

# Package ‘animalcules’

January 25, 2022

**Title** Interactive microbiome analysis toolkit

**Version** 1.11.1

**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

**Encoding** UTF-8

**LazyData** false

**Depends** R (>= 4.0.0)

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

**Suggests** BiocStyle, knitr, rmarkdown, testthat, usethis

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

**VignetteBuilder** knitr

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

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

**RoxygenNote** 7.1.1

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

**git\_branch** master

**git\_last\_commit** 66921d7

**git\_last\_commit\_date** 2022-01-14

**Date/Publication** 2022-01-25

**Author** Yue Zhao [aut, cre] (<<https://orcid.org/0000-0001-5257-5103>>),  
 Anthony Federico [aut] (<<https://orcid.org/0000-0002-9200-1689>>),  
 W. Evan Johnson [aut] (<<https://orcid.org/0000-0002-6247-6595>>)

**Maintainer** Yue Zhao <yuezh@bu.edu>

## R topics documented:

alpha_div_boxplot . . . . .	3
alpha_div_test . . . . .	4
counts_to_logcpm . . . . .	4
counts_to_relabu . . . . .	5
df_char_to_factor . . . . .	5
differential_abundance . . . . .	6
dimred_pca . . . . .	7
dimred_pcoa . . . . .	8
dimred_tsne . . . . .	9
dimred_umap . . . . .	10
diversities . . . . .	11
diversities_help . . . . .	12
diversity_beta_boxplot . . . . .	13
diversity_beta_heatmap . . . . .	14
diversity_beta_test . . . . .	15
do_alpha_div_test . . . . .	16
filter_categorize . . . . .	17
filter_summary_bar_density . . . . .	18
filter_summary_pie_box . . . . .	19
find_biomarker . . . . .	20
find_taxonomy . . . . .	21
find_taxonomy_300 . . . . .	21
find_taxon_mat . . . . .	22
gini_simpson . . . . .	23
grep_tid . . . . .	23
inverse_simpson . . . . .	24
is_categorical . . . . .	24
is_integer0 . . . . .	25
is_integer1 . . . . .	25
mae_pick_organisms . . . . .	26
mae_pick_samples . . . . .	26
pct2str . . . . .	27
percent . . . . .	28
read_pathoscope_data . . . . .	28
relabu_barplot . . . . .	29
relabu_boxplot . . . . .	30

*alpha\_div\_boxplot* 3

relabu_heatmap . . . . .	31
run_animalcules . . . . .	32
shannon . . . . .	33
simpson_index . . . . .	33
upsample_counts . . . . .	34
write_to_biom . . . . .	35

**Index** 36

---

*alpha\_div\_boxplot*      *Alpha diversity boxplot*

---

### 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
tax_level	The taxon level used for organisms
condition	Which condition to group samples
alpha_metric	Which alpha diversity metric to use

### 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
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      *Covert a counts table to a relative abundances table*

---

**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      *Factorize all categorical columns*

---

**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_pcoa(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

*Dimensionality reduction through t-SNE*


---

**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
*Get alpha diversity*

---

**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      *Get alpha diversity*

---

**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**

<code>MAE</code>	A multi-assay experiment object
<code>tax_level</code>	The taxon level used for organisms
<code>input_beta_method</code>	bray, jaccard
<code>input_bdhm_select_conditions</code>	Which condition to group samples
<code>input_bdhm_sort_by</code>	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\_test     *Beta diversity test (by default we use bray-curtis distance)*

---

### Description

Beta diversity test (by default we use bray-curtis distance)

### 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
tax_level	The taxon level used for organisms
input_beta_method	bray, jaccard
input_select_beta_condition	Which condition to group samples
input_select_beta_stat_method	PERMANOVA, Kruskal-Wallis, Wilcoxon test
input_num_permutation_permanova	number of permutations

### 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 = 'PERMANOVA',
  input_num_permutation_permanova = 999)
p
```

---

do\_alpha\_div\_test      *Alpha diversity statistical test*

---

### 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**

sam_table	A sample x condition dataframe
sample_condition	Continuous variable to categorize
new_label	Column name for categorized variable
nbins	Auto select ranges for n bins/categories
bin_breaks	Manually select ranges for bins/categories
bin_labels	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      *Find the taxonomy for maximum 300 tids*

---

**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      *Modify organisms of multi-assay experiment object*

---

**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              *Modify samples of multi-assay experiment object*

---

**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

*Converts decimal percentage to string with specified digits*

---

**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**

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

[alpha\\_div\\_boxplot](#), 3  
[alpha\\_div\\_test](#), 4

[counts\\_to\\_logcpm](#), 4  
[counts\\_to\\_relabu](#), 5

[df\\_char\\_to\\_factor](#), 5  
[differential\\_abundance](#), 6  
[dimred\\_pca](#), 7  
[dimred\\_pcoa](#), 8  
[dimred\\_tsne](#), 9  
[dimred\\_umap](#), 10  
[diversities](#), 11  
[diversities\\_help](#), 12  
[diversity\\_beta\\_boxplot](#), 13  
[diversity\\_beta\\_heatmap](#), 14  
[diversity\\_beta\\_test](#), 15  
[do\\_alpha\\_div\\_test](#), 16

[filter\\_categorize](#), 17  
[filter\\_summary\\_bar\\_density](#), 18  
[filter\\_summary\\_pie\\_box](#), 19  
[find\\_biomarker](#), 20  
[find\\_taxon\\_mat](#), 22  
[find\\_taxonomy](#), 21  
[find\\_taxonomy\\_300](#), 21

[gini\\_simpson](#), 23  
[grep\\_tid](#), 23

[inverse\\_simpson](#), 24  
[is\\_categorical](#), 24  
[is\\_integer0](#), 25  
[is\\_integer1](#), 25

[mae\\_pick\\_organisms](#), 26  
[mae\\_pick\\_samples](#), 26

[pct2str](#), 27  
[percent](#), 28

[read\\_pathoscope\\_data](#), 28  
[relabu\\_barplot](#), 29  
[relabu\\_boxplot](#), 30  
[relabu\\_heatmap](#), 31  
[run\\_animalcules](#), 32

[shannon](#), 33  
[simpson\\_index](#), 33

[upsample\\_counts](#), 34

[write\\_to\\_biom](#), 35