |    |
|--------------------------------|----|
| percent . . . . .              | 29 |
| read_pathoscope_data . . . . . | 30 |
| relabu_barplot . . . . .       | 30 |
| relabu_boxplot . . . . .       | 32 |
| relabu_heatmap . . . . .       | 33 |
| run_animalcules . . . . .      | 34 |
| shannon . . . . .              | 34 |
| simpson_index . . . . .        | 35 |
| upsample_counts . . . . .      | 35 |
| write_to_biom . . . . .        | 36 |

|              |           |
|--------------|-----------|
| <b>Index</b> | <b>37</b> |
|--------------|-----------|

---

|                     |                                                             |
|---------------------|-------------------------------------------------------------|
| animalcules-package | <i>animalcules: Interactive microbiome analysis toolkit</i> |
|---------------------|-------------------------------------------------------------|

---

## Description

animalcules is an R package for utilizing up-to-date data analytics, visualization methods, and machine learning models to provide users an easy-to-use interactive microbiome analysis framework. It can be used as a standalone software package or users can explore their data with the accompanying interactive R Shiny application. Traditional microbiome analysis such as alpha/beta diversity and differential abundance analysis are enhanced, while new methods like biomarker identification are introduced by animalcules. Powerful interactive and dynamic figures generated by animalcules enable users to understand their data better and discover new insights.

## Author(s)

**Maintainer:** Jessica McClintock <jessica.mcclintock@rutgers.edu>

Authors:

- Yue Zhao <yuezh@bu.edu> ([ORCID](#))
- Anthony Federico <anfed@bu.edu> ([ORCID](#))
- W. Evan Johnson <wej@bu.edu> ([ORCID](#))

## See Also

Useful links:

- <https://github.com/wejlab/animalcules>
- Report bugs at <https://github.com/wejlab/animalcules/issues>

---

|                   |                                |
|-------------------|--------------------------------|
| alpha_div_boxplot | <i>Alpha diversity boxplot</i> |
|-------------------|--------------------------------|

---

## Description

Alpha diversity boxplot

## Usage

```
alpha_div_boxplot(
  MAE,
  tax_level,
  condition,
  alpha_metric = c("inverse_simpson", "gini_simpson", "shannon", "fisher", "coverage",
    "unit")
)
```

## Arguments

|              |                                                                                                                                          |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------|
| MAE          | A multi-assay experiment object. Required.                                                                                               |
| tax_level    | The taxon level used for organisms. Required.                                                                                            |
| condition    | Which condition to group samples. Required.                                                                                              |
| alpha_metric | Which alpha diversity metric to use. Required. Can be one of: "inverse_simpson", "gini_simpson", "shannon", "fisher", "coverage", "unit" |

## Value

A plotly object

## Examples

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
p <- alpha_div_boxplot(toy_data,
  tax_level = "genus",
  condition = "DISEASE",
  alpha_metric = "shannon"
)
p
```

---

|                |                            |
|----------------|----------------------------|
| alpha_div_test | <i>Get alpha diversity</i> |
|----------------|----------------------------|

---

**Description**

Get alpha diversity

**Usage**

```
alpha_div_test(sam_table, alpha_stat)
```

**Arguments**

|            |                                                            |
|------------|------------------------------------------------------------|
| sam_table  | A dataframe with 2 cols, richness and condition. Required. |
| alpha_stat | Wilcoxon rank sum test or T-test for the test              |

**Value**

A dataframe

**Examples**

```
df_test <- data.frame(
  richness = seq_len(10),
  condition = c(rep(1, 5), rep(0, 5))
)
alpha_div_test(df_test, alpha_stat = "Wilcoxon rank sum test")
```

---

|                  |                                                             |
|------------------|-------------------------------------------------------------|
| counts_to_logcpm | <i>Covert a counts table to a relative abundances table</i> |
|------------------|-------------------------------------------------------------|

---

**Description**

Covert a counts table to a relative abundances table

**Usage**

```
counts_to_logcpm(counts_table)
```

**Arguments**

|              |                                          |
|--------------|------------------------------------------|
| counts_table | A organism x sample data frame of counts |
|--------------|------------------------------------------|

**Value**

A organism x sample data frame of logcpm counts

**Examples**

```
logcpm <- counts_to_logcpm(as.data.frame(matrix(seq_len(12), 4)))
```

---

|                  |                                                             |
|------------------|-------------------------------------------------------------|
| counts_to_relabu | <i>Covert a counts table to a relative abundances table</i> |
|------------------|-------------------------------------------------------------|

---

**Description**

Covert a counts table to a relative abundances table

**Usage**

```
counts_to_relabu(counts_table)
```

**Arguments**

counts\_table     A organism x sample data frame of counts

**Value**

A organism x sample data frame of relative abundances

**Examples**

```
counts_to_relabu(matrix(seq_len(12), 4))
```

---

|                   |                                          |
|-------------------|------------------------------------------|
| df_char_to_factor | <i>Factorize all categorical columns</i> |
|-------------------|------------------------------------------|

---

**Description**

Factorize all categorical columns

**Usage**

```
df_char_to_factor(df)
```

**Arguments**

df                     A sample x condition data frame

**Value**

A sample x condition data frame

**Examples**

```
df_char_to_factor(matrix(seq_len(12)))
```

---

differential\_abundance

*Differential abundance analysis*

---

**Description**

Differential abundance analysis

**Usage**

```
differential_abundance(  
  MAE,  
  tax_level,  
  input_da_condition = c(),  
  input_da_condition_covariate = NULL,  
  min_num_filter = 5,  
  input_da_padj_cutoff = 0.05,  
  method = "DESeq2"  
)
```

**Arguments**

|                              |                                             |
|------------------------------|---------------------------------------------|
| MAE                          | A multi-assay experiment object             |
| tax_level                    | The taxon level used for organisms          |
| input_da_condition           | Which condition is the target condition     |
| input_da_condition_covariate | Covariates added to linear function         |
| min_num_filter               | Minimum number reads mapped to this microbe |
| input_da_padj_cutoff         | adjusted pValue cutoff                      |
| method                       | choose between DESeq2 and limma             |

**Value**

A output dataframe

**Examples**

```

data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
differential_abundance(toy_data,
  tax_level = "phylum",
  input_da_condition = c("DISEASE"),
  min_num_filter = 2,
  input_da_padj_cutoff = 0.5,
  method = "DESeq2"
)

```

dimred\_pca

*Dimensionality reduction through PCA***Description**

Dimensionality reduction through PCA

**Usage**

```

dimred_pca(
  MAE,
  tax_level,
  color,
  shape = NULL,
  pcx = 1,
  pcy = 2,
  pcz = NULL,
  datatype = c("logcpm", "relabu", "counts")
)

```

**Arguments**

|           |                                                      |
|-----------|------------------------------------------------------|
| MAE       | A multi-assay experiment object                      |
| tax_level | The taxon level used for organisms                   |
| color     | A condition to color data points by e.g. "AGE"       |
| shape     | A condition to shape data points by e.g. "SEX"       |
| pcx       | Principal component on the x-axis e.g. 1             |
| pcy       | Principal component on the y-axis e.g. 2             |
| pcz       | Principal component on the z-axis e.g. 3             |
| datatype  | Datatype to use e.g. c("logcpm", "relabu", "counts") |

**Value**

A list with a plotly object and summary table

**Examples**

```

data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
result <- dimred_pca(toy_data,
  tax_level = "genus",
  color = "AGE",
  shape = "DISEASE",
  pcx = 1,
  pcy = 2,
  datatype = "logcpm"
)
result$plot
result$table

```

dimred\_pcoa

*Dimensionality reduction through PCoA***Description**

Dimensionality reduction through PCoA

**Usage**

```

dimred_pcoa(
  MAE,
  tax_level,
  color,
  shape = NULL,
  axx = 1,
  axy = 2,
  axz = NULL,
  method = c("bray", "jaccard")
)

```

**Arguments**

|           |                                                |
|-----------|------------------------------------------------|
| MAE       | A multi-assay experiment object                |
| tax_level | The taxon level used for organisms             |
| color     | A condition to color data points by e.g. "AGE" |
| shape     | A condition to shape data points by e.g. "SEX" |
| axx       | Principle coordinate on the x-axis e.g. 1      |
| axy       | Principle coordinate on the y-axis e.g. 2      |
| axz       | Principle coordinate on the z-axis e.g. 2      |
| method    | Method to use e.g. c("bray", "jaccard")        |

**Value**

A list with a plotly object and summary table

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
result <- dimred_pcoa(toy_data,
  tax_level = "genus",
  color = "AGE",
  shape = "DISEASE",
  axx = 1,
  axy = 2,
  method = "bray"
)
result$plot
result$table
```

---

|             |                                               |
|-------------|-----------------------------------------------|
| dimred_tsne | <i>Dimensionality reduction through t-SNE</i> |
|-------------|-----------------------------------------------|

---

**Description**

Dimensionality reduction through t-SNE

**Usage**

```
dimred_tsne(
  MAE,
  tax_level,
  color,
  shape = NULL,
  k = c("2D", "3D"),
  initial_dims = 30,
  perplexity = 10,
  datatype = c("logcpm", "relabu", "counts"),
  tsne_cache = NULL
)
```

**Arguments**

|           |                                                |
|-----------|------------------------------------------------|
| MAE       | A multi-assay experiment object                |
| tax_level | The taxon level used for organisms             |
| color     | A condition to color data points by e.g. "AGE" |
| shape     | A condition to shape data points by e.g. "SEX" |
| k         | Plot dimensions e.g. c("2D","3D")              |

|              |                                                      |
|--------------|------------------------------------------------------|
| initial_dims | The number of dimensions to use in reduction method  |
| perplexity   | Optimal number of neighbors                          |
| datatype     | Datatype to use e.g. c("logcpm", "relabu", "counts") |
| tsne_cache   | Pass the cached data back into the function          |

**Value**

A list with a plotly object and cached data

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
results <- dimred_tsne(toy_data,
  tax_level = "phylum",
  color = "AGE",
  shape = "GROUP",
  k = "3D",
  initial_dims = 30,
  perplexity = 10,
  datatype = "logcpm"
)
results$plot
```

---

dimred\_umap

---

*Dimensionality reduction through PCA*


---

**Description**

Dimensionality reduction through PCA

**Usage**

```
dimred_umap(
  MAE,
  tax_level,
  color,
  shape = NULL,
  cx = 1,
  cy = 2,
  cz = NULL,
  n_neighbors = 15,
  metric = c("euclidean", "manhattan"),
  n_epochs = 200,
  init = c("spectral", "random"),
  min_dist = 0.1,
  datatype = c("logcpm", "relabu", "counts")
)
```

Arguments

|             |                                                                 |
|-------------|-----------------------------------------------------------------|
| MAE         | A multi-assay experiment object                                 |
| tax_level   | The taxon level used for organisms                              |
| color       | A condition to color data points by e.g. "AGE"                  |
| shape       | A condition to shape data points by e.g. "SEX"                  |
| cx          | Component on the x-axis e.g. 1                                  |
| cy          | Component on the y-axis e.g. 2                                  |
| cz          | Component on the z-axis e.g. 3                                  |
| n_neighbors | Number of nearest neighbors                                     |
| metric      | Distance function e.g. c("euclidean", "manhattan")              |
| n_epochs    | Number of iterations                                            |
| init        | Initial embedding using eigenvector e.g c("spectral", "random") |
| min_dist    | Determines how close points appear in the final layout          |
| datatype    | Datatype to use e.g. c("logcpm", "relabu", "counts")            |

Value

A list with a plotly object and summary table

Examples

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
result <- dimred_umap(toy_data,
  tax_level = "genus",
  color = "AGE",
  shape = "DISEASE",
  cx = 1,
  cy = 2,
  datatype = "logcpm"
)
result$plot
```

---

|             |                            |
|-------------|----------------------------|
| diversities | <i>Get alpha diversity</i> |
|-------------|----------------------------|

---

Description

Get alpha diversity

Usage

```
diversities(counts_table, index = "all", zeroes = TRUE)
```

**Arguments**

|              |                                                                  |
|--------------|------------------------------------------------------------------|
| counts_table | A dataframe with organism x sample                               |
| index        | One of inverse_simpson,gini_simpson,shannon,fisher,coverage,unit |
| zeroes       | A boolean for whether to ignore zero values                      |

**Value**

A list of alpha diversity

**Examples**

```
diversities(matrix(seq_len(12), nrow = 3), index = "shannon")
```

---

|                  |                            |
|------------------|----------------------------|
| diversities_help | <i>Get alpha diversity</i> |
|------------------|----------------------------|

---

**Description**

Get alpha diversity

**Usage**

```
diversities_help(counts_table, index = "all", zeroes = TRUE)
```

**Arguments**

|              |                                                                  |
|--------------|------------------------------------------------------------------|
| counts_table | A dataframe with organism x sample                               |
| index        | one of inverse_simpson,gini_simpson,shannon,fisher,coverage,unit |
| zeroes       | A boolean for whether to ignore zero values                      |

**Value**

A list of alpha diversity

**Examples**

```
diversities_help(matrix(seq_len(12), nrow = 3), index = "shannon")
```

---

`diversity_beta_boxplot`*Beta diversity boxplot*

---

## Description

Beta diversity boxplot

## Usage

```
diversity_beta_boxplot(  
  MAE,  
  tax_level,  
  input_beta_method,  
  input_select_beta_condition  
)
```

## Arguments

|                             |                                    |
|-----------------------------|------------------------------------|
| MAE                         | A multi-assay experiment object    |
| tax_level                   | The taxon level used for organisms |
| input_beta_method           | bray, jaccard                      |
| input_select_beta_condition | Which condition to group samples   |

## Value

A plotly object

## Examples

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")  
toy_data <- readRDS(data_dir)  
p <- diversity_beta_boxplot(toy_data,  
  tax_level = "genus",  
  input_beta_method = "bray",  
  input_select_beta_condition = "DISEASE"  
)  
p
```

---

`diversity_beta_heatmap`*Beta diversity heatmap*

---

## Description

Beta diversity heatmap

## Usage

```
diversity_beta_heatmap(  
  MAE,  
  tax_level,  
  input_beta_method,  
  input_bdhm_select_conditions,  
  input_bdhm_sort_by = c("nosort", "conditions")  
)
```

## Arguments

|                              |                                            |
|------------------------------|--------------------------------------------|
| MAE                          | A multi-assay experiment object            |
| tax_level                    | The taxon level used for organisms         |
| input_beta_method            | bray, jaccard                              |
| input_bdhm_select_conditions | Which condition to group samples           |
| input_bdhm_sort_by           | Sorting option e.g. "nosort", "conditions" |

## Value

A plotly object

## Examples

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")  
toy_data <- readRDS(data_dir)  
p <- diversity_beta_heatmap(toy_data,  
  tax_level = "genus",  
  input_beta_method = "bray",  
  input_bdhm_select_conditions = "DISEASE",  
  input_bdhm_sort_by = "conditions"  
)  
p
```

---

|                     |                                 |
|---------------------|---------------------------------|
| diversity_beta_NMDS | <i>Beta diversity NMDS plot</i> |
|---------------------|---------------------------------|

---

**Description**

Beta diversity NMDS plot

**Usage**

```
diversity_beta_NMDS(  
  MAE,  
  tax_level,  
  input_beta_method,  
  input_select_beta_condition  
)
```

**Arguments**

|                             |                                    |
|-----------------------------|------------------------------------|
| MAE                         | A multi-assay experiment object    |
| tax_level                   | The taxon level used for organisms |
| input_beta_method           | bray, jaccard                      |
| input_select_beta_condition | Which condition to group samples   |

**Value**

A plotly object/NMDS plot

---

|                     |                                      |
|---------------------|--------------------------------------|
| diversity_beta_test | <i>Perform a beta diversity test</i> |
|---------------------|--------------------------------------|

---

**Description**

Perform a beta diversity test

**Usage**

```
diversity_beta_test(  
  MAE,  
  tax_level,  
  input_beta_method,  
  input_select_beta_condition,  
  input_select_beta_stat_method,  
  input_num_permutation_permanova = 999  
)
```

**Arguments**

MAE                    A Multi-Assay Experiment object. Required.

tax\_level            The taxon level at which organisms should be grouped. Req'd.

input\_beta\_method            Can be either "bray" or "jaccard". Required.

input\_select\_beta\_condition            Condition to group samples Should be a character string of a colData column name. Required.

input\_select\_beta\_stat\_method            The test to be used. Can be one of either "PERMANOVA", "Wilcoxon rank sum test", or "Kruskal-Wallis". Required.

input\_num\_permutation\_permanova            The number of permutations to be used.

**Value**

A plotly object.

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
p <- diversity_beta_test(toy_data,
  tax_level = "genus",
  input_beta_method = "bray",
  input_select_beta_condition = "DISEASE",
  input_select_beta_stat_method = "Wilcoxon rank sum test",
  input_num_permutation_permanova = 999
)
p
```

---

|                   |                                         |
|-------------------|-----------------------------------------|
| do_alpha_div_test | <i>Alpha diversity statistical test</i> |
|-------------------|-----------------------------------------|

---

**Description**

Alpha diversity statistical test

**Usage**

```
do_alpha_div_test(
  MAE,
  tax_level,
  condition,
  alpha_metric = c("inverse_simpson", "gini_simpson", "shannon", "fisher", "coverage",
    "unit"),
  alpha_stat = c("Wilcoxon rank sum test", "T-test", "Kruskal-Wallis")
)
```

**Arguments**

|              |                                     |
|--------------|-------------------------------------|
| MAE          | A multi-assay experiment object     |
| tax_level    | The taxon level used for organisms  |
| condition    | Which condition to group samples    |
| alpha_metric | Which alpha diversity metric to use |
| alpha_stat   | Which stat test to use              |

**Value**

A dataframe

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
p <- do_alpha_div_test(toy_data,
  tax_level = "genus",
  condition = "DISEASE",
  alpha_metric = "shannon",
  alpha_stat = "Wilcoxon rank sum test"
)
p
```

---

|                   |                                        |
|-------------------|----------------------------------------|
| filter_categorize | <i>Categorize continuous variables</i> |
|-------------------|----------------------------------------|

---

**Description**

Categorize continuous variables

**Usage**

```
filter_categorize(
  sam_table,
  sample_condition,
  new_label,
  nbins = NULL,
  bin_breaks = c(),
  bin_labels = c()
)
```

**Arguments**

|                               |                                            |
|-------------------------------|--------------------------------------------|
| <code>sam_table</code>        | A sample x condition dataframe             |
| <code>sample_condition</code> | Continuous variable to categorize          |
| <code>new_label</code>        | Column name for categorized variable       |
| <code>nbins</code>            | Auto select ranges for n bins/categories   |
| <code>bin_breaks</code>       | Manually select ranges for bins/categories |
| <code>bin_labels</code>       | Manually label bins/categories             |

**Value**

A list with an updated sample table and before/after plots

**Examples**

```
library(SummarizedExperiment)
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
microbe <- MultiAssayExperiment::experiments(toy_data)[[1]]
samples <- as.data.frame(colData(microbe))
result <- filter_categorize(samples,
  sample_condition = "AGE",
  new_label = "AGE_GROUP",
  bin_breaks = c(0, 55, 75, 100),
  bin_labels = c("Young", "Adult", "Elderly")
)
result$sam_table
result$plot.unbinned
result$plot.binned
```

---

filter\_summary\_bar\_density

*Data visualization by bar plot / density plot*

---

**Description**

Data visualization by bar plot / density plot

**Usage**

```
filter_summary_bar_density(
  MAE,
  samples_discard = NULL,
  filter_type,
  sample_condition
)
```

**Arguments**

MAE                    A multi-assay experiment object  
 samples\_discard        The list of samples to filter  
 filter\_type            Either 'By Microbes' or 'By Metadata'  
 sample\_condition       Which condition to check e.g. 'SEX'

**Value**

A plotly object

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
result <- filter_summary_bar_density(toy_data,
  samples_discard = c("subject_2", "subject_4"),
  filter_type = "By Metadata",
  sample_condition = "SEX"
)
result
```

---

filter\_summary\_pie\_box

*Data visualization by pie chart / box plot*

---

**Description**

Data visualization by pie chart / box plot

**Usage**

```
filter_summary_pie_box(
  MAE,
  samples_discard = NULL,
  filter_type,
  sample_condition
)
```

**Arguments**

MAE                    A multi-assay experiment object  
 samples\_discard        The list of samples to filter  
 filter\_type            Either 'By Microbes' or 'By Metadata'  
 sample\_condition       Which condition to check e.g. 'SEX'

**Value**

A plotly object

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
result <- filter_summary_pie_box(toy_data,
  samples_discard = c("subject_2", "subject_4"),
  filter_type = "By Microbes",
  sample_condition = "SEX"
)
result
```

---

|                |                            |
|----------------|----------------------------|
| find_biomarker | <i>Identify biomarkers</i> |
|----------------|----------------------------|

---

**Description**

Identify biomarkers

**Usage**

```
find_biomarker(
  MAE,
  tax_level,
  input_select_target_biomarker,
  nfolds = 3,
  nrepeats = 3,
  seed = 99,
  percent_top_biomarker = 0.2,
  model_name = c("logistic regression", "random forest")
)
```

**Arguments**

|                               |                                               |
|-------------------------------|-----------------------------------------------|
| MAE                           | A multi-assay experiment object               |
| tax_level                     | The taxon level used for organisms            |
| input_select_target_biomarker | Which condition is the target condition       |
| nfolds                        | number of splits in CV                        |
| nrepeats                      | number of CVs with different random splits    |
| seed                          | for repeatable research                       |
| percent_top_biomarker         | Top importance percentage to pick biomarker   |
| model_name                    | one of 'logistic regression', 'random forest' |

**Value**

A list

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
p <- find_biomarker(toy_data,
  tax_level = "family",
  input_select_target_biomarker = c("DISEASE"),
  nfolds = 3,
  nrepeats = 3,
  seed = 99,
  percent_top_biomarker = 0.2,
  model_name = "logistic regression"
)
p
```

---

find\_taxonomy

*Find the taxonomy for unlimited tids*

---

**Description**

Find the taxonomy for unlimited tids

**Usage**

```
find_taxonomy(tids)
```

**Arguments**

tids                      Given taxonomy ids

**Value**

A list of taxon levels with information

**Examples**

```
taxonLevels <- find_taxonomy(tids = 1200)
```

---

|                   |                                               |
|-------------------|-----------------------------------------------|
| find_taxonomy_300 | <i>Find the taxonomy for maximum 300 tids</i> |
|-------------------|-----------------------------------------------|

---

**Description**

Find the taxonomy for maximum 300 tids

**Usage**

```
find_taxonomy_300(tids)
```

**Arguments**

|      |                    |
|------|--------------------|
| tids | Given taxonomy ids |
|------|--------------------|

**Value**

taxondata Data with the taxonomy information

**Examples**

```
taxonLevels <- find_taxonomy_300(tids = 1200)
```

---

|                |                                             |
|----------------|---------------------------------------------|
| find_taxon_mat | <i>Find the Taxonomy Information Matrix</i> |
|----------------|---------------------------------------------|

---

**Description**

Find the Taxonomy Information Matrix

**Usage**

```
find_taxon_mat(names, taxonLevels)
```

**Arguments**

|             |                                  |
|-------------|----------------------------------|
| names       | Row names of the taxonomy matrix |
| taxonLevels | Taxon Levels of all tids         |

**Value**

taxmat Taxonomy Information Matrix

Examples

```
ids <- c("ti|54005", "ti|73001", "ti|573", "ti|228277", "ti|53458")
tids <- c("54005", "73001", "573", "228277", "53458")
taxonLevels <- find_taxonomy(tids)
tax_table <- find_taxon_mat(ids, taxonLevels)
```

---

|              |                                       |
|--------------|---------------------------------------|
| gini_simpson | <i>Get alpha diversity using gini</i> |
|--------------|---------------------------------------|

---

Description

Get alpha diversity using gini

Usage

```
gini_simpson(x)
```

Arguments

x                      A list of counts

Value

A single value

Examples

```
gini_simpson(seq_len(10))
```

---

|          |                                                       |
|----------|-------------------------------------------------------|
| grep_tid | <i>Greps the tid from the given identifier string</i> |
|----------|-------------------------------------------------------|

---

Description

Greps the tid from the given identifier string

Usage

```
grep_tid(id)
```

Arguments

id                      Given identifier string

**Value**

tid string

**Examples**

```
grep_tid("ti|700015|org|Coriobacterium_glomerans_PW2")
```

---

|                 |                                                  |
|-----------------|--------------------------------------------------|
| inverse_simpson | <i>Get alpha diversity using inverse simpson</i> |
|-----------------|--------------------------------------------------|

---

**Description**

Get alpha diversity using inverse simpson

**Usage**

```
inverse_simpson(x)
```

**Arguments**

|   |                  |
|---|------------------|
| x | A list of counts |
|---|------------------|

**Value**

A single value

**Examples**

```
inverse_simpson(seq_len(10))
```

---

|                |                                       |
|----------------|---------------------------------------|
| is_categorical | <i>Check if object is categorical</i> |
|----------------|---------------------------------------|

---

**Description**

Check if object is categorical

**Usage**

```
is_categorical(v)
```

**Arguments**

|   |                |
|---|----------------|
| v | A single value |
|---|----------------|

**Value**

Boolean

**Examples**

```
nums <- 2
is_categorical(nums)
```

---

|             |                            |
|-------------|----------------------------|
| is_integer0 | <i>check if integer(0)</i> |
|-------------|----------------------------|

---

**Description**

check if integer(0)

**Usage**

```
is_integer0(x)
```

**Arguments**

x                      A single value

**Value**

Boolean

**Examples**

```
nums <- 2
is_integer0(nums)
```

---

|             |                            |
|-------------|----------------------------|
| is_integer1 | <i>check if integer(1)</i> |
|-------------|----------------------------|

---

**Description**

check if integer(1)

**Usage**

```
is_integer1(x)
```

**Arguments**

x                      A single value

**Value**

Boolean

**Examples**

```
nums <- 2
is_integer1(nums)
```

---

|                    |                                                          |
|--------------------|----------------------------------------------------------|
| mae_pick_organisms | <i>Modify organisms of multi-assay experiment object</i> |
|--------------------|----------------------------------------------------------|

---

**Description**

Modify organisms of multi-assay experiment object

**Usage**

```
mae_pick_organisms(MAE, isolate_organisms = NULL, discard_organisms = NULL)
```

**Arguments**

|                   |                                                |
|-------------------|------------------------------------------------|
| MAE               | A multi-assay experiment object                |
| isolate_organisms | Isolate specific organisms e.g. til001, til002 |
| discard_organisms | Discard specific organisms e.g. til001, til002 |

**Value**

A multi-assay experiment object

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
subset <- mae_pick_organisms(toy_data,
  isolate_organisms = c("ti|001", "ti|002")
)
```

---

|                  |                                                        |
|------------------|--------------------------------------------------------|
| mae_pick_samples | <i>Modify samples of multi-assay experiment object</i> |
|------------------|--------------------------------------------------------|

---

**Description**

Modify samples of multi-assay experiment object

**Usage**

```
mae_pick_samples(MAE, isolate_samples = NULL, discard_samples = NULL)
```

**Arguments**

|                 |                                                     |
|-----------------|-----------------------------------------------------|
| MAE             | A multi-assay experiment object                     |
| isolate_samples | Isolate specific samples e.g. c('SAM_01', 'SAM_02') |
| discard_samples | Discard specific samples e.g. c('SAM_01', 'SAM_02') |

**Value**

A multi-assay experiment object

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
subset <- mae_pick_samples(toy_data,
  isolate_samples = c(
    "subject_9",
    "subject_14"
  )
)
```

---

|         |                                                                    |
|---------|--------------------------------------------------------------------|
| pct2str | <i>Converts decimal percentage to string with specified digits</i> |
|---------|--------------------------------------------------------------------|

---

**Description**

Converts decimal percentage to string with specified digits

**Usage**

```
pct2str(v, digits = 2)
```

**Arguments**

|        |                  |
|--------|------------------|
| v      | A single value   |
| digits | number of digits |

**Value**

Boolean

**Examples**

```
nums <- 0.23
pct2str(nums)
```

---

|         |                                       |
|---------|---------------------------------------|
| percent | <i>Format decimals to percentages</i> |
|---------|---------------------------------------|

---

**Description**

Format decimals to percentages

**Usage**

```
percent(x, digits = 2, format = "f")
```

**Arguments**

|        |                      |
|--------|----------------------|
| x      | An array of decimals |
| digits | number of digits     |
| format | f                    |

**Value**

An array of formatted strings

**Examples**

```
nums <- c(0.42, 0.15, 0.4, 0.563, 0.2)
percent(nums)
```

---

|                      |                                                                                                                   |
|----------------------|-------------------------------------------------------------------------------------------------------------------|
| read_pathoscope_data | <i>Reads the data from PathoScope reports and returns a list of final guess relative abundance and count data</i> |
|----------------------|-------------------------------------------------------------------------------------------------------------------|

---

### Description

Reads the data from PathoScope reports and returns a list of final guess relative abundance and count data

### Usage

```
read_pathoscope_data(
  input_dir = ".",
  pathoreport_file_suffix = "-sam-report.tsv",
  use.input.files = FALSE,
  input.files.path.vec = NULL,
  input.files.name.vec = NULL
)
```

### Arguments

|                         |                                                                    |
|-------------------------|--------------------------------------------------------------------|
| input_dir               | Directory where the tsv files from PathoScope are located          |
| pathoreport_file_suffix | PathoScope report files suffix                                     |
| use.input.files         | whether input dir to pathoscope files or directly pathoscope files |
| input.files.path.vec    | vector of pathoscope file paths                                    |
| input.files.name.vec    | vector of pathoscope file names                                    |

### Value

List of final guess relative abundance and count data

---

|                |                                                                    |
|----------------|--------------------------------------------------------------------|
| relabu_barplot | <i>Plot bar plots of sample and group level relative abundance</i> |
|----------------|--------------------------------------------------------------------|

---

### Description

Plot bar plots of sample and group level relative abundance

**Usage**

```
relabu_barplot(
  MAE,
  tax_level,
  order_organisms = c(),
  sort_by = c("nosort", "conditions", "organisms", "alphabetically"),
  group_samples = FALSE,
  group_conditions = "ALL",
  sample_conditions = c(),
  isolate_samples = c(),
  discard_samples = c(),
  show_legend = TRUE
)
```

**Arguments**

|                   |                                                                              |
|-------------------|------------------------------------------------------------------------------|
| MAE               | A multi-assay experiment object                                              |
| tax_level         | The taxon level used for organisms                                           |
| order_organisms   | A character list of organisms to send to top                                 |
| sort_by           | Sort bars by one of c("nosort", "conditions", "organisms", "alphabetically") |
| group_samples     | A bool specifying whether to group samples                                   |
| group_conditions  | Group by one or more conditions e.g. "ALL" or "SEX"                          |
| sample_conditions | Plot associated conditions with samples.                                     |
| isolate_samples   | Isolate specific samples e.g. c("SAM_01", "SAM_02")                          |
| discard_samples   | Discard specific samples e.g. c("SAM_01", "SAM_02")                          |
| show_legend       | A bool specifying whether or not to show organisms legend                    |

**Value**

A plotly object

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
p <- relabu_barplot(toy_data,
  tax_level = "family",
  order_organisms = c("Retroviridae"),
  sort_by = "organisms",
  sample_conditions = c("SEX", "AGE"),
  show_legend = TRUE
)
```

p

---

|                |                                                                                |
|----------------|--------------------------------------------------------------------------------|
| relabu_boxplot | <i>Plot boxplots comparing different organism prevalence across conditions</i> |
|----------------|--------------------------------------------------------------------------------|

---

## Description

Plot boxplots comparing different organism prevalence across conditions

## Usage

```
relabu_boxplot(
  MAE,
  tax_level,
  condition,
  organisms = c(),
  datatype = c("counts", "relative abundance", "logcpm")
)
```

## Arguments

|           |                                        |
|-----------|----------------------------------------|
| MAE       | A multi-assay experiment object        |
| tax_level | The taxon level used for organisms     |
| condition | Compare groups by condition e.g. 'SEX' |
| organisms | Include organisms for plotting.        |
| datatype  | counts, relative abundance, logcpm     |

## Value

A plotly object

## Examples

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
p <- relabu_boxplot(toy_data,
  tax_level = "genus",
  organisms = c("Escherichia", "Actinomyces"),
  condition = "SEX",
  datatype = "logcpm"
)
p
```

---

|                |                                                      |
|----------------|------------------------------------------------------|
| relabu_heatmap | <i>Plot heatmap of sample level counts in logcpm</i> |
|----------------|------------------------------------------------------|

---

**Description**

Plot heatmap of sample level counts in logcpm

**Usage**

```
relabu_heatmap(
  MAE,
  tax_level,
  sort_by = c("nosort", "conditions", "organisms", "alphabetically"),
  sample_conditions = c(),
  isolate_organisms = c(),
  isolate_samples = c(),
  discard_samples = c(),
  log_cpm = TRUE
)
```

**Arguments**

|                   |                                                                              |
|-------------------|------------------------------------------------------------------------------|
| MAE               | A multi-assay experiment object                                              |
| tax_level         | The taxon level used for organisms                                           |
| sort_by           | Sort bars by one of c('nosort', 'conditions', 'organisms', 'alphabetically') |
| sample_conditions | Plot conditions e.g. c('SEX', 'AGE')                                         |
| isolate_organisms | Isolate specific organisms e.g. c('Hepacivirus')                             |
| isolate_samples   | Isolate specific samples e.g. c('SAM_01', 'SAM_02')                          |
| discard_samples   | Discard specific samples e.g. c('SAM_01', 'SAM_02')                          |
| log_cpm           | Convert counts to logcpm                                                     |

**Value**

A plotly object

**Examples**

```
data_dir <- system.file("extdata/MAE.rds", package = "animalcules")
toy_data <- readRDS(data_dir)
p <- relabu_heatmap(toy_data,
  tax_level = "genus",
  sort_by = "conditions",
```

```
    sample_conditions = c("SEX", "AGE")
  )
p
```

---

|                 |                                  |
|-----------------|----------------------------------|
| run_animalcules | <i>Run animalcules shiny app</i> |
|-----------------|----------------------------------|

---

**Description**

Run animalcules shiny app

**Usage**

```
run_animalcules(dev = FALSE)
```

**Arguments**

|     |                                       |
|-----|---------------------------------------|
| dev | Run the applicaiton in developer mode |
|-----|---------------------------------------|

**Value**

The shiny app will open

**Examples**

```
## Not run:
run_animalcules()

## End(Not run)
```

---

|         |                                          |
|---------|------------------------------------------|
| shannon | <i>Get alpha diversity using shannon</i> |
|---------|------------------------------------------|

---

**Description**

Get alpha diversity using shannon

**Usage**

```
shannon(x)
```

**Arguments**

|   |                  |
|---|------------------|
| x | A list of counts |
|---|------------------|

**Value**

A single value

**Examples**

shannon(seq\_len(10))

---

|               |                                          |
|---------------|------------------------------------------|
| simpson_index | <i>Get alpha diversity using simpson</i> |
|---------------|------------------------------------------|

---

**Description**

Get alpha diversity using simpson

**Usage**

simpson\_index(x)

**Arguments**

x                      A list of counts

**Value**

A single value

**Examples**

simpson\_index(seq\_len(10))

---

|                 |                                                        |
|-----------------|--------------------------------------------------------|
| upsample_counts | <i>Upsample a counts table to a higher taxon level</i> |
|-----------------|--------------------------------------------------------|

---

**Description**

Upsample a counts table to a higher taxon level

**Usage**

upsample\_counts(counts\_table, tax\_table, higher\_level)

Arguments

- counts\_table     A organism x sample data frame of counts
- tax\_table        A organism x taxlev data frame of labels
- higher\_level     Higher taxon level to upsample to

Value

A organism x sample data frame of counts

Examples

```
toy_data <- readRDS(system.file("extdata/toy_data.rds",
package = "animalcules"))
tax_table <- toy_data$tax_table
sam_table <- toy_data$sam_table
counts_table <- toy_data$counts_table
counts_table <- upsample_counts(counts_table, tax_table, "phylum")
```

---

|               |                    |
|---------------|--------------------|
| write_to_biom | <i>Output biom</i> |
|---------------|--------------------|

---

Description

Output biom

Usage

```
write_to_biom(MAE, path_to_output)
```

Arguments

- MAE              A multi-assay experiment object
- path\_to\_output   The folder to output biom file

Value

A message

# Index

## \* **internal**

- animalcules-package, [3](#)
  
- [alpha\\_div\\_boxplot, 4](#)
- [alpha\\_div\\_test, 5](#)
- [animalcules \(animalcules-package\), 3](#)
- [animalcules-package, 3](#)
  
- [counts\\_to\\_logcpm, 5](#)
- [counts\\_to\\_relabu, 6](#)
  
- [df\\_char\\_to\\_factor, 6](#)
- [differential\\_abundance, 7](#)
- [dimred\\_pca, 8](#)
- [dimred\\_pcoa, 9](#)
- [dimred\\_tsne, 10](#)
- [dimred\\_umap, 11](#)
- [diversities, 12](#)
- [diversities\\_help, 13](#)
- [diversity\\_beta\\_boxplot, 14](#)
- [diversity\\_beta\\_heatmap, 15](#)
- [diversity\\_beta\\_NMDS, 16](#)
- [diversity\\_beta\\_test, 16](#)
- [do\\_alpha\\_div\\_test, 17](#)
  
- [filter\\_categorize, 18](#)
- [filter\\_summary\\_bar\\_density, 19](#)
- [filter\\_summary\\_pie\\_box, 20](#)
- [find\\_biomarker, 21](#)
- [find\\_taxon\\_mat, 23](#)
- [find\\_taxonomy, 22](#)
- [find\\_taxonomy\\_300, 23](#)
  
- [gini\\_simpson, 24](#)
- [grep\\_tid, 24](#)
  
- [inverse\\_simpson, 25](#)
- [is\\_categorical, 25](#)
- [is\\_integer0, 26](#)
- [is\\_integer1, 26](#)
  
- [mae\\_pick\\_organisms, 27](#)
- [mae\\_pick\\_samples, 28](#)
  
- [pct2str, 28](#)
- [percent, 29](#)
  
- [read\\_pathoscope\\_data, 30](#)
- [relabu\\_barplot, 30](#)
- [relabu\\_boxplot, 32](#)
- [relabu\\_heatmap, 33](#)
- [run\\_animalcules, 34](#)
  
- [shannon, 34](#)
- [simpson\\_index, 35](#)
  
- [upsample\\_counts, 35](#)
  
- [write\\_to\\_biom, 36](#)