Package ‘MiscMetabar’

June 21, 2025

Type Package

Title Miscellaneous Functions for Metabarcoding Analysis

Version 0.14.3

Description Facilitate the description, transformation, exploration, and reproducibility of metabarcoding analyses. 'MiscMetabar' is mainly built on top of the 'phyloseq', 'dada2' and 'targets' R packages. It helps to build reproducible and robust bioinformatics pipelines in R. 'MiscMetabar' makes ecological analysis of alpha and beta-diversity easier, more reproducible and more powerful by integrating a large number of tools. Important features are described in Taudière A. (2023) <doi:10.21105/joss.06038>.

License AGPL-3

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0), phyloseq, ggplot2 (>= 3.5.0), dada2, dplyr, purrr

Suggests adespatial, ANCOMBC, BiocManager, Biostrings, circlize, ComplexUpset, DECIPHER, DESeq2, devtools, DT, edgeR, formattable, ggalluvial, ggfittext, gghalves, ggh4x, ggstatsplot, ggridges, ggVennDiagram, glmulti, gtsummary, grDevices, grid, gridExtra, here, httr, iNEXT, indicspecies, IRanges, jsonlite, knitr, magrittr, methods, mia, mixtools, multcompView, networkD3, pak, patchwork, pbapply, permute, phangorn, phyloseqGraphTest, pkgnet, plotly, plyr, reshape2, rmarkdown, rotl, Rtsne, scales, seqinr, SRS, stringr, SummarizedExperiment, testthat (>= 3.0.0), tibble, tidyr, treemapify, umap, uwot, vegan, venneuler, vctrs, viridis, withr, spelling

RoxygenNote 7.3.2

URL `https://github.com/adrientaudiere/MiscMetabar`, `https://adrientaudiere.github.io/MiscMetabar/`

biocViews Sequencing, Microbiome, Metagenomics, Clustering, Classification, Visualization

BugReports `https://github.com/adrientaudiere/MiscMetabar/issues`

Imports ape, lifecycle, rlang, stats

Config/testthat/edition 3

Config/testthat/parallel true

VignetteBuilder knitr

Language en-US

NeedsCompilation no

Author Adrien Taudière [aut, cre, cph] (ORCID: <https://orcid.org/0000-0003-1088-1182>)

Maintainer Adrien Taudière `<adrien.taudiere@zaclys.net>`

Repository CRAN

Date/Publication 2025-06-21 14:00:02 UTC

Contents

`MiscMetabar-package` `MiscMetabar` *package*

Description

Functions to help analyze and visualize metabarcoding data. Mainly based on the phyloseq and dada2 packages.

`accu_plot` *Plot accumulation curves for* `phyloseq-class` *object*

Description

Note that as most bioinformatic pipeline discard singleton, accumulation curves from metabarcoding cannot be interpreted in the same way as with conventional biodiversity sampling techniques.

Usage

```
accu_plot(
  physeq,
  fact = NULL,
  add_nb_seq = TRUE,
  step = NULL,
  by.fact = FALSE,
  ci_col = NULL,
  col = NULL,
  lwd = 3,
  leg = TRUE,
  print_sam_names = FALSE,
  ci = 2,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required) Name of the factor in `physeq@sam_data` used to plot different lines
`add_nb_seq`	(default: TRUE, logical) Either plot accumulation curves using sequences or using samples
`step`	(Integer) distance among points calculated to plot lines. A low value give better plot but is more time consuming. Only used if `add_nb_seq` = TRUE.
`by.fact`	(default: FALSE, logical) First merge the OTU table by factor to plot only one line by factor
`ci_col`	Color vector for confidence interval. Only use if `add_nb_seq` = FALSE. If `add_nb_seq` = TRUE, you can use ggplot to modify the plot.
`col`	Color vector for lines. Only use if `add_nb_seq` = FALSE. If `add_nb_seq` = TRUE, you can use ggplot to modify the plot.
`lwd`	(default: 3) thickness for lines. Only use if `add_nb_seq` = FALSE.
`leg`	(default: TRUE, logical) Plot legend or not. Only use if `add_nb_seq` = FALSE.
`print_sam_names`	(default: FALSE, logical) Print samples names or not? Only use if `add_nb_seq` = TRUE.
`ci`	(default: 2, integer) Confidence interval value used to multiply the standard error to plot confidence interval
`...`	Additional arguments passed on to `ggplot` if `add_nb_seq` = TRUE or to `plot` if `add_nb_seq` = FALSE

Value

A `ggplot`2 plot representing the richness accumulation plot if add_nb_seq = TRUE, else, if add_nb_seq = FALSE return a base plot.

Author(s)

Adrien Taudière

See Also

specaccum accu_samp_threshold()

Examples

```
data("GlobalPatterns", package = "phyloseq")
GP <- subset_taxa(GlobalPatterns, GlobalPatterns@tax_table[, 1] == "Archaea")
GP <- rarefy_even_depth(subset_samples_pq(GP, sample_sums(GP) > 3000))
p <- accu_plot(GP, "SampleType", add_nb_seq = TRUE, by.fact = TRUE, step = 10)
p <- accu_plot(GP, "SampleType", add_nb_seq = TRUE, step = 10)

p + theme(legend.position = "none")

p + xlim(c(0, 400))
```

---

accu_plot_balanced_modality

Plot accumulation curves with balanced modality and depth rarefaction

Description

This function (i) rarefy (equalize) the number of samples per modality of a factor and (ii) rarefy the number of sequences per sample (depth). The seed is set to 1:nperm. Thus, with exacly the same parameter, including nperm values, results must be identical.

Usage

```
accu_plot_balanced_modality(
  physeq,
  fact,
  nperm = 99,
  step = 2000,
  by.fact = TRUE,
  progress_bar = TRUE,
  quantile_prob = 0.975,
  rarefy_by_sample_before_merging = TRUE,
  sample.size = 1000,
  verbose = FALSE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required) The variable to rarefy. Must be present in the `sam_data` slot of the physeq object.
`nperm`	(int) The number of permutations to perform.
`step`	(int) distance among points calculated to plot lines. A low value give better plot but is more time consuming.
`by.fact`	(logical, default TRUE) First merge the OTU table by factor to plot only one line by factor
`progress_bar`	(logical, default TRUE) Do we print progress during the calculation?
`quantile_prob`	(float, `[0:1]`) the value to compute the quantile. Minimum quantile is compute using 1-quantile_prob.
`rarefy_by_sample_before_merging`	(logical, default TRUE): rarefy_by_sample_before_merging = FALSE is buggy for the moment.Please only use rarefy_by_sample_before_merging = TRUE
`sample.size`	(int) A single integer value equal to the number of reads being simulated, also known as the depth. See `phyloseq::rarefy_even_depth()`.
`verbose`	(logical). If TRUE, print additional information.
`...`	Other params for be passed on to `accu_plot()` function

Value

A ggplot2 plot representing the richness accumulation plot

Author(s)

Adrien Taudière

See Also

`accu_plot()`, `rarefy_sample_count_by_modality()`, `phyloseq::rarefy_even_depth()`

Examples

```
data_fungi_woNA4Time <-
  subset_samples(data_fungi, !is.na(Time))
data_fungi_woNA4Time@sam_data$Time <- paste0("time-", data_fungi_woNA4Time@sam_data$Time)
accu_plot_balanced_modality(data_fungi_woNA4Time, "Time", nperm = 3)

data_fungi_woNA4Height <-
  subset_samples(data_fungi, !is.na(Height))
accu_plot_balanced_modality(data_fungi_woNA4Height, "Height", nperm = 3)
```

`accu_samp_threshold` *Compute the number of sequence to obtain a given proportion of ASV in accumulation curves*

Description

Note that as most bioinformatic pipeline discard singleton, accumulation curves from metabarcoding cannot be interpreted in the same way as with conventional biodiversity sampling techniques.

Usage

```
accu_samp_threshold(res_accuplot, threshold = 0.95)
```

Arguments

`res_accuplot`	the result of the function accu_plot()
`threshold`	the proportion of ASV to obtain in each samples

Value

a value for each sample of the number of sequences needed to obtain `threshold` proportion of the ASV

Author(s)

Adrien Taudière

See Also

`accu_plot()`

Examples

```
data("GlobalPatterns", package = "phyloseq")
GP <- subset_taxa(GlobalPatterns, GlobalPatterns@tax_table[, 1] == "Archaea")
GP <- rarefy_even_depth(subset_samples_pq(GP, sample_sums(GP) > 3000))
p <- accu_plot(GP, "SampleType", add_nb_seq = TRUE, by.fact = TRUE, step = 10)

val_threshold <- accu_samp_threshold(p)

summary(val_threshold)

##'  Plot the number of sequences needed to accumulate 0.95% of ASV in 50%, 75%
##'  and 100% of samples
p + geom_vline(xintercept = quantile(val_threshold, probs = c(0.50, 0.75, 1)))
```

`add_blast_info` *Add information from* `blast_pq()` *to the* `tax_table` *slot of a* phyloseq *object*

Description

Basically a wrapper of `blast_pq()` with option `unique_per_seq = TRUE` and `score_filter = FALSE`.

Add the information to the taxtable

Usage

```
add_blast_info(
  physeq,
  fasta_for_db,
  silent = FALSE,
  suffix = "blast_info",
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`fasta_for_db`	path to a fasta file to make the blast database
`silent`	(logical) If true, no message are printing.
`suffix`	(character) The suffix to name the new columns. Set the suffix to "" in order to remove any suffix.
`...`	Additional arguments passed on to `blast_pq()` function.

Value

A new `phyloseq-class` object with more information in tax_table based on a blast on a given database

Author(s)

Adrien Taudière

`add_dna_to_phyloseq` *Add dna in* `refseq` *slot of a* `physeq` *object using taxa names and renames taxa using prefix_taxa_names and number (default Taxa_1, Taxa_2 ...)*

Description

Useful in targets bioinformatic pipeline.

Usage

```
add_dna_to_phyloseq(physeq, prefix_taxa_names = "Taxa_")
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`prefix_taxa_names`	(default "Taxa_"): the prefix of taxa names (eg. "ASV_" or "OTU_")

Value

A new `phyloseq-class` object with `refseq` slot and new taxa names

Author(s)

Adrien Taudière

`add_funguild_info` *Add information about Guild for FUNGI the FUNGuild databse*

Description

Please cite Nguyen et al. 2016 (doi:10.1016/j.funeco.2015.06.006)

Usage

```
add_funguild_info(
  physeq,
 taxLevels = c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species"),
  db_url = "http://www.stbates.org/funguild_db_2.php"
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`taxLevels`	Name of the 7 columns in tax_table required by funguild
`db_url`	a length 1 character string giving the URL to retrieve the database from

Details

This function is mainly a wrapper of the work of others. Please make a reference to `FUNGuildR` package and the associate publication (doi:10.1016/j.funeco.2015.06.006) if you use this function.

Value

A new object of class `physeq` with Guild information added to `tax_table` slot

Author(s)

Adrien Taudière

See Also

`plot_guild_pq()`

Examples

```
## Not run:
# to avoid bug in CRAN when internet is not available
if (requireNamespace("httr")) {
  d_fung_mini <- add_funguild_info(data_fungi_mini,
    taxLevels = c(
      "Domain",
      "Phylum",
      "Class",
      "Order",
      "Family",
      "Genus",
      "Species"
    )
  )
  sort(table(d_fung_mini@tax_table[, "guild"]), decreasing = TRUE)
}

## End(Not run)
```

`add_info_to_sam_data` *Add information to sample_data slot of a phyloseq-class object*

Description

Warning: The value nb_seq and nb_otu may be outdated if you transform your phyloseq object, e.g. using the `subset_taxa_pq()` function

Usage

```
add_info_to_sam_data(
  physeq,
  df_info = NULL,
  add_nb_seq = TRUE,
  add_nb_otu = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`df_info`	: A dataframe with rownames matching for sample names of the phyloseq object
`add_nb_seq`	(Logical, default TRUE) Does we add a column nb_seq collecting the number of sequences per sample?
`add_nb_otu`	(Logical, default TRUE) Does we add a column nb_otu collecting the number of OTUs per sample?

Value

A phyloseq object with an updated sam_data slot

Author(s)

Adrien Taudière

Examples

```
data_fungi <- add_info_to_sam_data(data_fungi)
boxplot(data_fungi@sam_data$nb_otu ~ data_fungi@sam_data$Time)

new_df <- data.frame(
  variable_1 = runif(n = nsamples(data_fungi), min = 1, max = 20),
  variable_2 = runif(n = nsamples(data_fungi), min = 1, max = 2)
)
rownames(new_df) <- sample_names(data_fungi)
data_fungi <- add_info_to_sam_data(data_fungi, new_df)
plot(data_fungi@sam_data$nb_otu ~ data_fungi@sam_data$variable_1)
```

`add_new_taxonomy_pq` *Add new taxonomic rank to a phyloseq object.*

Description

One of main use of this function is to add taxonomic assignment from a new database.

Usage

```
add_new_taxonomy_pq(
  physeq,
  ref_fasta,
  suffix = NULL,
  method = c("dada2", "sintax", "lca", "idtaxa", "blastn", "dada2_2steps"),
  trainingSet = NULL,
  min_bootstrap = NULL,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`ref_fasta`	(required) A link to a database. passed on to `dada2::assignTaxonomy`.
`suffix`	(character) The suffix to name the new columns. If set to NULL (the default), the basename of the file reFasta is used with the name of the method. Set suffix to "" in order to remove any suffix.
`method`	(required, default "dada2") : • "dada2": `dada2::assignTaxonomy()` • "dada2_2step": `assign_dada2()` • "sintax": see `assign_sintax()` • "lca": see `assign_vsearch_lca()` • "idtaxa": see `assign_idtaxa()` • "blastn": see `assign_blastn()`
`trainingSet`	see `assign_idtaxa()`. Only used if method = "idtaxa". Note that if trainingSet is not NULL, the ref_fasta is overwrite by the trainingSet parameter. To customize learning parameters of the idtaxa algorithm you must use trainingSet computed by the function `learn_idtaxa()`.
`min_bootstrap`	(Float [0:1]) Minimum bootstrap value to inform taxonomy. For each bootstrap below the min_bootstrap value, the taxonomy information is set to NA. Correspond to parameters : • dada2 & dada2_2step: `minBoot`, default value = 0.5 • sintax: `min_bootstrap`, default value = 0.5 • lca: `id`, default value = 0.5. Note in that case, the bootstrap value is different. See the id parameter in `assign_vsearch_lca()` • idtaxa: `threshold`, default value = 0.6 • blastn: This method do not take different bootstrap value. You may use method="vote" with different `vote_algorithm` as well as different filters parameters (min_id, min_bit_score, min_cover and min_e_value)
`...`	Additional arguments passed on to the taxonomic assignation method.

Value

A new `phyloseq-class` object with a larger slot tax_table"

Author(s)

Adrien Taudière

See Also

`dada2::assignTaxonomy()`, `assign_sintax()`, `assign_vsearch_lca()`, `assign_sintax()`, `assign_blastn()`, `assign_dada2()`

Examples

```
## Not run:
ref_fasta <- system.file("extdata",
  "mini_UNITE_fungi.fasta.gz",
  package = "MiscMetabar", mustWork = TRUE
)
add_new_taxonomy_pq(data_fungi_mini, ref_fasta, method = "dada2")
add_new_taxonomy_pq(data_fungi_mini, ref_fasta, method = "dada2_2steps")
add_new_taxonomy_pq(data_fungi_mini, ref_fasta, method = "lca")
add_new_taxonomy_pq(data_fungi_mini, ref_fasta, method = "idtaxa")

# blastn doesn't work with fasta.gz format
ref_fasta <- system.file("extdata",
  "100_sp_UNITE_sh_general_release_dynamic_sintax.fasta",
  package = "MiscMetabar", mustWork = TRUE
)

dp <- add_new_taxonomy_pq(data_fungi_mini, ref_fasta,
  method = "blastn", min_id = 80, min_cover = 50, min_bit_score = 20,
  min_e_value = 1e-20
)
dp_tophit <- add_new_taxonomy_pq(data_fungi_mini, ref_fasta,
  method = "blastn", min_id = 80, min_cover = 50, min_bit_score = 20,
  min_e_value = 1e-20, method_algo = "top_hit"
)

## End(Not run)
```

`adonis_pq` *Permanova on a phyloseq object*

Description

A wrapper for the `vegan::adonis2()` function in the case of physeq object.

Usage

```
adonis_pq(
  physeq,
  formula,
  dist_method = "bray",
  merge_sample_by = NULL,
  na_remove = FALSE,
  correction_for_sample_size = FALSE,
  rarefy_nb_seqs = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`formula`	(required) the right part of a formula for `vegan::adonis2()`. Variables must be present in the `physeq@sam_data` slot.
`dist_method`	(default "bray") the distance used. See `phyloseq::distance()` for all available distances or run `phyloseq::distanceMethodList()`. For aitchison and robust.aitchison distance, `vegan::vegdist()` function is directly used.
`merge_sample_by`	a vector to determine which samples to merge using the `merge_samples2()` function. Need to be in `physeq@sam_data`
`na_remove`	(logical, default FALSE) If set to TRUE, remove samples with NA in the variables set in formula.
`correction_for_sample_size`	(logical, default FALSE) If set to TRUE, the sample size (number of sequences by samples) is added to formula in the form `y~Library_Size + Biological_Effect` following recommendation of Weiss et al. 2017. `correction_for_sample_size` overcome `rarefy_nb_seqs` if both are TRUE.
`rarefy_nb_seqs`	(logical, default FALSE) Rarefy each sample (before merging if merge_sample_by is set) using `phyloseq::rarefy_even_depth()`. if `correction_for_sample_size` is TRUE, rarefy_nb_seqs will have no effect.
`verbose`	(logical, default TRUE) If TRUE, prompt some messages.
`...`	Additional arguments passed on to `vegan::adonis2()` function. Note that the parameter by is important. If by is set to NULL (default) the p-value is computed for the entire model. by = NULL will assess the overall significance of all terms together, by = "terms" will assess significance for each term (sequentially from first to last), setting by = "margin" will assess the marginal effects of the terms (each marginal term analyzed in a model with all other variables), by = "onedf" will analyze one-degree-of-freedom contrasts sequentially. The argument is passed on to anova.cca.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `vegan::adonis2()` if you use this function.

Value

The function returns an anova.cca result object with a new column for partial R^2. See help of `vegan::adonis2()` for more information.

Author(s)

Adrien Taudière

Examples

```
data(enterotype)

adonis_pq(enterotype, "SeqTech*Enterotype", na_remove = TRUE)
adonis_pq(enterotype, "SeqTech*Enterotype", na_remove = TRUE, by = "terms")
adonis_pq(enterotype, "SeqTech*Enterotype", na_remove = TRUE, by = "onedf")
adonis_pq(enterotype, "SeqTech*Enterotype", na_remove = TRUE, by = "margin")

adonis_pq(enterotype, "SeqTech", dist_method = "jaccard", by = "terms")
adonis_pq(enterotype, "SeqTech", dist_method = "robust.aitchison", by = "terms")
```

`adonis_rarperm_pq` *Permanova (adonis) on permutations of rarefaction even depth*

Description

Permanova are computed on a given number of rarefaction with different seed.number. This reduce the risk of a random drawing of a exceptional situation of an unique rarefaction.

Usage

```
adonis_rarperm_pq(
  physeq,
  formula,
  dist_method = "bray",
  merge_sample_by = NULL,
  na_remove = FALSE,
  rarefy_nb_seqs = FALSE,
  verbose = TRUE,
  nperm = 99,
  progress_bar = TRUE,
  quantile_prob = 0.975,
  sample.size = min(sample_sums(physeq)),
```

```
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`formula`	(required) the right part of a formula for `vegan::adonis2()`. Variables must be present in the `physeq@sam_data` slot.
`dist_method`	(default "bray") the distance used. See `phyloseq::distance()` for all available distances or run `phyloseq::distanceMethodList()`. For aitchison and robust.aitchison distance, `vegan::vegdist()` function is directly used.
`merge_sample_by`	a vector to determine which samples to merge using the `merge_samples2()` function. Need to be in `physeq@sam_data`
`na_remove`	(logical, default FALSE) If set to TRUE, remove samples with NA in the variables set in formula.
`rarefy_nb_seqs`	(logical, default FALSE) Rarefy each sample (before merging if merge_sample_by is set) using `phyloseq::rarefy_even_depth()`. if `correction_for_sample_size` is TRUE, rarefy_nb_seqs will have no effect.
`verbose`	(logical, default TRUE) If TRUE, prompt some messages.
`nperm`	(int, default = 99) The number of permutations to perform.
`progress_bar`	(logical, default TRUE) Do we print progress during the calculation.
`quantile_prob`	(float, `[0:1]`) the value to compute the quantile. Minimum quantile is computed using 1-quantile_prob.
`sample.size`	(int) A single integer value equal to the number of reads being simulated, also known as the depth. See `phyloseq::rarefy_even_depth()`.
`...`	Other params to be passed on to `adonis_pq()` function

Value

A list of three dataframe representing the mean, the minimum quantile and the maximum quantile value for adonis results. See `adonis_pq()`.

Author(s)

Adrien Taudière

See Also

`adonis_pq()`

Examples

```
if (requireNamespace("vegan")) {
  data_fungi_woNA <-
    subset_samples(data_fungi, !is.na(Time) & !is.na(Height))
  adonis_rarperm_pq(data_fungi_woNA, "Time*Height", na_remove = TRUE, nperm = 3)
}
```

`all_object_size` *List the size of all objects of the GlobalEnv.*

Description

Code from https://tolstoy.newcastle.edu.au/R/e6/help/09/01/1121.html

Usage

```
all_object_size()
```

Value

a list of size

`ancombc_pq` *Run ANCOMBC2 on phyloseq object*

Description

A wrapper for the `ANCOMBC::ancombc2()` function

Usage

```
ancombc_pq(physeq, fact, levels_fact = NULL, tax_level = "Class", ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required) Name of the factor in `physeq@sam_data` used to plot different lines
`levels_fact`	(default NULL) The order of the level in the factor. Used for reorder levels and select levels (filter out levels not present en levels_fact)
`tax_level`	The taxonomic level passed on to `ANCOMBC::ancombc2()`
`...`	Additional arguments passed on to `ANCOMBC::ancombc2()` function.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `ANCOMBC::ancombc2()` if you use this function.

Value

The result of `ANCOMBC::ancombc2()` function

Author(s)

Adrien Taudière

Examples

```
## Not run:
if (requireNamespace("mia")) {
  data_fungi_mini@tax_table <- phyloseq::tax_table(cbind(
    data_fungi_mini@tax_table,
    "taxon" = taxa_names(data_fungi_mini)
  ))
  res_height <- ancombc_pq(
    data_fungi_mini,
    fact = "Height",
    levels_fact = c("Low", "High"),
    verbose = TRUE
  )

  ggplot(
    res_height$res,
    aes(
      y = reorder(taxon, lfc_HeightHigh),
      x = lfc_HeightHigh,
      color = diff_HeightHigh
    )
  ) +
    geom_vline(xintercept = 0) +
    geom_segment(aes(
      xend = 0, y = reorder(taxon, lfc_HeightHigh),
      yend = reorder(taxon, lfc_HeightHigh)
    ), color = "darkgrey") +
    geom_point()

  res_time <- ancombc_pq(
    data_fungi_mini,
    fact = "Time",
    levels_fact = c("0", "15"),
    tax_level = "Family",
    verbose = TRUE
  )
}

## End(Not run)
```

`are_modality_even_depth` *Test if the mean number of sequences by samples is link to the modality of a factor*

Description

The aim of this function is to provide a warnings if samples depth significantly vary among the modalities of a factor present in the `sam_data` slot.

This function apply a Kruskal-Wallis rank sum test to the number of sequences per samples in function of the factor `fact`.

Usage

```
are_modality_even_depth(physeq, fact, boxplot = FALSE)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`fact`	(required): Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`.
`boxplot`	(logical) Do you want to plot boxplot?

Value

The result of a Kruskal-Wallis rank sum test

Author(s)

Adrien Taudière

Examples

```
are_modality_even_depth(data_fungi_mini, "Time")$p.value
are_modality_even_depth(rarefy_even_depth(data_fungi_mini), "Time")$p.value
are_modality_even_depth(data_fungi_mini, "Height", boxplot = TRUE)
```

`assign_blastn` *Assign taxonomy using blastn algorithm and the blast software*

Description

Use the blast software.

Usage

```
assign_blastn(
  physeq,
  ref_fasta = NULL,
  database = NULL,
  blastpath = NULL,
  behavior = c("return_matrix", "add_to_phyloseq"),
  method_algo = c("vote", "top-hit"),
```

```
  suffix = "_blastn",
  min_id = 95,
  min_bit_score = 50,
  min_cover = 95,
  min_e_value = 1e-30,
  nb_voting = NULL,
 column_names = c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species"),
  vote_algorithm = c("consensus", "rel_majority", "abs_majority", "unanimity"),
  strict = FALSE,
  nb_agree_threshold = 1,
  preference_index = NULL,
  collapse_string = "/",
  replace_collapsed_rank_by_NA = TRUE,
  simplify_taxo = TRUE,
  keep_blast_metrics = FALSE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`ref_fasta`	Either a DNAStringSet object or a path to a fasta file to make the blast database. It must be in sintax format. See `assign_sintax()`.
`database`	path to a blast database. Only used if ref_fasta is not set.
`blastpath`	path to blast program.
`behavior`	Either "return_matrix" (default), or "add_to_phyloseq": • "return_matrix" return a list of two matrix with taxonomic value in the first element of the list and bootstrap value in the second one. • "add_to_phyloseq" return a phyloseq object with amended slot `@taxtable`. Only available if using physeq input and not seq2search input.
`method_algo`	(One of "vote" or "top-hit"). If top-hit, only the better match is used to assign taxonomy. If vote, the algorithm takes all (or `nb_voting` if `nb_voting` is not null) select assignation and resolve the conflict using the function `resolve_vector_ranks()`.
`suffix`	(character) The suffix to name the new columns. If set to "" (the default), the taxa_ranks algorithm is used without suffix.
`min_id`	(default: 95) the identity percent to take into account a references taxa
`min_bit_score`	(default: 50) the minimum bit score to take into account a references taxa
`min_cover`	(default: 50) cut of in query cover (%) to keep result
`min_e_value`	(default: 1e-30) cut of in e-value (%) to keep result The BLAST E-value is the number of expected hits of similar quality (score) that could be found just by chance.
`nb_voting`	(Int, default NULL). The number of taxa to keep before apply a vote to resolve conflict. If NULL all taxa passing the filters (min_id, min_bit_score, min_cover and min_e_value) are selected.

`column_names`	A vector of names for taxonomic ranks. Must correspond to names in the ref_fasta files.
`vote_algorithm`	the method to vote among "consensus", "rel_majority", "abs_majority" and "unanimity". See `resolve_vector_ranks()` for more details.
`strict`	(Logical, default FALSE). See `resolve_vector_ranks()` for more details.
`nb_agree_threshold`	See `resolve_vector_ranks()` for more details.
`preference_index`	See `resolve_vector_ranks()` for more details.
`collapse_string`	See `resolve_vector_ranks()` for more details.
`replace_collapsed_rank_by_NA`	(Logical, default TRUE) See `resolve_vector_ranks()` for more details.
`simplify_taxo`	(logical default TRUE). Do we apply the function `simplify_taxo()` to the phyloseq object?
`keep_blast_metrics`	(Logical, default FALSE). If TRUE, the blast metrics ("Query seq. length", "Taxa seq. length", "Alignment length", "% id. match", "e-value", "bit score" and "Query cover") are stored in the tax_table.
`...`	Additional arguments passed on to `blast_pq()`

Value

- If behavior == "return_matrix" :
  - If method_algo = "top-hit" a matrix of taxonomic assignation
  - If method_algo = "vote", a list of two matrix, the first is the raw taxonomic assignation (before vote). The second one is the taxonomic assignation in which conflicts are resolved using vote.
- If behavior == "add_to_phyloseq", return a new phyloseq object

Author(s)

Adrien Taudière

Examples

```
## Not run:
ref_fasta <- Biostrings::readDNAStringSet(system.file("extdata",
  "mini_UNITE_fungi.fasta.gz",
  package = "MiscMetabar", mustWork = TRUE
))

# assign_blastn(data_fungi_mini, ref_fasta = ref_fasta) # error because not
# enough sequences in db so none blast query passed the filters.
# So we used low score filter hereafter.

mat <- assign_blastn(data_fungi_mini,
```

```
  ref_fasta = ref_fasta,
  method_algo = "top-hit", min_id = 70, min_e_value = 1e-3, min_cover = 50,
  min_bit_score = 20
)
head(mat)

assign_blastn(data_fungi_mini,
  ref_fasta = ref_fasta, method_algo = "vote",
  vote_algorithm = "rel_majority", min_id = 90, min_cover = 50,
  behavior = "add_to_phyloseq"
)@tax_table

assign_blastn(data_fungi_mini,
  ref_fasta = ref_fasta, method_algo = "vote",
  vote_algorithm = "consensus", replace_collapsed_rank_by_NA = FALSE,
  min_id = 90, min_cover = 50, behavior = "add_to_phyloseq"
)@tax_table

## End(Not run)
```

`assign_dada2` *Assign taxonomy with dada2 using 2 steps assignTaxonomy and assignSpecies*

Description

Mainly a wrapper of `dada2::assignTaxonomy()` and `dada2::assignSpecies()`

Usage

```
assign_dada2(
  physeq = NULL,
  ref_fasta = NULL,
  seq2search = NULL,
  min_bootstrap = 0.5,
  tryRC = FALSE,
 taxa_ranks = c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species",
    "taxId"),
  use_assignSpecies = TRUE,
  trunc_absent_ranks = FALSE,
  nproc = 1,
  suffix = "",
  verbose = TRUE,
  seq_at_one_time = 2000,
  allowMultiple = FALSE,
  from_sintax = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`ref_fasta`	(required) A link to a database in fasta.
`seq2search`	A DNAStringSet object of sequences to search for. Replace the physeq object.
`min_bootstrap`	(Float [0:1], default 0.5), See `dada2::assignTaxonomy()`
`tryRC`	See `dada2::assignTaxonomy()`
`taxa_ranks`	(vector of character) names for the column of the taxonomy
`use_assignSpecies`	(logical, default TRUE) Do the Species rank is obtained using `dada2::assignSpecies()` ?
`trunc_absent_ranks`	(logical, default FALSE) Do ranks present in taxa_ranks but not present in the database are removed ?
`nproc`	(Float [0:1], default 0.5)
`suffix`	(character) The suffix to name the new columns. Default to "_idtaxa".
`verbose`	(logical). If TRUE, print additional information.
`seq_at_one_time`	How many sequences are treated at one time. See param `n` in `dada2::assignSpecies()`
`allowMultiple`	(logical, default FALSE). Unchanged from `dada2::assignSpecies()`. Defines the behavior when multiple exact matches against different species are returned. By default only unambiguous identifications are return. If TRUE, a concatenated string of all exactly matched species is returned. If an integer is provided, multiple identifications up to that many are returned as a concatenated string.
`from_sintax`	(logical, default FALSE). Set to TRUE if the ref_fasta database is in sintax format. See `assign_sintax()` for more information about the sintax format.

Value

Either a an object of class phyloseq (if `physeq` is not NULL), or a taxonomic table if `seq2search` is used in place of `physeq`

Examples

```
## Not run:
data_fungi_mini2 <- assign_dada2(data_fungi_mini,
  ref_fasta = system.file("extdata", "mini_UNITE_fungi.fasta.gz",
    package = "MiscMetabar"
  ), suffix = "_dada2",
  from_sintax = TRUE
)

## End(Not run)
```

`assign_idtaxa` *A wrapper of* `IdTaxa`

Description

This function is basically a wrapper of functions `DECIPHER::IdTaxa()` and `DECIPHER::LearnTaxa()`, please cite the DECIPHER package if you use this function. Note that if you want to specify parameters for the learning step you must used the trainingSet param instead of the a fasta_for_training. The training file can be obtain using the function `learn_idtaxa()`.

It requires:

- either a physeq or seq2search object.
- either a trainingSet or a fasta_for_training

Usage

```
assign_idtaxa(
  physeq,
  trainingSet = NULL,
  seq2search = NULL,
  fasta_for_training = NULL,
  behavior = "return_matrix",
  threshold = 60,
 column_names = c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species"),
  suffix = "_idtaxa",
  nproc = 1,
  unite = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

`physeq` (required): a `phyloseq-class` object obtained using the phyloseq package.

`trainingSet` An object of class Taxa and subclass Train compatible with the class of test.

`seq2search` A DNAStringSet object of sequences to search for. Replace the physeq object.

`fasta_for_training`
A fasta file (can be gzip) to train the trainingSet using the function `learn_idtaxa()`. Only used if trainingSet is NULL.

The reference database must contain taxonomic information in the header of each sequence in the form of a string starting with ";tax=" and followed by a comma-separated list of up to nine taxonomic identifiers.

The only exception is if `unite=TRUE`. In that case the UNITE taxonomy is automatically formatted.

`behavior` Either "return_matrix" (default), or "add_to_phyloseq":

- "return_matrix" return a list of two objects. The first element is the taxonomic matrix and the second element is the raw results from DECIPHER::IdTaxa() function.
- "add_to_phyloseq" return a phyloseq object with amended slot `@taxtable`. Only available if using physeq input and not seq2search input.

`threshold`	(Int, default 60) Numeric specifying the confidence at which to truncate the output taxonomic classifications. Lower values of threshold will classify deeper into the taxonomic tree at the expense of accuracy, and vise-versa for higher values of threshold. See `DECIPHER::IdTaxa()` man page.
`column_names`	(vector of character) names for the column of the taxonomy
`suffix`	(character) The suffix to name the new columns. Default to "_idtaxa".
`nproc`	(default: 1) Set to number of cpus/processors to use
`unite`	(logical, default FALSE). If set to TRUE, the fasta_for_training file is formatted from UNITE format to sintax one, needed in fasta_for_training. Only used if trainingSet is NULL.
`verbose`	(logical). If TRUE, print additional information.
`...`	Additional arguments passed on to `IdTaxa`

Details

This function is mainly a wrapper of the work of others. Please make a reference to `DECIPHER::IdTaxa()` if you use this function.

Value

Either a new phyloseq object with additional information in the @tax_table slot or a list of two objects if behavior is "return_matrix"

Author(s)

Adrien Taudière

See Also

`assign_sintax()`, `add_new_taxonomy_pq()`, `assign_vsearch_lca()`, `assign_blastn()`

Examples

```
## Not run:
# /!\ The value of threshold must be change for real database (recommend
#  value are between 50 and 70).

data_fungi_mini_new <- assign_idtaxa(data_fungi_mini,
  fasta_for_training = system.file("extdata", "mini_UNITE_fungi.fasta.gz",
    package = "MiscMetabar"
  ), threshold = 20, behavior = "add_to_phyloseq"
)
```

```
result_idtaxa <- assign_idtaxa(data_fungi_mini,
  fasta_for_training = system.file("extdata", "mini_UNITE_fungi.fasta.gz",
    package = "MiscMetabar"
  ), threshold = 20
)

plot(result_idtaxa$idtaxa_raw)

## End(Not run)
```

`assign_sintax` *Assign Taxonomy using Sintax algorithm of Vsearch*

Description

Please cite Vsearch if you use this function to assign taxonomy.

Usage

```
assign_sintax(
  physeq = NULL,
  ref_fasta = NULL,
  seq2search = NULL,
  behavior = c("return_matrix", "add_to_phyloseq", "return_cmd"),
  vsearchpath = "vsearch",
  clean_pq = TRUE,
  nproc = 1,
  suffix = "",
 taxa_ranks = c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species"),
  min_bootstrap = 0.5,
  keep_temporary_files = FALSE,
  verbose = FALSE,
  temporary_fasta_file = "temp.fasta",
  cmd_args = "--sintax_random",
  too_few = "align_start",
  too_many = "drop"
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`ref_fasta`	(required) A link to a database in vsearch format The reference database must contain taxonomic information in the header of each sequence in the form of a string starting with ";tax=" and followed by a comma-separated list of up to nine taxonomic identifiers. Each taxonomic identifier must start with an indication of the rank by one of the letters d (for domain) k (kingdom), p (phylum), c (class), o (order), f (family), g (genus), s (species), or t (strain). The letter is followed by

	a colon (:) and the name of that rank. Commas and semicolons are not allowed in the name of the rank. Non-ascii characters should be avoided in the names. Example: \>X80725_S000004313;tax=d:Bacteria,p:Proteobacteria,c:Gammaproteobacteria,o:Enterobacteriales,f:E 12_substr._MG1655
`seq2search`	A DNAStringSet object of sequences to search for. Replace the physeq object.
`behavior`	Either "return_matrix" (default), "return_cmd", or "add_to_phyloseq": • "return_matrix" return a list of two matrix with taxonomic value in the first element of the list and bootstrap value in the second one. • "return_cmd" return the command to run without running it. • "add_to_phyloseq" return a phyloseq object with amended slot `@taxtable`. Only available if using physeq input and not seq2search input.
`vsearchpath`	(default: "vsearch") path to vsearch
`clean_pq`	(logical, default TRUE) If set to TRUE, empty samples and empty ASV are discarded before clustering.
`nproc`	(default: 1) Set to number of cpus/processors to use
`suffix`	(character) The suffix to name the new columns. If set to "" (the default), the taxa_ranks algorithm is used without suffix.
`taxa_ranks`	A list with the name of the taxonomic rank present in ref_fasta
`min_bootstrap`	(Float [0:1], default 0.5) Minimum bootstrap value to inform taxonomy. For each bootstrap below the min_bootstrap value, the taxonomy information is set to NA.
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files? • temporary_fasta_file (default "temp.fasta") : the fasta file from physeq or seq2search • "output_taxo_vs.txt" : see Vsearch Manual for parameter –tabbedout
`verbose`	(logical). If TRUE, print additional information.
`temporary_fasta_file`	The name of a temporary_fasta_file (default "temp.fasta")
`cmd_args`	Additional arguments passed on to vsearch sintax cmd. By default cmd_args is equal to "–sintax_random" as recommended by Torognes.
`too_few`	(default value "align_start") see `tidyr::separate_wider_delim()`
`too_many`	(default value "drop") see `tidyr::separate_wider_delim()`

Details

This function is mainly a wrapper of the work of others. Please cite vsearch.

Value

See param behavior

Author(s)

Adrien Taudière

Examples

```
assign_sintax(data_fungi_mini,
  ref_fasta = system.file("extdata", "mini_UNITE_fungi.fasta.gz", package = "MiscMetabar"),
  behavior = "return_cmd"
)

data_fungi_mini_new <- assign_sintax(data_fungi_mini,
  ref_fasta = system.file("extdata", "mini_UNITE_fungi.fasta.gz", package = "MiscMetabar"),
  behavior = "add_to_phyloseq"
)

assignation_results <- assign_sintax(data_fungi_mini,
  ref_fasta = system.file("extdata", "mini_UNITE_fungi.fasta.gz", package = "MiscMetabar")
)

left_join(
  tidyr::pivot_longer(assignation_results$taxo_value, -taxa_names),
  tidyr::pivot_longer(assignation_results$taxo_bootstrap, -taxa_names),
  by = join_by(taxa_names, name),
  suffix = c("rank", "bootstrap")
) |>
  mutate(name = factor(name,
    levels = c(
      "Kingdom", "Phylum", "Class",
      "Order", "Family", "Genus", "Species"
    )
  )) |>
  # mutate(valuerank = forcats::fct_reorder(valuerank,
  #   as.integer(name), .desc = TRUE)) |>
  ggplot(aes(valuebootstrap,
    valuerank,
    fill = name
  )) +
  geom_jitter(alpha = 0.8, aes(color = name)) +
  geom_boxplot(alpha = 0.3)
```

`assign_vsearch_lca` *Assign taxonomy using LCA*

Description

Please cite Vsearch and stampa if you use this function to assign taxonomy.

1. If top_hits_only is TRUE, the algorithm is the one of stampa.
2. If top_hits_only is FALSE and vote_algorithm is NULL, you need to carefully define `maxaccept`, `id` and `lca_cutoff` parameters. The algorithm is internal to vsearch using the lcaout output.
3. If top_hits_only is FALSE and vote_algorithm is not NULL, conflict among the list of taxonomic assignations is resolve using the function `resolve_vector_ranks()`. The possible values for vote_algorithm are "consensus", "rel_majority", "abs_majority" and "unanimity". See `resolve_vector_ranks()` for more details.

Usage

```
assign_vsearch_lca(
  physeq = NULL,
  ref_fasta = NULL,
  seq2search = NULL,
  behavior = c("return_matrix", "add_to_phyloseq", "return_cmd"),
  vsearchpath = "vsearch",
  clean_pq = TRUE,
 taxa_ranks = c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species"),
  nproc = 1,
  suffix = "_sintax",
  id = 0.5,
  lca_cutoff = 1,
  maxrejects = 32,
  top_hits_only = TRUE,
  maxaccepts = 0,
  keep_temporary_files = FALSE,
  verbose = TRUE,
  temporary_fasta_file = "temp.fasta",
  cmd_args = "",
  too_few = "align_start",
  vote_algorithm = NULL,
  nb_voting = NULL,
  strict = FALSE,
  nb_agree_threshold = 1,
  preference_index = NULL,
  collapse_string = "/",
  replace_collapsed_rank_by_NA = TRUE,
  simplify_taxo = TRUE,
  keep_vsearch_score = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`ref_fasta`	(required) A link to a database in vsearch format The reference database must contain taxonomic information in the header of each sequence in the form of a string starting with ";tax=" and followed by a comma-separated list of up to nine taxonomic identifiers. Each taxonomic identifier must start with an indication of

	the rank by one of the letters d (for domain) k (kingdom), p (phylum), c (class), o (order), f (family), g (genus), s (species), or t (strain). The letter is followed by a colon (:) and the name of that rank. Commas and semicolons are not allowed in the name of the rank. Non-ascii characters should be avoided in the names. Example: \>X80725_S000004313;tax=d:Bacteria,p:Proteobacteria,c:Gammaproteobacteria,o:Enterobacteriales,f:E 12_substr._MG1655
`seq2search`	A DNAStringSet object of sequences to search for. Replace the physeq object.
`behavior`	Either "return_matrix" (default), "return_cmd", or "add_to_phyloseq": • "return_matrix" return a list of two matrix with taxonomic value in the first element of the list and bootstrap value in the second one. • "return_cmd" return the command to run without running it. • "add_to_phyloseq" return a phyloseq object with amended slot `@taxtable`. Only available if using physeq input and not seq2search input.
`vsearchpath`	(default: "vsearch") path to vsearch
`clean_pq`	(logical, default TRUE) If set to TRUE, empty samples and empty ASV are discarded before clustering.
`taxa_ranks`	A list with the name of the taxonomic rank present in ref_fasta
`nproc`	(int, default: 1) Set to number of cpus/processors to use
`suffix`	(character) The suffix to name the new columns. If set to "" (the default), the taxa_ranks algorithm is used without suffix.
`id`	(Float [0:1] default 0.5). Default value is based on stampa. See Vsearch Manual for parameter `--id`
`lca_cutoff`	(int, default 1). Fraction of matching hits required for the last common ancestor (LCA) output. For example, a value of 0.9 imply that if less than 10% of assigned species are not congruent the taxonomy is filled. Default value is based on stampa. See Vsearch Manual for parameter `--lca_cutoff` Text from vsearch manual : "Adjust the fraction of matching hits required for the last common ancestor (LCA) output with the –lcaout option during searches. The default value is 1.0 which requires all hits to match at each taxonomic rank for that rank to be included. If a lower cutoff value is used, e.g. 0.95, a small fraction of non-matching hits are allowed while that rank will still be reported. The argument to this option must be larger than 0.5, but not larger than 1.0"
`maxrejects`	(int, default: 32) Maximum number of non-matching target sequences to consider before stopping the search for a given query. Default value is based on stampa See Vsearch Manual for parameter `--maxrejects`.
`top_hits_only`	(Logical, default TRUE) Only the top hits with an equally high percentage of identity between the query and database sequence sets are written to the output. If you set top_hits_only you may need to set a lower `maxaccepts` and/or `lca_cutoff`. Default value is based on stampa See Vsearch Manual for parameter `--top_hits_only`
`maxaccepts`	(int, default: 0) Default value is based on stampa. Maximum number of matching target sequences to accept before stopping the search for a given query. See Vsearch Manual for parameter `--maxaccepts`

`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files? • temporary_fasta_file (default "temp.fasta") : the fasta file from physeq or seq2search • "out_lca.txt" : see Vsearch Manual for parameter –lcaout • "userout.txt" : see Vsearch Manual for parameter –userout
`verbose`	(logical). If TRUE, print additional information.
`temporary_fasta_file`	Name of the temporary fasta file. Only useful with keep_temporary_files = TRUE.
`cmd_args`	Additional arguments passed on to vsearch usearch_global cmd.
`too_few`	(default value "align_start") see `tidyr::separate_wider_delim()`
`vote_algorithm`	(default NULL) the method to vote among "consensus", "rel_majority", "abs_majority" and "unanimity". See `resolve_vector_ranks()` for more details.
`nb_voting`	(Int, default NULL). The number of taxa to keep before apply a vote to resolve conflict. If NULL all taxa passing the filters (min_id, min_bit_score, min_cover and min_e_value) are selected.
`strict`	(Logical, default FALSE). See `resolve_vector_ranks()` for more details.
`nb_agree_threshold`	See `resolve_vector_ranks()` for more details.
`preference_index`	See `resolve_vector_ranks()` for more details.
`collapse_string`	See `resolve_vector_ranks()` for more details.
`replace_collapsed_rank_by_NA`	(Logical, default TRUE) See `resolve_vector_ranks()` for more details.
`simplify_taxo`	(logical default TRUE). Do we apply the function `simplify_taxo()` to the phyloseq object?
`keep_vsearch_score`	(Logical, default FALSE). If TRUE, the mean and sd of id score are stored in the tax_table.

Details

This function is mainly a wrapper of the work of others. Please cite vsearch and stampa

Value

See param behavior

Author(s)

Adrien Taudière

See Also

`assign_sintax()`, `add_new_taxonomy_pq()`

Examples

```
data_fungi_mini_new <- assign_vsearch_lca(data_fungi_mini,
 ref_fasta = system.file("extdata", "mini_UNITE_fungi.fasta.gz", package = "MiscMetabar"),
  lca_cutoff = 0.9, behavior = "add_to_phyloseq"
)

data_fungi_mini_new2 <- assign_vsearch_lca(data_fungi_mini,
 ref_fasta = system.file("extdata", "mini_UNITE_fungi.fasta.gz", package = "MiscMetabar"),
  id = 0.8, behavior = "add_to_phyloseq", top_hits_only = FALSE
)

data_fungi_mini_new3 <- assign_vsearch_lca(data_fungi_mini,
 ref_fasta = system.file("extdata", "mini_UNITE_fungi.fasta.gz", package = "MiscMetabar"),
 id = 0.5, behavior = "add_to_phyloseq", top_hits_only = FALSE, vote_algorithm = "rel_majority"
)
```

`as_binary_otu_table` *Transform the otu_table of a* `phyloseq-class` *object into a* `phyloseq-class` *object with a binary otu_table.*

Description

Useful to test if the results are not biased by sequences bias that appended during PCR or NGS pipeline.

Usage

```
as_binary_otu_table(physeq, min_number = 1)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`min_number`	(int) the minimum number of sequences to put a 1 in the OTU table.

Value

A physeq object with only 0/1 in the OTU table

Author(s)

Adrien Taudière

Examples

```
data(enterotype)
enterotype_bin <- as_binary_otu_table(enterotype)
```

---

biplot_pq *Visualization of two samples for comparison*

Description

Graphical representation of distribution of taxa across two samples.

Usage

```
biplot_pq(
  physeq,
  fact = NULL,
  merge_sample_by = NULL,
  rarefy_after_merging = FALSE,
  inverse_side = FALSE,
  left_name = NULL,
  left_name_col = "#4B3E1E",
  left_fill = "#4B3E1E",
  left_col = "#f3f2d9",
  right_name = NULL,
  right_name_col = "#1d2949",
  right_fill = "#1d2949",
  right_col = "#1d2949",
  log10trans = TRUE,
  nudge_y = c(0.3, 0.3),
  geom_label = FALSE,
  text_size = 3,
  size_names = 5,
  y_names = NA,
  ylim_modif = c(1, 1),
  nb_samples_info = TRUE,
  plotly_version = FALSE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(default: NULL) Name of the factor in `physeq@sam_data`. If left to NULL use the `left_name` and `right_name` parameter as modality.
`merge_sample_by`	(default: NULL) if not `NULL` samples of physeq are merged using the vector set by `merge_sample_by`. This merging used the `merge_samples2()`. In the case of `biplot_pq()` this must be a factor with two levels only.
`rarefy_after_merging`	Rarefy each sample after merging by the modalities merge_sample_by

`inverse_side`	Inverse the side (put the right modality in the left side).
`left_name`	Name fo the left sample.
`left_name_col`	Color for the left name
`left_fill`	Fill fo the left sample.
`left_col`	Color fo the left sample.
`right_name`	Name fo the right sample.
`right_name_col`	Color for the right name
`right_fill`	Fill fo the right sample.
`right_col`	Color fo the right sample.
`log10trans`	(logical) Does abundancy is log10 transformed ?
`nudge_y`	A parameter to control the y position of abundancy values. If a vector of two values are set. The first value is for the left side. and the second value for the right one. If one value is set, this value is used for both side.
`geom_label`	(default: FALSE, logical) if TRUE use the `ggplot2::geom_label()` function instead of `ggplot2::geom_text()` to indicate the numbers of sequences.
`text_size`	size for the number of sequences
`size_names`	size for the names of the 2 samples
`y_names`	y position for the names of the 2 samples. If NA (default), computed using the maximum abundances values.
`ylim_modif`	vector of two values. Modificator (by a multiplication) of ylim. If one value is set, this value is used for both limits.
`nb_samples_info`	(default: TRUE, logical) if TRUE and merge_sample_by is set, add the number of samples merged for both levels.
`plotly_version`	If TRUE, use `plotly::ggplotly()` to return a interactive ggplot.
`...`	Other arguments for the ggplot function

Value

A plot

Author(s)

Adrien Taudière

Examples

```
data_fungi_2Height <- subset_samples(data_fungi_mini, Height %in% c("Low", "High"))
biplot_pq(data_fungi_2Height, "Height", merge_sample_by = "Height")
```

`blast_pq` *Blast all sequence of* `refseq` *slot of a* `phyloseq-class` *object against a custom database.*

Description

Use the blast software.

Usage

```
blast_pq(
  physeq,
  fasta_for_db = NULL,
  database = NULL,
  blastpath = NULL,
  id_cut = 90,
  bit_score_cut = 50,
  min_cover_cut = 50,
  e_value_cut = 1e-30,
  unique_per_seq = FALSE,
  score_filter = TRUE,
  nproc = 1,
  args_makedb = NULL,
  args_blastn = NULL,
  keep_temporary_files = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fasta_for_db`	Either a DNAStringSet object or a path to a fasta file to make the blast database.
`database`	path to a blast database
`blastpath`	path to blast program
`id_cut`	(default: 90) cut of in identity percent to keep result
`bit_score_cut`	(default: 50) cut of in bit score to keep result The higher the bit-score, the better the sequence similarity. The bit-score is the requires size of a sequence database in which the current match could be found just by chance. The bit-score is a log2 scaled and normalized raw-score. Each increase by one doubles the required database size (2bit-score).
`min_cover_cut`	(default: 50) cut of in query cover (%) to keep result
`e_value_cut`	(default: 1e-30) cut of in e-value (%) to keep result The BLAST E-value is the number of expected hits of similar quality (score) that could be found just by chance.
`unique_per_seq`	(logical, default FALSE) if TRUE only return the better match (higher **bit score**) for each sequence

`score_filter`	(logical, default TRUE) does results are filter by score? If FALSE, `id_cut`,`bit_score_cut`, `e_value_cut` and `min_cover_cut` are ignored
`nproc`	(default: 1) Set to number of cpus/processors to use for blast (args -num_threads for blastn command)
`args_makedb`	Additional arguments passed on to makeblastdb command
`args_blastn`	Additional arguments passed on to blastn command
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files • db.fasta (refseq transformed into a database) • dbase list of files (output of blastn) • blast_result.txt the summary result of blastn using `-outfmt "6 qseqid qlen sseqid slen length p` • temp.fasta if fasta_for_db was a DNAStringSet object.

Value

a blast table

See Also

`blast_to_phyloseq()` to use `refseq` slot as a database

`blast_to_derep` *Blast some sequence against sequences from of a `derep-class` object.*

Description

Use the blast software.

Usage

```
blast_to_derep(
  derep,
  seq2search,
  blastpath = NULL,
  id_cut = 90,
  bit_score_cut = 50,
  min_cover_cut = 50,
  e_value_cut = 1e-30,
  unique_per_seq = FALSE,
  score_filter = FALSE,
  list_no_output_query = FALSE,
  min_length_seq = 200,
  args_makedb = NULL,
  args_blastn = NULL,
  nproc = 1,
  keep_temporary_files = FALSE
)
```

Arguments

`derep`	The result of `dada2::derepFastq()`. A list of `derep-class` object.
`seq2search`	(required) path to a fasta file defining the sequences you want to blast against the taxa (ASV, OTU) sequences from the physeq object.
`blastpath`	path to blast program
`id_cut`	(default: 90) cut of in identity percent to keep result
`bit_score_cut`	(default: 50) cut of in bit score to keep result The higher the bit-score, the better the sequence similarity. The bit-score is the requires size of a sequence database in which the current match could be found just by chance. The bit-score is a log2 scaled and normalized raw-score. Each increase by one doubles the required database size (2bit-score).
`min_cover_cut`	(default: 50) cut of in query cover (%) to keep result
`e_value_cut`	(default: 1e-30) cut of in e-value (%) to keep result The BLAST E-value is the number of expected hits of similar quality (score) that could be found just by chance.
`unique_per_seq`	(logical, default FALSE) if TRUE only return the better match (higher **bit score**) for each sequence
`score_filter`	(logical, default TRUE) does results are filter by score? If FALSE, `id_cut`,`bit_score_cut`, `e_value_cut` and `min_cover_cut` are ignored
`list_no_output_query`	(logical) does the result table include query sequences for which `blastn` does not find any correspondence?
`min_length_seq`	(default: 200) Removed sequences with less than `min_length_seq` from derep before blast. Set to 0 to discard filtering sequences by length.
`args_makedb`	Additional arguments passed on to makeblastdb command
`args_blastn`	Additional arguments passed on to blastn command
`nproc`	(default: 1) Set to number of cpus/processors to use for blast (args -num_threads for blastn command)
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files : • db.fasta (refseq transformed into a database) • dbase list of files (output of blastn) • blast_result.txt the summary result of blastn using `-outfmt "6 qseqid qlen sseqid slen length p`

Value

A blast table

Author(s)

Adrien Taudière

See Also

`blast_pq()` to use `refseq` slot as query sequences against un custom database and `blast_to_phyloseq()` to use `refseq` slot as a database

`blast_to_phyloseq` *Blast some sequence against* `refseq` *slot of a* `phyloseq-class` *object.*

Description

Use the blast software.

Usage

```
blast_to_phyloseq(
  physeq,
  seq2search,
  blastpath = NULL,
  id_cut = 90,
  bit_score_cut = 50,
  min_cover_cut = 50,
  e_value_cut = 1e-30,
  unique_per_seq = FALSE,
  score_filter = TRUE,
  list_no_output_query = FALSE,
  args_makedb = NULL,
  args_blastn = NULL,
  nproc = 1,
  keep_temporary_files = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`seq2search`	(required) path to a fasta file defining the sequences you want to blast against the taxa (ASV, OTU) sequences from the physeq object.
`blastpath`	path to blast program
`id_cut`	(default: 90) cut of in identity percent to keep result
`bit_score_cut`	(default: 50) cut of in bit score to keep result The higher the bit-score, the better the sequence similarity. The bit-score is the requires size of a sequence database in which the current match could be found just by chance. The bit-score is a log2 scaled and normalized raw-score. Each increase by one doubles the required database size (2bit-score).
`min_cover_cut`	(default: 50) cut of in query cover (%) to keep result
`e_value_cut`	(default: 1e-30) cut of in e-value (%) to keep result The BLAST E-value is the number of expected hits of similar quality (score) that could be found just by chance.
`unique_per_seq`	(logical, default FALSE) if TRUE only return the better match (higher **bit score**) for each sequence

`score_filter`	(logical, default TRUE) does results are filter by score? If FALSE, `id_cut`,`bit_score_cut`, `e_value_cut` and `min_cover_cut` are ignored
`list_no_output_query`	(logical) does the result table include query sequences for which `blastn` does not find any correspondence?
`args_makedb`	Additional arguments passed on to makeblastdb command
`args_blastn`	Additional arguments passed on to blastn command
`nproc`	(default: 1) Set to number of cpus/processors to use for blast (args -num_threads for blastn command)
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files • db.fasta (refseq transformed into a database) • dbase list of files (output of blastn) • blast_result.txt the summary result of blastn using `-outfmt "6 qseqid qlen sseqid slen length p`

Value

the blast table

See Also

`blast_pq()` to use `refseq` slot as query sequences against un custom database.

Examples

```
## Not run:
blastpath <- "...YOUR_PATH_TO_BLAST..."
blast_to_phyloseq(data_fungi,
  seq2search = system.file("extdata", "ex.fasta",
    package = "MiscMetabar", mustWork = TRUE
  ),
  blastpath = blastpath
)

## End(Not run)
```

`build_phytree_pq` *Build phylogenetic trees from refseq slot of a phyloseq object*

Description

This function build tree phylogenetic tree and if nb_bootstrap is set, it build also the 3 corresponding bootstrapped tree.

Default parameters are based on doi:10.12688/f1000research.8986.2 and phangorn vignette Estimating phylogenetic trees with phangorn. You should understand your data, especially the markers, before using this function.

Note that phylogenetic reconstruction with markers used for metabarcoding are not robust. You must verify the robustness of your phylogenetic tree using taxonomic classification (see vignette Tree visualization) and bootstrap or multi-tree visualization

Usage

```
build_phytree_pq(
  physeq,
  nb_bootstrap = 0,
  model = "GTR",
  optInv = TRUE,
  optGamma = TRUE,
  rearrangement = "NNI",
  control = phangorn::pml.control(trace = 0),
  optNni = TRUE,
  multicore = FALSE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`nb_bootstrap`	(default 0): If a positive number is set, the function also build 3 bootstrapped trees using `nb_bootstrap` bootstrap samples
`model`	allows to choose an amino acid models or nucleotide model, see `phangorn::optim.pml()` for more details
`optInv`	Logical value indicating whether topology gets optimized (NNI). See `phangorn::optim.pml()` for more details
`optGamma`	Logical value indicating whether gamma rate parameter gets optimized. See `phangorn::optim.pml()` for more details
`rearrangement`	type of tree tree rearrangements to perform, one of "NNI", "stochastic" or "ratchet" see `phangorn::optim.pml()` for more details
`control`	A list of parameters for controlling the fitting process. see `phangorn::optim.pml()` for more details
`optNni`	Logical value indicating whether topology gets optimized (NNI). see `phangorn::optim.pml()` for more details
`multicore`	(logical) whether models should estimated in parallel. see `phangorn::bootstrap.pml()` for more details
`...`	Other params for be passed on to `phangorn::optim.pml()` function

Details

This function is mainly a wrapper of the work of others. Please make a reference to `phangorn` package if you use this function.

Value

A list of phylogenetic tree

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("phangorn")) {
  set.seed(22)
  df <- subset_taxa_pq(data_fungi_mini, taxa_sums(data_fungi_mini) > 9000)
  df_tree <- build_phytree_pq(df, nb_bootstrap = 2)
  plot(df_tree$UPGMA)
  phangorn::plotBS(df_tree$UPGMA, df_tree$UPGMA_bs, main = "UPGMA")
  plot(df_tree$NJ, "unrooted")
  plot(df_tree$ML)

  phangorn::plotBS(df_tree$ML$tree, df_tree$ML_bs, p = 20, frame = "circle")
  phangorn::plotBS(
    df_tree$ML$tree,
    df_tree$ML_bs,
    p = 20,
    frame = "circle",
    method = "TBE"
  )
  plot(phangorn::consensusNet(df_tree$ML_bs))
  plot(phangorn::consensusNet(df_tree$NJ_bs))
  ps_tree <- merge_phyloseq(df, df_tree$ML$tree)
}
```

`chimera_detection_vs` *Detect for chimera taxa using* R*hrefhttps://github.com/torognes/vsearchvsearch*

Description

Use the VSEARCH software.

Usage

```
chimera_detection_vs(
  seq2search,
  nb_seq,
  vsearchpath = "vsearch",
  abskew = 2,
  min_seq_length = 100,
  vsearch_args = "--fasta_width 0",
  keep_temporary_files = FALSE
)
```

Arguments

`seq2search`	(required) a list of DNA sequences coercible by function `Biostrings::DNAStringSet()`
`nb_seq`	(required) a numeric vector giving the number of sequences for each DNA sequences
`vsearchpath`	(default: "vsearch") path to vsearch
`abskew`	(int, default 2) The abundance skew is used to distinguish in a three way alignment which sequence is the chimera and which are the parents. The assumption is that chimeras appear later in the PCR amplification process and are therefore less abundant than their parents. The default value is 2.0, which means that the parents should be at least 2 times more abundant than their chimera. Any positive value equal or greater than 1.0 can be used.
`min_seq_length`	(int, default 100)) Minimum length of sequences to be part of the analysis
`vsearch_args`	(default "–fasta_width 0") A list of other args for vsearch command
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files ? • non_chimeras.fasta • chimeras.fasta • borderline.fasta

Details

This function is mainly a wrapper of the work of others. Please make vsearch.

Value

A list of 3 including non-chimera taxa (`$non_chimera`), chimera taxa (`$chimera`) and bordeline taxa (`$borderline`)

Author(s)

Adrien Taudière

See Also

`chimera_removal_vs()`, `dada2::removeBimeraDenovo()`

Examples

```
chimera_detection_vs(
  seq2search = data_fungi@refseq,
  nb_seq = taxa_sums(data_fungi)
)
```

`chimera_removal_vs` *Search for a list of sequence in an object to remove chimera taxa using* R*hrefhttps://github.com/torognes/vsearchvsearch*

Description

Use the VSEARCH software.

Usage

```
chimera_removal_vs(object, type = "Discard_only_chim", clean_pq = FALSE, ...)
```

Arguments

`object`	(required) A phyloseq-class object or one of dada, derep, data.frame or list coercible to sequences table using the function `dada2::makeSequenceTable()`
`type`	(default "Discard_only_chim"). The type define the type of filtering. • "Discard_only_chim" will only discard taxa classify as chimera by vsearch • "Select_only_non_chim" will only select taxa classify as non-chimera by vsearch(after filtering taxa based on their sequence length by the parameter `min_seq_length` from the `chimera_detection_vs()` function) • "Select_only_chim" will only select taxa classify as chimera by vsearch (after filtering taxa based on their sequence length by the parameter `min_seq_length` from the `chimera_detection_vs()` function)
`clean_pq`	(logical; default FALSE) If TRUE, return the phyloseq object after cleaning using the default parameter of `clean_pq()` function.
`...`	Additional arguments passed on to `chimera_detection_vs()` function

Details

This function is mainly a wrapper of the work of others. Please make vsearch.

Value

- I/ a sequences tables if object is of class dada, derep, data.frame or list.
- II/ a phyloseq object without (or with if type = 'Select_only_chim') chimeric taxa

Author(s)

Adrien Taudière

See Also

`chimera_detection_vs()`, `dada2::removeBimeraDenovo()`

Examples

```
data_fungi_nochim <- chimera_removal_vs(data_fungi)
data_fungi_nochim_16 <- chimera_removal_vs(data_fungi,
  abskew = 16,
  min_seq_length = 10
)
data_fungi_nochim2 <-
  chimera_removal_vs(data_fungi, type = "Select_only_non_chim")
data_fungi_chimera <-
  chimera_removal_vs(data_fungi, type = "Select_only_chim")
```

`circle_pq` *Plot OTU circle for* `phyloseq-class` *object*

Description

Graphical representation of distribution of taxa across a factor.

Usage

```
circle_pq(
  physeq = NULL,
  fact = NULL,
  taxa = "Order",
  nproc = 1,
  add_nb_seq = TRUE,
  rarefy = FALSE,
  min_prop_tax = 0.01,
  min_prop_mod = 0.1,
  gap_degree = NULL,
  start_degree = NULL,
  row_col = NULL,
  grid_col = NULL,
  log10trans = FALSE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required) Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`.
`taxa`	(default: 'Order') Name of the taxonomic rank of interest
`nproc`	(default 1) Set to number of cpus/processors to use for parallelization

`add_nb_seq`	(default: TRUE) Represent the number of sequences or the number of OTUs (add_nb_seq = FALSE)
`rarefy`	(logical) Does each samples modalities need to be rarefy in order to compare them with the same amount of sequences?
`min_prop_tax`	(default: 0.01) The minimum proportion for taxa to be plotted
`min_prop_mod`	(default: 0.1) The minimum proportion for modalities to be plotted
`gap_degree`	Gap between two neighbour sectors. It can be a single value or a vector. If it is a vector, the first value corresponds to the gap after the first sector.
`start_degree`	The starting degree from which the circle begins to draw. Note this degree is measured in the standard polar coordinate which means it is always reverse-clockwise.
`row_col`	Color vector for row
`grid_col`	Grid colors which correspond to sectors. The length of the vector should be either 1 or the number of sectors. It's preferred that grid_col is a named vector of which names correspond to sectors. If it is not a named vector, the order of grid_col corresponds to order of sectors.
`log10trans`	(logical) Should sequence be log10 transformed (more precisely by log10(1+x))?
`...`	Additional arguments passed on to `chordDiagram` or `circos.par`

Value

A `chordDiagram` plot representing the distribution of OTUs or sequences in the different modalities of the factor fact

Author(s)

Adrien Taudière

See Also

`chordDiagram`

`circos.par`

Examples

```
if (requireNamespace("pbapply")) {
  data("GlobalPatterns", package = "phyloseq")
  GP <- subset_taxa(GlobalPatterns, GlobalPatterns@tax_table[, 1] == "Archaea")
  circle_pq(GP, "SampleType")
  circle_pq(GP, "SampleType", add_nb_seq = FALSE)
  circle_pq(GP, "SampleType", taxa = "Class")
}
```

`clean_pq` *Clean phyloseq object by removing empty samples and taxa*

Description

In addition, this function check for discrepancy (and rename) between (i) taxa names in refseq, taxonomy table and otu_table and between (ii) sample names in sam_data and otu_table.

Usage

```
clean_pq(
  physeq,
  remove_empty_samples = TRUE,
  remove_empty_taxa = TRUE,
  clean_samples_names = TRUE,
  silent = FALSE,
  verbose = FALSE,
  force_taxa_as_columns = FALSE,
  force_taxa_as_rows = FALSE,
  reorder_taxa = FALSE,
  rename_taxa = FALSE,
  simplify_taxo = FALSE,
  prefix_taxa_names = "_Taxa"
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`remove_empty_samples`	(logical) Do you want to remove samples without sequences (this is done after removing empty taxa)
`remove_empty_taxa`	(logical) Do you want to remove taxa without sequences (this is done before removing empty samples)
`clean_samples_names`	(logical) Do you want to clean samples names?
`silent`	(logical) If true, no message are printing.
`verbose`	(logical) Additional informations in the message the verbose parameter overwrite the silent parameter.
`force_taxa_as_columns`	(logical) If true, if the taxa are rows transpose the otu_table and set taxa_are_rows to false
`force_taxa_as_rows`	(logical) If true, if the taxa are columns transpose the otu_table and set taxa_are_rows to true

`reorder_taxa`	(logical) if TRUE the otu_table is ordered by the number of sequences of taxa (ASV, OTU) in descending order. Default to FALSE.
`rename_taxa`	(logical) if TRUE, taxa (ASV, OTU) are renamed by their position in the OTU_table and prefix_taxa_names param (by default: Taxa_1, Taxa_2, ...). Default to FALSE. If rename taxa (ASV, OTU) is true, the taxa (ASV, OTU) names in verbose information can be misleading.
`simplify_taxo`	(logical) if TRUE, correct the taxonomy_table using the `MiscMetabar::simplify_taxo()` function
`prefix_taxa_names`	(default "Taxa_"): the prefix of taxa names (eg. "ASV_" or "OTU_")

Value

A new `phyloseq-class` object

Author(s)

Adrien Taudière

`compare_pairs_pq` *Compare samples in pairs using diversity and number of ASV including shared ASV.*

Description

For the moment refseq slot need to be not Null.

Usage

```
compare_pairs_pq(
  physeq = NULL,
  bifactor = NULL,
  modality = NULL,
  merge_sample_by = NULL,
  nb_min_seq = 0,
  veg_index = "shannon",
  na_remove = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`bifactor`	(required) a factor (present in the `sam_data` slot of the physeq object) presenting the pair names
`modality`	the name of the column in the `sam_data` slot of the physeq object to split samples by pairs

`merge_sample_by`	a vector to determine which samples to merge using the `merge_samples2()` function. Need to be in `physeq@sam_data`
`nb_min_seq`	minimum number of sequences per sample to count the ASV/OTU
`veg_index`	(default: "shannon") index for the `vegan::diversity` function
`na_remove`	(logical, default TRUE) If set to TRUE, remove samples with NA in the variables set in bifactor, modality and merge_sample_by. NA in variables are well managed even if na_remove = FALSE, so na_remove may be useless.

Value

A tibble with information about the number of shared ASV, shared number of sequences and diversity

Examples

```
data_fungi_low_high <- subset_samples(data_fungi, Height %in% c("Low", "High"))
compare_pairs_pq(data_fungi_low_high, bifactor = "Height", merge_sample_by = "Height")
compare_pairs_pq(data_fungi_low_high,
  bifactor = "Height",
  merge_sample_by = "Height", modality = "Time"
)
```

`count_seq` *Count sequences in fasta or fastq file*

Description

Use grep to count the number of line with only one '+' (fastq, fastq.gz) or lines starting with a '>' (fasta) to count sequences.

Usage

```
count_seq(file_path = NULL, folder_path = NULL, pattern = NULL)
```

Arguments

`file_path`	The path to a fasta, fastq or fastq.gz file
`folder_path`	The path to a folder with fasta, fastq or fastq.gz files
`pattern`	A pattern to filter files in a folder. E.g. *R2*

Value

the number of sequences

Author(s)

Adrien Taudière

Examples

```
count_seq(file_path = system.file(
  "extdata",
  "ex.fasta",
  package = "MiscMetabar",
  mustWork = TRUE
))
count_seq(
  folder_path = system.file("extdata", package = "MiscMetabar"),
  pattern = "*.fasta"
)
```

`cutadapt_remove_primers`

Remove primers using R*hrefhttps://github.com/marcelm/cutadapt/cutadapt*

Description

You need to install Cutadapt. See also https://github.com/VascoElbrecht/JAMP/blob/master/JAMP/R/Cutadapt.R for another call to cutadapt from R

Usage

```
cutadapt_remove_primers(
  path_to_fastq,
  primer_fw = NULL,
  primer_rev = NULL,
  folder_output = "wo_primers",
  nproc = 1,
  pattern = "fastq.gz",
  pattern_R1 = "_R1",
  pattern_R2 = "_R2",
  nb_files = Inf,
  cmd_is_run = TRUE,
  return_file_path = FALSE,
  args_before_cutadapt =
   "source ~/miniconda3/etc/profile.d/conda.sh && conda activate cutadaptenv && "
)
```

Arguments

`path_to_fastq`	(Required) A path to a folder with fastq files. See `list_fastq_files()` for help.
`primer_fw`	(Required, String) The forward primer DNA sequence.
`primer_rev`	(String) The reverse primer DNA sequence.
`folder_output`	The path to a folder for output files

`nproc`	(default 1) Set to number of cpus/processors to use for the clustering
`pattern`	a pattern to filter files (passed on to list.files function).
`pattern_R1`	a pattern to filter R1 files (default "*R1*")
`pattern_R2`	a pattern to filter R2 files (default "*R2*")
`nb_files`	the number of fastq files to list (default FALSE)
`cmd_is_run`	(logical, default TRUE) Do the cutadapt command is run. If set to FALSE, the only effect of the function is to return a list of command to manually run in a terminal.
`return_file_path`	(logical, default FALSE) If true, the function return the path of the output folder (param `folder_output`). Useful in targets workflow
`args_before_cutadapt`	(String) A one line bash command to run before to run cutadapt. For examples, "source ~/miniconda3/etc/profile.d/conda.sh && conda activate cutadaptenv &&" allow to bypass the conda init which asks to restart the shell

Details

This function is mainly a wrapper of the work of others. Please cite cutadapt (doi:10.14806/ej.17.1.200).

Value

a list of command or if `return_file_path` is TRUE, the path to the output folder

Author(s)

Adrien Taudière

Examples

```
## Not run:
cutadapt_remove_primers(system.file("extdata", package = "MiscMetabar"),
  "TTC",
  "GAA",
  folder_output = tempdir()
)

cutadapt_remove_primers(
  system.file("extdata",
    package = "dada2"
  ),
  pattern_R1 = "F.fastq.gz",
  pattern_R2 = "R.fastq.gz",
  primer_fw = "TTC",
  primer_rev = "GAA",
  folder_output = tempdir()
)
```

```
cutadapt_remove_primers(
  system.file("extdata",
    package = "dada2"
  ),
  pattern_R1 = "F.fastq.gz",
  primer_fw = "TTC",
  folder_output = tempdir(),
  cmd_is_run = FALSE
)

unlink(tempdir(), recursive = TRUE)

## End(Not run)
```

`data_fungi` *Fungal OTU in phyloseq format*

Description

Fungal OTU in phyloseq format

Usage

```
data(data_fungi)
```

Format

A physeq object containing 1420 taxa with references sequences described by 14 taxonomic ranks and 185 samples described by 7 sample variables:

- *X*: the name of the fastq-file
- *Sample_names*: the names of ... the samples
- *Treename*: the name of an tree
- *Sample_id*: identifier for each sample
- *Height*: height of the sample in the tree
- *Diameter*: diameter of the trunk
- *Time*: time since the dead of the tree

`data_fungi_mini` *Fungal OTU in phyloseq format*

Description

It is a subset of the data_fungi dataset including only Basidiomycota with more than 5000 sequences.

Usage

```
data(data_fungi_mini)
```

```
data(data_fungi_mini)
```

Format

A physeq object containing 45 taxa with references sequences described by 14 taxonomic ranks and 137 samples described by 7 sample variables:

- *X*: the name of the fastq-file
- *Sample_names*: the names of ... the samples
- *Treename*: the name of an tree
- *Sample_id*: identifier for each sample
- *Height*: height of the sample in the tree
- *Diameter*: diameter of the trunk
- *Time*: time since the dead of the tree

A physeq object containing 45 taxa with references sequences described by 14 taxonomic ranks and 137 samples described by 7 sample variables:

- *X*: the name of the fastq-file
- *Sample_names*: the names of ... the samples
- *Treename*: the name of an tree
- *Sample_id*: identifier for each sample
- *Height*: height of the sample in the tree
- *Diameter*: diameter of the trunk
- *Time*: time since the dead of the tree

Details

Obtain using `data_fungi_mini <- subset_taxa(data_fungi, Phylum == "Basidiomycota")` and then `data_fungi_mini <- subset_taxa_pq(data_fungi_mini, colSums(data_fungi_mini@otu_table) > 5000)`

`data_fungi_sp_known` *Fungal OTU in phyloseq format*

Description

It is a subset of the data_fungi dataset including only taxa with information at the species level

Usage

```
data(data_fungi_sp_known)
```

Format

A physeq object containing 651 taxa with references sequences described by 14 taxonomic ranks and 185 samples described by 7 sample variables:

- *X*: the name of the fastq-file
- *Sample_names*: the names of ... the samples
- *Treename*: the name of an tree
- *Sample_id*: identifier for each sample
- *Height*: height of the sample in the tree
- *Diameter*: diameter of the trunk
- *Time*: time since the dead of the tree

Details

Obtain using `data_fungi_sp_known <- subset_taxa(data_fungi, !is.na(data_fungi@tax_table[,"Species"]))`

`diff_fct_diff_class` *Compute different functions for different class of vector.*

Description

Mainly an internal function useful in "sapply(..., tapply)" methods

Usage

```
diff_fct_diff_class(
  x,
  numeric_fonction = mean,
  logical_method = "TRUE_if_one",
  character_method = "unique_or_na",
  ...
)
```

Arguments

`x`	: a vector
`numeric_fonction`	: a function for numeric vector. For ex. `sum` or `mean`
`logical_method`	: A method for logical vector. One of : • TRUE_if_one (default) • NA_if_not_all_TRUE • FALSE_if_not_all_TRUE
`character_method`	: A method for character vector (and factor). One of : • unique_or_na (default) • more_frequent • more_frequent_without_equality
`...`	Additional arguments passed on to the numeric function (ex. na.rm=TRUE)

Value

a single value

Author(s)

Adrien Taudière

Examples

```
diff_fct_diff_class(
  data_fungi@sam_data$Sample_id,
  numeric_fonction = sum,
  na.rm = TRUE
)
diff_fct_diff_class(
  data_fungi@sam_data$Time,
  numeric_fonction = mean,
  na.rm = TRUE
)
diff_fct_diff_class(
  data_fungi@sam_data$Height == "Low",
  logical_method = "TRUE_if_one"
)
diff_fct_diff_class(
  data_fungi@sam_data$Height == "Low",
  logical_method = "NA_if_not_all_TRUE"
)
diff_fct_diff_class(
  data_fungi@sam_data$Height == "Low",
  logical_method = "FALSE_if_not_all_TRUE"
)
diff_fct_diff_class(
  data_fungi@sam_data$Height,
```

```
  character_method = "unique_or_na"
)
diff_fct_diff_class(
  c("IE", "IE"),
  character_method = "unique_or_na"
)
diff_fct_diff_class(
  c("IE", "IE", "TE", "TE"),
  character_method = "more_frequent"
)
diff_fct_diff_class(
  c("IE", "IE", "TE", "TE"),
  character_method = "more_frequent_without_equality"
)
```

`distri_1_taxa` *Distribution of sequences across a factor for one taxon*

Description

Focus on one taxon and one factor.

Usage

```
distri_1_taxa(physeq, fact, taxa_name, digits = 2)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required) Name of the factor in `physeq@sam_data` used to plot different lines
`taxa_name`	(required): the name of the taxa
`digits`	(default = 2) integer indicating the number of decimal places to be used (see `?round` for more information)

Value

a dataframe with levels as rows and information as column :

- the number of sequences of the taxa (nb_seq)
- the number of samples of the taxa (nb_samp)
- the mean (mean_nb_seq) and standard deviation (sd_nb_seq) of the *nb_seq*
- the mean (mean_nb_seq_when_present) *nb_seq* excluding samples with zero
- the total number of samples (nb_total_samp)
- the proportion of samples with the taxa

Author(s)

Adrien Taudière

Examples

```
distri_1_taxa(data_fungi, "Height", "ASV2")
distri_1_taxa(data_fungi, "Time", "ASV81", digits = 1)
```

`dist_bycol` *Compute paired distances among matrix (e.g. otu_table)*

Description

May be used to verify ecological distance among samples.

Usage

```
dist_bycol(x, y, method = "bray", nperm = 99, ...)
```

Arguments

`x`	(required) A first matrix.
`y`	(required) A second matrix.
`method`	(default: 'bray') the method to use internally in the vegdist function.
`nperm`	(int) The number of permutations to perform.
`...`	Additional arguments passed on to`vegan::vegdist` function

Value

A list of length two : (i) a vector of observed distance ($obs) and (ii) a matrix of the distance after randomization ($null)

Note

the first column of the first matrix is compare to the first column of the second matrix, the second column of the first matrix is compare to the second column of the second matrix and so on.

Author(s)

Adrien Taudière

See Also

`vegdist`

`dist_pos_control` *Calculate ecological distance among positive controls vs distance for all samples*

Description

Compute distance among positive controls, i.e. samples which are duplicated to test for variation, for example in (i) a step in the sampling, (ii) a step in the extraction, (iii) a step in the sequencing.

Usage

```
dist_pos_control(physeq, samples_names, method = "bray")
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`samples_names`	(required) a vector of names for samples with positives controls of the same samples having the same name
`method`	(default: "bray") a method to calculate the distance, parsed to `vegan::vegdist()`. See ?vegdist for a list of possible values.

Value

A list of two data-frames with (i) the distance among positive controls and (ii) the distance among all samples

Author(s)

Adrien Taudière

Examples

```
data("enterotype")
sam_name_factice <- gsub("TS1_V2", "TS10_V2", sample_names(enterotype))
res_dist_cont <- dist_pos_control(enterotype, sam_name_factice)
hist(unlist(res_dist_cont$distAllSamples))
abline(
  v = mean(unlist(res_dist_cont$dist_controlontrolSamples), na.rm = TRUE),
  col = "red", lwd = 3
)
```

`fac2col` *Translates a factor into colors.*

Description

Translates a factor into colors.

Usage

```
fac2col(x, col.pal = funky_color, na.col = "grey", seed = NULL)
```

Arguments

`x`	a numeric vector (for num2col) or a vector converted to a factor (for fac2col).
`col.pal`	(default funky_color) a function generating colors according to a given palette.
`na.col`	(default grey) the color to be used for missing values (NAs)
`seed`	(default NULL) a seed for R's random number generated, used to fix the random permutation of colors in the palette used; if NULL, no randomization is used and the colors are taken from the palette according to the ordering of the levels

Value

a color vector

Author(s)

Thibaut Jombart in `adegenet` package

See Also

The R package RColorBrewer, proposing a nice selection of color palettes. The viridis package, with many excellent palettes

---

`filter_asv_blast` *Filter undesirable taxa using blast against a custom database.*

Description

Use the blast software.

Usage

```
filter_asv_blast(
  physeq,
  fasta_for_db = NULL,
  database = NULL,
  clean_pq = TRUE,
  add_info_to_taxtable = TRUE,
  id_filter = 90,
  bit_score_filter = 50,
  min_cover_filter = 50,
  e_value_filter = 1e-30,
  ...
)

filter_taxa_blast(
  physeq,
  fasta_for_db = NULL,
  database = NULL,
  clean_pq = TRUE,
  add_info_to_taxtable = TRUE,
  id_filter = 90,
  bit_score_filter = 50,
  min_cover_filter = 50,
  e_value_filter = 1e-30,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fasta_for_db`	path to a fasta file to make the blast database
`database`	path to a blast database
`clean_pq`	(logical) If set to TRUE, empty samples and empty taxa (ASV, OTU) are discarded after filtering.
`add_info_to_taxtable`	(logical, default TRUE) Does the blast information are added to the taxtable ?
`id_filter`	(default: 90) cut of in identity percent to keep result
`bit_score_filter`	(default: 50) cut of in bit score to keep result The higher the bit-score, the better the sequence similarity. The bit-score is the requires size of a sequence database in which the current match could be found just by chance. The bit-score is a log2 scaled and normalized raw-score. Each increase by one doubles the required database size (2bit-score).
`min_cover_filter`	(default: 50) cut of in query cover (%) to keep result

`e_value_filter`	(default: 1e-30) cut of in e-value (%) to keep result The BLAST E-value is the number of expected hits of similar quality (score) that could be found just by chance.
`...`	Additional arguments passed on toblast_pq() function. See ?blast_pq. Note that params `unique_per_seq` must be left to TRUE and `score_filter` must be left to FALSE.

Value

A new `phyloseq-class` object.

`filter_trim` *A wrapper of the function* `dada2::filterAndTrim()` *to use in* R*hrefhttps://books.ropensci.org/targets/targets pipeline*

Description

This function filter and trim (with parameters passed on to `dada2::filterAndTrim()` function) forward sequences or paired end sequence if 'rev' parameter is set. It return the list of files to subsequent analysis in a targets pipeline.

Usage

```
filter_trim(
  fw = NULL,
  rev = NULL,
  output_fw = file.path(paste(getwd(), "/output/filterAndTrim_fwd", sep = "")),
  output_rev = file.path(paste(getwd(), "/output/filterAndTrim_rev", sep = "")),
  return_a_vector = FALSE,
  ...
)
```

Arguments

`fw`	(required) a list of forward fastq files
`rev`	a list of reverse fastq files for paired end trimming
`output_fw`	Path to output folder for forward files. By default, this function will create a folder "output/filterAndTrim_fwd" in the current working directory.
`output_rev`	Path to output folder for reverse files. By default, this function will create a folder "output/filterAndTrim_fwd" in the current working directory.
`return_a_vector`	(logical, default FALSE) If true, the return is a vector of path (usefull when used with targets::tar_targets(..., format="file"))
`...`	Other parameters passed on to `dada2::filterAndTrim()` function.

Value

A list of files. If rev is set, will return a list of two lists. The first list is a list of forward files, and the second one is a list of reverse files.

Author(s)

Adrien Taudière

See Also

`dada2::filterAndTrim()`

Examples

```
testFastqs_fw <- c(
  system.file("extdata", "sam1F.fastq.gz", package = "dada2"),
  system.file("extdata", "sam2F.fastq.gz", package = "dada2")
)
testFastqs_rev <- c(
  system.file("extdata", "sam1R.fastq.gz", package = "dada2"),
  system.file("extdata", "sam2R.fastq.gz", package = "dada2")
)

filt_fastq_fw <- filter_trim(testFastqs_fw, output_fw = tempdir())
derep_fw <- derepFastq(filt_fastq_fw[1])
derep_fw

## Not run:
filt_fastq_pe <- filter_trim(testFastqs_fw,
  testFastqs_rev,
  output_fw = paste0(tempdir(), "/", "fw"),
  output_rev = paste0(tempdir(), "rev")
)
derep_fw_pe <- derepFastq(filt_fastq_pe[[1]])
derep_rv_pe <- derepFastq(filt_fastq_pe[[2]])
derep_fw_pe
derep_rv_pe

## End(Not run)
```

`filt_taxa_pq` *Filter taxa of a phyloseq object based on the minimum number of sequences/samples*

Description

Basically a wraper of `subset_taxa_pq()`.

Usage

```
filt_taxa_pq(
  physeq,
  min_nb_seq = NULL,
  min_occurence = NULL,
  combination = "AND",
  clean_pq = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`min_nb_seq`	(int default NULL) minimum number of sequences by taxa.
`min_occurence`	(int default NULL) minimum number of sample by taxa.
`combination`	Either "AND" (default) or "OR". If set to "AND" and both min_nb_seq and min_occurence are not NULL, the taxa must match the two condition to passe the filter. If set to "OR", taxa matching only one condition are kept.
`clean_pq`	(logical) If set to TRUE, empty samples and empty taxa (ASV, OTU) are discarded after filtering.

Value

a new phyloseq object

Author(s)

Adrien Taudière

Examples

```
filt_taxa_pq(data_fungi, min_nb_seq = 20)
filt_taxa_pq(data_fungi, min_occurence = 2)
filt_taxa_pq(data_fungi,
  min_occurence = 2,
  min_nb_seq = 10, clean_pq = FALSE
)
filt_taxa_pq(data_fungi,
  min_occurence = 2,
  min_nb_seq = 10,
  combination = "OR"
)
```

`filt_taxa_wo_NA` *Filter taxa by cleaning taxa with NA at given taxonomic rank(s)*

Description

Basically a wrapper of subset_taxa_pq()

Usage

```
filt_taxa_wo_NA(
  physeq,
  taxa_ranks = NULL,
  n_NA = 0,
  verbose = TRUE,
  NA_equivalent = NULL,
  clean_pq = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`taxa_ranks`	A vector of taxonomic ranks. For examples c("Family","Genus"). If taxa_ranks is NULL (default), all ranks are used, i.e. all taxa with at least 1 NA will be filtered out. Numeric position of taxonomic ranks can also be used.
`n_NA`	(int default = 0). Number of allowed NA by taxa in the list of the taxonomic ranks
`verbose`	(logical). If TRUE, print additional information.
`NA_equivalent`	(vector of character, default NULL). Exact matching of the character listed in the vector are converted as NA before to filter out taxa.
`clean_pq`	(logical, default TRUE) If set to TRUE, empty samples are discarded after filtering. See `clean_pq()`.

Value

An object of class phyloseq

Author(s)

Adrien Taudière

See Also

`subset_taxa_pq()`

Examples

```
data_fungi_wo_NA <- filt_taxa_wo_NA(data_fungi)
filt_taxa_wo_NA(data_fungi, n_NA = 1)
filt_taxa_wo_NA(data_fungi, taxa_ranks = c(1:3))

filt_taxa_wo_NA(data_fungi, taxa_ranks = c("Trait", "Confidence.Ranking"))
filt_taxa_wo_NA(data_fungi,
  taxa_ranks = c("Trait", "Confidence.Ranking"),
  NA_equivalent = c("-", "NULL")
)
```

`format2dada2` *Format a fasta database in dada2 format*

Description

First format in sintax format and then in dada2 format

Usage

```
format2dada2(
  fasta_db = NULL,
  taxnames = NULL,
  output_path = NULL,
  from_sintax = TRUE,
  pattern_to_remove = NULL,
  ...
)
```

Arguments

`fasta_db`	A link to a fasta files
`taxnames`	A list of names to format. You must specify either fasta_db OR taxnames, not both.
`output_path`	(optional) A path to an output fasta files. Only used if fasta_db is set.
`from_sintax`	(logical, default FALSE) Is the original fasta file in sintax format?
`pattern_to_remove`	(a regular expression) Define a pattern to remove. For example, pattern_to_remove = "|rep.*" remove all character after '|rep' to force `dada2::assignTaxonomy()` to not use the database as a Unite-formated database
`...`	Additional arguments passed on to `format2sintax()` function

Value

Either an object of class DNAStringSet or a vector of reformated names

Author(s)

Adrien Taudière

See Also

`format2dada2_species()`, `format2sintax()`

`format2dada2_species` *Format a fasta database in dada2 format for Species assignment*

Description

First format in sintax format and then in dada2 format

Usage

```
format2dada2_species(
  fasta_db = NULL,
  taxnames = NULL,
  from_sintax = FALSE,
  output_path = NULL,
  ...
)
```

Arguments

`fasta_db`	A link to a fasta files
`taxnames`	A list of names to format. You must specify either fasta_db OR taxnames, not both.
`from_sintax`	(logical, default FALSE) Is the original fasta file in sintax format?
`output_path`	(optional) A path to an output fasta files. Only used if fasta_db is set.
`...`	Additional arguments passed on to `format2sintax()` function

Value

Either an object of class DNAStringSet or a vector of reformated names

Author(s)

Adrien Taudière

See Also

`format2dada2_species()`, `format2sintax()`

`format2sintax` *Format a fasta database in sintax format*

Description

Only tested with Unite and Eukaryome fasta file for the moment. Rely on the presence of the pattern pattern_tax default "k__" to format the header.

A reference database in sintax format contain taxonomic information in the header of each sequence in the form of a string starting with ";tax=" and followed by a comma-separated list of up to nine taxonomic identifiers. Each taxonomic identifier must start with an indication of the rank by one of the letters d (for domain) k (kingdom), p (phylum), c (class), o (order), f (family), g (genus), s (species), or t (strain). The letter is followed by a colon (:) and the name of that rank. Commas and semicolons are not allowed in the name of the rank. Non-ascii characters should be avoided in the names.

Example:

\>X80725_S000004313;tax=d:Bacteria,p:Proteobacteria,c:Gammaproteobacteria,o:Enterobacteriales,f:Enterobacteriaceae,g
12_substr._MG1655

Usage

```
format2sintax(
  fasta_db = NULL,
  taxnames = NULL,
  pattern_tax = "k__",
  pattern_sintax = "tax=k:",
  output_path = NULL
)
```

Arguments

`fasta_db`	A link to a fasta files
`taxnames`	A list of names to format. You must specify either fasta_db OR taxnames, not both.
`pattern_tax`	(default "k__") The pattern to replace by pattern_sintax.
`pattern_sintax`	(default "tax=k:") Useless for most users. Sometimes you may want to replacte by "tax=d:" (d for domain instead of kingdom).
`output_path`	(optional) A path to an output fasta files. Only used if fasta_db is set.

Value

Either an object of class DNAStringSet or a vector of reformated names

Author(s)

Adrien Taudière

See Also

`format2dada2_species()`, `format2dada2()`

`formattable_pq` *Create a visualization table to describe taxa distribution across a modality*

Description

Allow to visualize a table with graphical input.

Usage

```
formattable_pq(
  physeq,
  modality,
  taxonomic_levels = c("Phylum", "Order", "Family", "Genus"),
  min_nb_seq_taxa = 1000,
  log10trans = FALSE,
  void_style = FALSE,
  lev_col_taxa = "Phylum",
  arrange_by = "nb_seq",
  descending_order = TRUE,
  na_remove = TRUE,
  formattable_args = NULL
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`modality`	(required) The name of a column present in the `@sam_data` slot of the physeq object. Must be a character vector or a factor.
`taxonomic_levels`	(default = c("Phylum", "Order", "Family", "Genus")) The taxonomic levels (must be present in the `@sam_data` slot) you want to see and/or used (for example to compute a color) in the table.
`min_nb_seq_taxa`	(default = 1000) filter out taxa with less than `min_nb_seq_taxa` sequences
`log10trans`	(logical, default TRUE) Do sequences count is log10 transformed (using log10(x + 1) to allow 0)
`void_style`	(logical, default FALSE) Do the default style is discard ?
`lev_col_taxa`	Taxonomic level used to plot the background color of taxa names
`arrange_by`	The column used to sort the table. Can take the values NULL, "proportion_samp", "nb_seq" (default), , "nb_sam" "OTU", or a column names from the levels of modality or from taxonomic levels

`descending_order`	(logical, default TRUE) Do we use descending order when sort the table (if arrange_by is not NULL) ?
`na_remove`	(logical, default TRUE) if TRUE remove all the samples with NA in the `split_by` variable of the `physeq@sam_data` slot
`formattable_args`	Other args to the formattable function. See examples and `formattable::formattable()`

Details

This function is mainly a wrapper of the work of others. Please make a reference to `formattable::formattable()` if you use this function.

Value

A datatable

Author(s)

Adrien Taudière

See Also

`formattable::formattable()`

Examples

```
if (requireNamespace("formattable")) {
  ## Distribution of the nb of sequences per OTU across Height
  ##   modality (nb of sequences are log-transformed).
  ## Only OTU with more than 10000 sequences are taking into account
  ## The Phylum column is discarded
  formattable_pq(
    data_fungi,
    "Height",
    min_nb_seq_taxa = 10000,
    formattable_args = list("Phylum" = FALSE),
    log10trans = TRUE
  )

  ## Distribution of the nb of samples per OTU across Height modality
  ## Only OTU  present in more than 50 samples are taking into account
  formattable_pq(
    as_binary_otu_table(data_fungi),
    "Height",
    min_nb_seq_taxa = 50,
    formattable_args = list("nb_seq" = FALSE),
  )

  ## Distribution of the nb of sequences per OTU across Time modality
  ##  arranged by Family Name in ascending order.
```

```
  ## Only OTU with more than 10000 sequences are taking into account
  ## The Phylum column is discarded
  formattable_pq(
    data_fungi,
    "Time",
    min_nb_seq_taxa = 10000,
    taxonomic_levels = c("Order", "Family", "Genus", "Species"),
    formattable_args = list(
      Order = FALSE,
      Species = formattable::formatter(
        "span",
        style = x ~ formattable::style(
          "font-style" = "italic",
          `color` = ifelse(is.na(x), "white", "grey")
        )
      )
    ),
    arrange_by = "Family",
    descending_order = FALSE
  )
}

if (requireNamespace("formattable")) {
  ## Distribution of the nb of sequences per OTU across Height modality
  ##  (nb of sequences are log-transformed).
  ## OTU name background is light gray for Basidiomycota
  ##  and dark grey otherwise (Ascomycota)
  ## A different color is defined for each modality level
  formattable_pq(
    data_fungi,
    "Height",
    taxonomic_levels = c("Phylum", "Family", "Genus"),
    void_style = TRUE,
    formattable_args = list(
      OTU = formattable::formatter(
        "span",
        style = ~ formattable::style(
          "display" = "block",
          `border-radius` = "5px",
          `background-color` = ifelse(Phylum == "Basidiomycota", transp("gray"), "gray")
        ),
        `padding-right` = "2px"
      ),
      High = formattable::formatter(
        "span",
        style = x ~ formattable::style(
          "font-size" = "80%",
          "display" = "inline-block",
          direction = "rtl",
          `border-radius` = "0px",
          `padding-right` = "2px",
          `background-color` = formattable::csscolor(formattable::gradient(
            as.numeric(x), transp("#1a91ff"), "#1a91ff"
```

```
          )),
        width = formattable::percent(formattable::proportion(as.numeric(x), na.rm = TRUE))
        )
      ),
      Low = formattable::formatter(
        "span",
        style = x ~ formattable::style(
          "font-size" = "80%",
          "display" = "inline-block",
          direction = "rtl",
          `border-radius` = "0px",
          `padding-right` = "2px",
          `background-color` = formattable::csscolor(formattable::gradient(
            as.numeric(x),
            transp("green"), "green"
          )),
        width = formattable::percent(formattable::proportion(as.numeric(x), na.rm = TRUE))
        )
      ),
      Middle = formattable::formatter(
        "span",
        style = x ~ formattable::style(
          "font-size" = "80%",
          "display" = "inline-block",
          direction = "rtl",
          `border-radius` = "0px",
          `padding-right` = "2px",
          `background-color` = formattable::csscolor(formattable::gradient(
            as.numeric(x), transp("orange"), "orange"
          )),
        width = formattable::percent(formattable::proportion(as.numeric(x), na.rm = TRUE))
        )
      )
    )
  )
}
```

`funguild_assign` *Assign Guilds to Organisms Based on Taxonomic Classification*

Description

The original function and documentation was written by Brendan Furneaux in the FUNGuildR package.

These functions have identical behavior if supplied with a database; however they download the database corresponding to their name by default.

Taxa present in the database are matched to the taxa present in the supplied `otu_table` by exact name. In the case of multiple matches, the lowest (most specific) rank is chosen. No attempt is made to check or correct the classification in `otu_table$Taxonomy`.

Usage

```
funguild_assign(
  otu_table,
  db_url = NULL,
  db_funguild = NULL,
  tax_col = "Taxonomy"
)
```

Arguments

`otu_table`	A `data.frame` with a `character` column named "`Taxonomy`" (or another name as specified in `tax_col`), as well as any other columns. Each entry in "`otu_table$Taxonomy`" should be a comma-, colon-, underscore-, or semicolon-delimited classification of an organism. Rank indicators as given by Sintax ("`k:`", "`p:`"...) or Unite ("`k__`, "`p__`", ...) are also allowed. A `character` vector, representing only the taxonomic classification, is also accepted.
`db_url`	a length 1 character string giving the URL to retrieve the database from
`db_funguild`	A `data.frame` representing the FUNGuild as returned by `get_funguild_db()` If not supplied, the default database will be downloaded.
`tax_col`	A `character` string, optionally giving an alternate column name in `otu_table` to use instead of `otu_table$Taxonomy`.

Value

A `tibble::tibble` containing all columns of `otu_table`, plus relevant columns of information from the FUNGuild

Author(s)

Brendan Furneaux (orcid: 0000-0003-3522-7363), modified by Adrien Taudière

References

Nguyen NH, Song Z, Bates ST, Branco S, Tedersoo L, Menke J, Schilling JS, Kennedy PG. 2016. *FUNGuild: An open annotation tool for parsing fungal community datasets by ecological guild.* Fungal Ecology 20:241-248.

`funky_color` *Funky palette color*

Description

Funky palette color

Usage

```
funky_color(n)
```

Arguments

`n`	a number of colors

Value

a color palette

Author(s)

Thibaut Jombart in `adegenet` package

See Also

The R package RColorBrewer, proposing a nice selection of color palettes. The viridis package, with many excellent palettes

---

`get_file_extension` *Get the extension of a file*

Description

Internally used in `count_seq()`. Warning: don't work when there is '.' in the name of the file before the extension

Usage

```
get_file_extension(file_path)
```

Arguments

`file_path`	(required): path to a file

Value

The extension of a file.

Author(s)

Adrien Taudière

`get_funguild_db` *Retrieve the FUNGuild database*

Description

The original function and documentation was written by Brendan Furneaux in the FUNGuildR package.

Please cite this publication (doi:10.1016/j.funeco.2015.06.006).

Usage

```
get_funguild_db(db_url = "http://www.stbates.org/funguild_db_2.php")
```

Arguments

`db_url`	a length 1 character string giving the URL to retrieve the database from

Value

a `tibble::tibble` containing the database, which can be passed to the db argument of `funguild_assign()`

Author(s)

Brendan Furneaux (orcid: 0000-0003-3522-7363), modified by Adrien Taudière

References

Nguyen NH, Song Z, Bates ST, Branco S, Tedersoo L, Menke J, Schilling JS, Kennedy PG. 2016. *FUNGuild: An open annotation tool for parsing fungal community datasets by ecological guild.* Fungal Ecology 20:241-248.

`ggaluv_pq` *Alluvial plot for taxonomy and samples factor vizualisation*

Description

Basically a wrapper of ggalluvial package

Usage

```
ggaluv_pq(
  physeq,
  taxa_ranks = c("Phylum", "Class", "Order", "Family"),
  wrap_factor = NULL,
  by_sample = FALSE,
  rarefy_by_sample = FALSE,
  fact = NULL,
  type = "nb_seq",
  width = 1.2,
  min.size = 3,
  na_remove = FALSE,
  use_ggfittext = FALSE,
  use_geom_label = FALSE,
  size_lab = 2,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`taxa_ranks`	A vector of taxonomic ranks. For examples c("Family","Genus"). If taxa ranks is not set (default value = c("Phylum", "Class", "Order", "Family")).
`wrap_factor`	A name to determine which samples to merge using `merge_samples2()` function. Need to be in `physeq@sam_data`. Need to be use when you want to wrap by factor the final plot with the number of taxa (type="nb_taxa")
`by_sample`	(logical) If FALSE (default), sample information is not taking into account, so the taxonomy is studied globally. If fact is not NULL, by_sample is automatically set to TRUE.
`rarefy_by_sample`	(logical, default FALSE) If TRUE, rarefy samples using `phyloseq::rarefy_even_depth()` function.
`fact`	(required) Name of the factor in `physeq@sam_data` used to plot the last column
`type`	If "nb_seq" (default), the number of sequences is used in plot. If "nb_taxa", the number of ASV is plotted.
`width`	(passed on to `ggalluvial::geom_flow()`) the width of each stratum, as a proportion of the distance between axes. Defaults to 1/3.
`min.size`	(passed on to `ggfittext::geom_fit_text()`) Minimum font size, in points. Text that would need to be shrunk below this size to fit the box will be hidden. Defaults to 4 pt.
`na_remove`	(logical, default FALSE) If set to TRUE, remove samples with NA in the variables set in formula.
`use_ggfittext`	(logical, default FALSE) Do we use ggfittext to plot labels?
`use_geom_label`	(logical, default FALSE) Do we use geom_label to plot labels?
`size_lab`	Size for label if use_ggfittext is FALSE
`...`	Additional arguments passed on to `ggalluvial::geom_flow()` function.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `ggalluvial` package if you use this function.

Value

A ggplot object

Author(s)

Adrien Taudière

See Also

`sankey_pq()`

Examples

```
if (requireNamespace("ggalluvial")) {
  ggaluv_pq(data_fungi_mini)
}

if (requireNamespace("ggalluvial")) {
  ggaluv_pq(data_fungi_mini, type = "nb_taxa")

  ggaluv_pq(data_fungi_mini, wrap_factor = "Height", by_sample = TRUE, type = "nb_taxa") +
    facet_wrap("Height")

  ggaluv_pq(data_fungi_mini,
    width = 0.9, min.size = 10,
    type = "nb_taxa", taxa_ranks = c("Phylum", "Class", "Order", "Family", "Genus")
  ) +
    coord_flip() + scale_x_discrete(limits = rev)
}
```

`ggbetween_pq` *Box/Violin plots for between-subjects comparisons of Hill Number*

Description

Note that contrary to `hill_pq()`, this function does not take into account for difference in the number of sequences per samples/modalities. You may use rarefy_by_sample = TRUE if the mean number of sequences per samples differs among modalities.

Basically a wrapper of function `ggstatsplot::ggbetweenstats()` for object of class phyloseq

Usage

```
ggbetween_pq(physeq, fact, one_plot = FALSE, rarefy_by_sample = FALSE, ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`fact`	(required): The variable to test. Must be present in the `sam_data` slot of the physeq object.
`one_plot`	(logical, default FALSE) If TRUE, return a unique plot with the three plot inside using the patchwork package.
`rarefy_by_sample`	(logical, default FALSE) If TRUE, rarefy samples using `phyloseq::rarefy_even_depth()` function
`...`	Additional arguments passed on to `ggstatsplot::ggbetweenstats()` function.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `ggstatsplot::ggbetweenstats()` if you use this function.

Value

Either an unique ggplot2 object (if one_plot is TRUE) or a list of 3 ggplot2 plot:

- plot_Hill_0 : the ggbetweenstats of Hill number 0 (= species richness) against the variable fact
- plot_Hill_1 : the ggbetweenstats of Hill number 1 (= Shannon index) against the variable fact
- plot_Hill_2 : the ggbetweenstats of Hill number 2 (= Simpson index) against the variable fact

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("ggstatsplot")) {
  p <- ggbetween_pq(data_fungi, fact = "Time", p.adjust.method = "BH")
  p[[1]]
  ggbetween_pq(data_fungi, fact = "Height", one_plot = TRUE)
  ggbetween_pq(data_fungi, fact = "Height", one_plot = TRUE, rarefy_by_sample = TRUE)
}
```

`ggscatt_pq` *Scatterplot with marginal distributions and statistical results against Hill diversity of phyloseq object*

Description

Basically a wrapper of function `ggstatsplot::ggscatterstats()` for object of class phyloseq and Hill number.

Usage

```
ggscatt_pq(
  physeq,
  num_modality,
  hill_scales = c(0, 1, 2),
  rarefy_by_sample = FALSE,
  one_plot = TRUE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`num_modality`	(required) Name of the numeric column in `physeq@sam_data` to plot and test against hill number
`hill_scales`	(a vector of integer) The list of q values to compute the hill number H^q. If Null, no hill number are computed. Default value compute the Hill number 0 (Species richness), the Hill number 1 (exponential of Shannon Index) and the Hill number 2 (inverse of Simpson Index).
`rarefy_by_sample`	(logical, default FALSE) If TRUE, rarefy samples using `phyloseq::rarefy_even_depth()` function.
`one_plot`	(logical, default FALSE) If TRUE, return a unique plot with the three plot inside using the patchwork package.
`...`	Additional arguments passed on to `ggstatsplot::ggscatterstats()` function.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `ggstatsplot::ggscatterstats()` if you use this function.

Value

Either an unique ggplot2 object (if one_plot is TRUE) or a list of ggplot2 plot for each hill_scales.

Author(s)

Adrien Taudière

See Also

`ggbetween_pq()`

Examples

```
if (requireNamespace("ggstatsplot")) {
  ggscatt_pq(data_fungi_mini, "Time", type = "non-parametric")
  ggscatt_pq(data_fungi_mini, "Time", hill_scales = 1:4, type = "parametric")
  ggscatt_pq(data_fungi_mini, "Sample_id",
    hill_scales = c(0, 0.5),
    one_plot = FALSE
  )
}
```

---

ggvenn_pq *Venn diagram of* phyloseq-class *object using* ggVennDiagram::ggVennDiagram *function*

Description

Note that you can use ggplot2 function to customize the plot for ex. `+ scale_fill_distiller(palette = "BuPu", direction = 1)` and `+ scale_x_continuous(expand = expansion(mult = 0.5))`. See examples.

Usage

```
ggvenn_pq(
  physeq = NULL,
  fact = NULL,
  min_nb_seq = 0,
  taxonomic_rank = NULL,
  split_by = NULL,
  add_nb_samples = TRUE,
  add_nb_seq = FALSE,
  rarefy_before_merging = FALSE,
  rarefy_after_merging = FALSE,
  return_data_for_venn = FALSE,
  verbose = TRUE,
  type = "nb_taxa",
  na_remove = TRUE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required): Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`.
`min_nb_seq`	minimum number of sequences by OTUs by samples to take into count this OTUs in this sample. For example, if min_nb_seq=2,each value of 2 or less in the OTU table will not count in the venn diagram
`taxonomic_rank`	Name (or number) of a taxonomic rank to count. If set to Null (the default) the number of OTUs is counted.
`split_by`	Split into multiple plot using variable split_by. The name of a variable must be present in `sam_data` slot of the physeq object.
`add_nb_samples`	(logical, default TRUE) Add the number of samples to levels names
`add_nb_seq`	(logical, default FALSE) Add the number of sequences to levels names
`rarefy_before_merging`	Rarefy each sample before merging by the modalities of args `fact`. Use `phyloseq::rarefy_even_depth` function
`rarefy_after_merging`	Rarefy each sample after merging by the modalities of args `fact`.
`return_data_for_venn`	(logical, default FALSE) If TRUE, the plot is not returned, but the resulting dataframe to plot with ggVennDiagram package is returned.
`verbose`	(logical, default TRUE) If TRUE, prompt some messages.
`type`	If "nb_taxa" (default), the number of taxa (ASV, OTU or taxonomic_rank if `taxonomic_rank` is not NULL) is used in plot. If "nb_seq", the number of sequences is plotted. `taxonomic_rank` is never used if type = "nb_seq".
`na_remove`	(logical, default TRUE) If set to TRUE, remove samples with NA in the variables set in `fact` param
`...`	Other arguments for the `ggVennDiagram::ggVennDiagram` function for ex. `category.names`.

Value

A `ggplot`2 plot representing Venn diagram of modalities of the argument `factor` or if split_by is set a list of plots.

Author(s)

Adrien Taudière

See Also

`upset_pq()`

Examples

```
if (requireNamespace("ggVennDiagram")) {
  ggvenn_pq(data_fungi, fact = "Height")
}

if (requireNamespace("ggVennDiagram")) {
  ggvenn_pq(data_fungi, fact = "Height") +
    ggplot2::scale_fill_distiller(palette = "BuPu", direction = 1)
  pl <- ggvenn_pq(data_fungi, fact = "Height", split_by = "Time")
  for (i in seq_along(pl)) {
    p <- pl[[i]] +
      scale_fill_distiller(palette = "BuPu", direction = 1) +
      theme(plot.title = element_text(hjust = 0.5, size = 22))
    print(p)
  }

  data_fungi2 <- subset_samples(data_fungi, data_fungi@sam_data$Tree_name == "A10-005" |
    data_fungi@sam_data$Height %in% c("Low", "High"))
  ggvenn_pq(data_fungi2, fact = "Height")

  ggvenn_pq(data_fungi2, fact = "Height", type = "nb_seq")

  ggvenn_pq(data_fungi, fact = "Height", add_nb_seq = TRUE, set_size = 4)
  ggvenn_pq(data_fungi, fact = "Height", rarefy_before_merging = TRUE)
  ggvenn_pq(data_fungi, fact = "Height", rarefy_after_merging = TRUE) +
    scale_x_continuous(expand = expansion(mult = 0.5))

  # For more flexibility, you can save the dataset for more precise construction
  # with ggplot2 and ggVennDiagramm
  # (https://gaospecial.github.io/ggVennDiagram/articles/fully-customed.html)
  res_venn <- ggvenn_pq(data_fungi, fact = "Height", return_data_for_venn = TRUE)

  ggplot() +
    # 1. region count layer
    geom_polygon(aes(X, Y, group = id, fill = name),
      data = ggVennDiagram::venn_regionedge(res_venn)
    ) +
    scale_fill_manual(values = funky_color(7)) +
    # 2. set edge layer
    geom_path(aes(X, Y, color = id, group = id),
      data = ggVennDiagram::venn_setedge(res_venn),
      show.legend = FALSE, linewidth = 2
    ) +
    scale_color_manual(values = c("red", "red", "blue")) +
    # 3. set label layer
    geom_text(aes(X, Y, label = name),
      data = ggVennDiagram::venn_setlabel(res_venn)
    ) +
    # 4. region label layer
    geom_label(
      aes(X, Y, label = paste0(
        count, " (",
```

```
        scales::percent(count / sum(count), accuracy = 2), ")"
      )),
      data = ggVennDiagram::venn_regionlabel(res_venn)
    ) +
    theme_void()
}
```

---

glmutli_pq *Automated model selection and multimodel inference with (G)LMs for phyloseq*

Description

See `glmulti::glmulti()` for more information.

Usage

```
glmutli_pq(
  physeq,
  formula,
  fitfunction = "lm",
  hill_scales = c(0, 1, 2),
  aic_step = 2,
  confsetsize = 100,
  plotty = FALSE,
  level = 1,
  method = "h",
  crit = "aicc",
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`formula`	(required) a formula for `glmulti::glmulti()` Variables must be present in the `physeq@sam_data` slot or be one of hill number defined in hill_scales or the variable Abundance which refer to the number of sequences per sample.
`fitfunction`	(default "lm")
`hill_scales`	(a vector of integer) The list of q values to compute the hill number H^q. If Null, no hill number are computed. Default value compute the Hill number 0 (Species richness), the Hill number 1 (exponential of Shannon Index) and the Hill number 2 (inverse of Simpson Index).
`aic_step`	The value between AIC scores to cut for.
`confsetsize`	The number of models to be looked for, i.e. the size of the returned confidence set.

`plotty`	(logical) Whether to plot the progress of the IC profile when running.
`level`	If 1, only main effects (terms of order 1) are used to build the candidate set. If 2, pairwise interactions are also used (higher order interactions are currently ignored)
`method`	The method to be used to explore the candidate set of models. If "h" (default) an exhaustive screening is undertaken. If "g" the genetic algorithm is employed (recommended for large candidate sets). If "l", a very fast exhaustive branch-and-bound algorithm is used. Package leaps must then be loaded, and this can only be applied to linear models with covariates and no interactions. If "d", a simple summary of the candidate set is printed, including the number of candidate models.
`crit`	The Information Criterion to be used. Default is the small-sample corrected AIC (aicc). This should be a function that accepts a fitted model as first argument. Other provided functions are the classic AIC, the Bayes IC (bic), and QAIC/QAICc (qaic and qaicc).
`...`	Additional arguments passed on to `glmulti::glmulti()` function

Details

This function is mainly a wrapper of the work of others. Please make a reference to `glmulti::glmulti()` if you use this function.

Value

A data.frame summarizing the glmulti results with columns

-estimates -unconditional_interval -nb_model" -importance -alpha

See Also

`glmulti::glmulti()`

Examples

```
if (requireNamespace(”glmulti”)) {
  res_glmulti <-
    glmutli_pq(data_fungi, ”Hill_0 ~ Hill_1 + Abundance + Time + Height”, level = 1)
  res_glmulti
  res_glmulti_interaction <-
    glmutli_pq(data_fungi, ”Hill_0 ~ Abundance + Time + Height”, level = 2)
  res_glmulti
}
```

`graph_test_pq` *Performs graph-based permutation tests on phyloseq object*

Description

A wrapper of `phyloseqGraphTest::graph_perm_test()` for quick plot with important statistics

Usage

```
graph_test_pq(
  physeq,
  fact,
  merge_sample_by = NULL,
  nperm = 999,
  return_plot = TRUE,
  title = "Graph Test",
  na_remove = FALSE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required) Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`. This should be a factor with two or more levels.
`merge_sample_by`	a vector to determine which samples to merge using `merge_samples2()` function. Need to be in `physeq@sam_data`
`nperm`	(int) The number of permutations to perform.
`return_plot`	(logical) Do we return only the result of the test, or do we plot the result?
`title`	The title of the Graph.
`na_remove`	(logical, default FALSE) If set to TRUE, remove samples with NA in the variables set in formula.
`...`	Other params for be passed on to `phyloseqGraphTest::graph_perm_test()` function

Details

This function is mainly a wrapper of the work of others. Please cite `phyloseqGraphTest` package.

Value

A `ggplot`2 plot with a subtitle indicating the pvalue and the number of permutations

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("phyloseqGraphTest")) {
  data(enterotype)
  graph_test_pq(enterotype, fact = "SeqTech")
  graph_test_pq(enterotype, fact = "Enterotype", na_remove = TRUE)
}
```

`hill_curves_pq` *Hill Diversities and Corresponding Accumulation Curves for phyloseq*

Description

Basically a wrapper of `vegan::renyi()` and `vegan::renyiaccum()` functions

Usage

```
hill_curves_pq(
  physeq,
  merge_sample_by = NULL,
  color_fac = NULL,
  hill_scales = c(0, 0.25, 0.5, 1, 2, 4, 8, 16, 32, 64, Inf),
  nperm = NULL,
  na_remove = TRUE,
  wrap_factor = TRUE,
  plot_legend = TRUE,
  linewidth = 2,
  size_point = 2,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`merge_sample_by`	a vector to determine which samples to merge using the `merge_samples2()` function. Need to be in physeq@sam_data
`color_fac`	(optional): The variable to color the barplot. For ex. same as fact. If merge_sample_by is set, color_fac must be nested in the merge_sample_by factor. See examples.
`hill_scales`	Scales of Rényi diversity.
`nperm`	(int Default NULL) If a integer is set to nperm, nperm permutation are computed to draw confidence interval for each curves. The function use `vegan::renyi()` if nperm is NULL and `vegan::renyiaccum()` else.

`na_remove`	(logical, default FALSE) If set to TRUE, remove samples with NA in the variables set in merge_sample_by. Not used if merge_sample_by is NULL.
`wrap_factor`	(logical, default TRUE) Do the plot is wrap by the factor
`plot_legend`	(logical, default TRUE) If set to FALSE, no legend are plotted.
`linewidth`	(int, default 2) The linewidth of lines.
`size_point`	(int, default 1) The size of the point.
`...`	Additional arguments passed on to `vegan::renyi()` function or `vegan::renyiaccum()` if nperm is not NULL.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `vegan::renyi()` or `vegan::renyiaccum()` functions

Value

A ggplot2 object

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("vegan")) {
  hill_curves_pq(data_fungi_mini, merge_sample_by = "Time")
  hill_curves_pq(data_fungi_mini, color_fac = "Time", plot_legend = FALSE)
  hill_curves_pq(data_fungi_mini,
    color_fac = "Time", plot_legend = FALSE,
    nperm = 9, size_point = 1, linewidth = 0.5
  )

  hill_curves_pq(data_fungi_mini,
    nperm = 9, plot_legend = FALSE, size_point = 1,
    linewidth = 0.5
  )
  hill_curves_pq(data_fungi_mini, "Height",
    hill_scales = c(0, 1, 2, 8), plot_legend = FALSE
  )
  hill_curves_pq(data_fungi_mini, "Height",
    hill_scales = c(0, 0.5, 1, 2, 4, 8),
    nperm = 9
  )
  hill_curves_pq(data_fungi_mini, "Height", nperm = 9, wrap_factor = FALSE)

  data_fungi_mini@sam_data$H_T <- paste0(
    data_fungi_mini@sam_data$Height,
    "_", data_fungi_mini@sam_data$Time
  )
  merge_samples2(data_fungi_mini, "H_T")
```

```
  hill_curves_pq(data_fungi_mini, "H_T", color_fac = "Time", nperm = 9)
}
```

---

hill_pq *Graphical representation of hill number 0, 1 and 2 across a factor*

Description

Hill numbers are the number of equiprobable species giving the same diversity value as the observed distribution. The Hill number 0 correspond to Species richness), the Hill number 1 to the exponential of Shannon Index and the Hill number 2 to the inverse of Simpson Index)

Note that (if correction_for_sample_size is TRUE, default behavior) this function use a sqrt of the read numbers in the linear model in order to correct for uneven sampling depth. This correction is only done before tuckey HSD plot and do not change the hill number computed.

Usage

```
hill_pq(
  physeq,
  fact = NULL,
  variable = NULL,
  hill_scales = c(0, 1, 2),
  color_fac = NA,
  letters = FALSE,
  add_points = FALSE,
  add_info = TRUE,
  kruskal_test = TRUE,
  one_plot = FALSE,
  plot_with_tuckey = TRUE,
  correction_for_sample_size = TRUE,
  na_remove = TRUE,
  vioplot = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required): The variable to test. Must be present in the `sam_data` slot of the physeq object.
`variable`	: Alias for factor. Kept only for backward compatibility.
`hill_scales`	(a vector of integer) The list of q values to compute the hill number H^q. If Null, no hill number are computed. Default value compute the Hill number 0 (Species richness), the Hill number 1 (exponential of Shannon Index) and the Hill number 2 (inverse of Simpson Index).

`color_fac`	(optional): The variable to color the barplot. For ex. same as fact. Not very useful because ggplot2 plot colors can be change using `scale_color_XXX()` function.
`letters`	(optional, default FALSE): If set to TRUE, the plot show letters based on p-values for comparison. Use the `multcompLetters` function from the package multcompLetters. BROKEN for the moment. Note that na values in The variable param need to be removed (see examples) to use letters.
`add_points`	(logical, default FALSE): add jitter point on boxplot
`add_info`	(logical, default TRUE) Do we add a subtitle with information about the number of samples per modality ?
`kruskal_test`	(logical, default TRUE) Do we test for global effect of our factor on each hill scales values? When kruskal_test is TRUE, the resulting test value are add in each plot in subtitle (unless add_info is FALSE). Moreover, if at least one hill scales is not significantly link to fact (pval>0.05), a message is prompt saying that Tuckey HSD plot is not informative for those Hill scales and letters are not printed.
`one_plot`	(logical, default FALSE) If TRUE, return a unique plot with the four plot inside using the patchwork package. Note that if letters and one_plot are both TRUE, tuckey HSD results are discarded from the unique plot. In that case, use one_plot = FALSE to see the tuckey HSD results in the fourth plot of the resulting list.
`plot_with_tuckey`	(logical, default TRUE). If one_plot is set to TRUE and letters to FALSE, allow to discard the tuckey plot part with plot_with_tuckey = FALSE
`correction_for_sample_size`	(logical, default TRUE) This function use a sqrt of the read numbers in the linear model in order to correct for uneven sampling depth in the Tuckey TEST. This params do not change value of Hill number but only the test associated values (including the pvalues). To rarefy samples, you may use the function `phyloseq::rarefy_even_depth()`.
`na_remove`	(logical, default TRUE) Do we remove samples with NA in the factor fact ? Note that na_remove is always TRUE when using letters = TRUE
`vioplot`	(logical, default FALSE) Do we plot violin plot instead of boxplot ?

Value

Either an unique ggplot2 object (if one_plot is TRUE) or a list of n+1 ggplot2 plot (with n the number of hill scale value). For example, with the default scale value:

- plot_Hill_0 : the boxplot of Hill number 0 (= species richness) against the variable
- plot_Hill_1 : the boxplot of Hill number 1 (= Shannon index) against the variable
- plot_Hill_2 : the boxplot of Hill number 2 (= Simpson index) against the variable
- plot_tuckey : plot the result of the Tuckey HSD test

Author(s)

Adrien Taudière

See Also

`psmelt_samples_pq()` and `ggbetween_pq()`

Examples

```
p <- hill_pq(data_fungi_mini, "Height", hill_scales = 1:2)
p_h1 <- p[[1]] + theme(legend.position = "none")
p_h2 <- p[[2]] + theme(legend.position = "none")
multiplot(plotlist = list(p_h1, p_h2, p[[3]]), cols = 4)

if (requireNamespace("multcompView")) {
  p2 <- hill_pq(data_fungi, "Time",
    correction_for_sample_size = FALSE,
    letters = TRUE, add_points = TRUE,
    plot_with_tuckey = FALSE
  )
  if (requireNamespace("patchwork")) {
    patchwork::wrap_plots(p2, guides = "collect")
  }
  # Artificially modify data_fungi to force alpha-diversity effect
 data_fungi_modif <- clean_pq(subset_samples_pq(data_fungi, !is.na(data_fungi@sam_data$Height)))
  data_fungi_modif@otu_table[data_fungi_modif@sam_data$Height == "High", ] <-
    data_fungi_modif@otu_table[data_fungi_modif@sam_data$Height == "High", ] +
   sample(c(rep(0, ntaxa(data_fungi_modif) / 2), rep(100, ntaxa(data_fungi_modif) / 2)))
  p3 <- hill_pq(data_fungi_modif, "Height",
    letters = TRUE, vioplot = TRUE,
    add_points = TRUE
  )
}
```

`hill_test_rarperm_pq` *Test multiple times effect of factor on Hill diversity with different rarefaction even depth*

Description

This reduce the risk of a random drawing of a exceptional situation of an unique rarefaction.

Usage

```
hill_test_rarperm_pq(
  physeq,
  fact,
  hill_scales = c(0, 1, 2),
  nperm = 99,
  sample.size = min(sample_sums(physeq)),
  verbose = FALSE,
  progress_bar = TRUE,
```

```
  p_val_signif = 0.05,
  type = "non-parametrique",
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required) Name of the factor in `physeq@sam_data` used to plot different lines
`hill_scales`	(a vector of integer) The list of q values to compute the hill number H^q. If Null, no hill number are computed. Default value compute the Hill number 0 (Species richness), the Hill number 1 (exponential of Shannon Index) and the Hill number 2 (inverse of Simpson Index).
`nperm`	(int) The number of permutations to perform.
`sample.size`	(int) A single integer value equal to the number of reads being simulated, also known as the depth. See `phyloseq::rarefy_even_depth()`.
`verbose`	(logical). If TRUE, print additional information.
`progress_bar`	(logical, default TRUE) Do we print progress during the calculation?
`p_val_signif`	(float, `[0:1]`) The mimimum value of p-value to count a test as significant int the `prop_signif` result.
`type`	A character specifying the type of statistical approach (See `ggstatsplot::ggbetweenstats()` for more details): • "parametric" • "nonparametric" • "robust" • "bayes"
`...`	Additional arguments passed on to `ggstatsplot::ggbetweenstats()` function

Value

A list of 6 components :

- method
- expressions
- plots
- pvals
- prop_signif
- statistics

Author(s)

Adrien Taudière

See Also

`ggstatsplot::ggbetweenstats()`, `hill_pq()`

Examples

```
if (requireNamespace("ggstatsplot")) {
  hill_test_rarperm_pq(data_fungi, "Time", nperm = 2)
  res <- hill_test_rarperm_pq(data_fungi, "Height", nperm = 9, p.val = 0.9)
  patchwork::wrap_plots(res$plots[[1]])
  res$plots[[1]][[1]] + res$plots[[2]][[1]] + res$plots[[3]][[1]]
  res$prop_signif
 res_para <- hill_test_rarperm_pq(data_fungi, "Height", nperm = 9, type = "parametrique")
  res_para$plots[[1]][[1]] + res_para$plots[[2]][[1]] + res_para$plots[[3]][[1]]
  res_para$pvals
  res_para$method
  res_para$expressions[[1]]
}
```

`hill_tuckey_pq` *Calculate hill number and compute Tuckey post-hoc test*

Description

Note that, by default, this function use a sqrt of the read numbers in the linear model in order to correct for uneven sampling depth.

Usage

```
hill_tuckey_pq(
  physeq,
  modality,
  hill_scales = c(0, 1, 2),
  silent = TRUE,
  correction_for_sample_size = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`modality`	(required) the variable to test
`hill_scales`	(a vector of integer) The list of q values to compute the hill number H^q. If Null, no hill number are computed. Default value compute the Hill number 0 (Species richness), the Hill number 1 (exponential of Shannon Index) and the Hill number 2 (inverse of Simpson Index).
`silent`	(logical) If TRUE, no message are printing.
`correction_for_sample_size`	(logical, default TRUE) This function use a sqrt of the read numbers in the linear model in order to correct for uneven sampling depth.

Value

A ggplot2 object

Author(s)

Adrien Taudière

Examples

```
data("GlobalPatterns", package = "phyloseq")
GlobalPatterns@sam_data[, "Soil_logical"] <-
  ifelse(GlobalPatterns@sam_data[, "SampleType"] == "Soil", "Soil", "Not Soil")
hill_tuckey_pq(GlobalPatterns, "Soil_logical")
hill_tuckey_pq(GlobalPatterns, "Soil_logical", hill_scales = 1:2)
```

`iNEXT_pq` *iNterpolation and EXTrapolation of Hill numbers (with iNEXT)*

Description

Note that this function is quite time-consuming due to high dimensionality in metabarcoding community matrix.

Usage

```
iNEXT_pq(physeq, merge_sample_by = NULL, ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`merge_sample_by`	(default: NULL) if not `NULL` samples of physeq are merged using the vector set by `merge_sample_by`. This merging used the `merge_samples2()`. In the case of `biplot_pq()` this must be a factor with two levels only.
`...`	Other arguments for the `iNEXT::iNEXT()` function

Value

see `iNEXT::iNEXT()` documentation

Author(s)

Adrien Taudière This function is mainly a wrapper of the work of others. Please make a reference to `iNEXT::iNEXT()` if you use this function.

Examples

```
if (requireNamespace("iNEXT")) {
  data("GlobalPatterns", package = "phyloseq")
  GPsubset <- subset_taxa(
    GlobalPatterns,
    GlobalPatterns@tax_table[, 1] == "Bacteria"
  )
  GPsubset <- subset_taxa(
    GPsubset,
    rowSums(GPsubset@otu_table) > 20000
  )
  GPsubset <- subset_taxa(
    GPsubset,
    rowSums(is.na(GPsubset@tax_table)) == 0
  )
  GPsubset@sam_data$human <- GPsubset@sam_data$SampleType %in%
    c("Skin", "Feces", "Tong")
  res_iNEXT <- iNEXT_pq(
    GPsubset,
    merge_sample_by = "human",
    q = 1,
    datatype = "abundance",
    nboot = 2
  )
  iNEXT::ggiNEXT(res_iNEXT)
  iNEXT::ggiNEXT(res_iNEXT, type = 2)
  iNEXT::ggiNEXT(res_iNEXT, type = 3)
}
```

`is_cutadapt_installed` *Test if cutadapt is installed.*

Description

Useful for testthat and examples compilation for R CMD CHECK and test coverage

Usage

```
is_cutadapt_installed(
  args_before_cutadapt =
   "source ~/miniconda3/etc/profile.d/conda.sh && conda activate cutadaptenv && "
)
```

Arguments

`args_before_cutadapt`	: (String) A one line bash command to run before to run cutadapt. For examples, "source ~/miniconda3/etc/profile.d/conda.sh && conda activate cutadaptenv &&" allow to bypass the conda init which asks to restart the shell

Value

A logical that say if cutadapt is install in

Author(s)

Adrien Taudière

Examples

```
MiscMetabar::is_cutadapt_installed()
```

`is_falco_installed` *Test if falco is installed.*

Description

Useful for testthat and examples compilation for R CMD CHECK and test coverage

Usage

```
is_falco_installed(path = "falco")
```

Arguments

`path`	(default: falco) Path to falco

Value

A logical that say if falco is install in

Author(s)

Adrien Taudière

Examples

```
MiscMetabar::is_falco_installed()
```

`is_krona_installed` *Test if krona is installed.*

Description

Useful for testthat and examples compilation for R CMD CHECK and test coverage

Usage

```
is_krona_installed(path = "ktImportKrona")
```

Arguments

`path`	(default: krona) Path to krona

Value

A logical that say if krona is install in

Author(s)

Adrien Taudière

Examples

```
MiscMetabar::is_krona_installed()
```

`is_mumu_installed` *Test if mumu is installed.*

Description

Useful for testthat and examples compilation for R CMD CHECK and test coverage

Usage

```
is_mumu_installed(path = "mumu")
```

Arguments

`path`	(default: mumu) Path to mumu

Value

A logical that say if mumu is install in

Author(s)

Adrien Taudière

Examples

```
MiscMetabar::is_mumu_installed()
```

`is_swarm_installed` *Test if swarm is installed.*

Description

Useful for testthat and examples compilation for R CMD CHECK and test coverage

Usage

```
is_swarm_installed(path = "swarm")
```

Arguments

`path`	(default: swarm) Path to falco

Value

A logical that say if swarm is install in

Author(s)

Adrien Taudière

Examples

```
MiscMetabar::is_swarm_installed()
```

`is_vsearch_installed` *Test if vsearch is installed.*

Description

Useful for testthat and examples compilation for R CMD CHECK and test coverage

Usage

```
is_vsearch_installed(path = "vsearch")
```

Arguments

`path`	(default: vsearch) Path to vsearch

Value

A logical that say if vsearch is install in

Author(s)

Adrien Taudière

Examples

```
MiscMetabar::is_vsearch_installed()
```

`krona` *Make Krona files using R hrefhttps://github.com/marbl/Krona/wikiKronaTools.*

Description

Need the installation of kronatools on the computer (installation instruction).

Usage

```
krona(
  physeq,
  file = "krona.html",
  nb_seq = TRUE,
  ranks = "All",
  add_unassigned_rank = 0,
  name = NULL
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`file`	(required) the location of the html file to save
`nb_seq`	(logical) If true, Krona set the distribution of sequences in the taxonomy. If False, Krona set the distribution of ASVs in the taxonomy.
`ranks`	Number of the taxonomic ranks to plot (num of the column in `tax_table` slot of your `physeq` object). Default setting plot all the ranks (argument 'All').
`add_unassigned_rank`	(int) Add unassigned for rank inferior to 'add_unassigned_rank' when necessary.
`name`	A name for intermediary files, Useful to name your krona result files before merging using `merge_krona()`. Must not contain space.

Details

This function is mainly a wrapper of the work of others. Please cite Krona if you use this function.

Value

A html file

Author(s)

Adrien Taudière

See Also

`merge_krona`

Examples

```
data("GlobalPatterns", package = "phyloseq")
GA <- subset_taxa(GlobalPatterns, Phylum == "Acidobacteria")
## Not run:
krona(GA, "Number.of.sequences.html")
krona(GA, "Number.of.ASVs.html", nb_seq = FALSE)
merge_krona(c("Number.of.sequences.html", "Number.of.ASVs.html"))

## End(Not run)
```

`LCBD_pq` *Compute and test local contributions to beta diversity (LCBD) of samples*

Description

A wrapper for the `adespatial::beta.div()` function in the case of `physeq` object.

Usage

```
LCBD_pq(physeq, p_adjust_method = "BH", ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`p_adjust_method`	(chr, default "BH"): the method used to adjust p-value
`...`	Additional arguments passed on to `adespatial::beta.div()` function

Value

An object of class `beta.div` see `adespatial::beta.div()` function for more information

Author(s)

Adrien Taudière This function is mainly a wrapper of the work of others. Please make a reference to `adespatial::beta.div()` if you use this function.

See Also

plot_LCBD_pq, `adespatial::beta.div()`

Examples

```
if (requireNamespace("adespatial")) {
  res <- LCBD_pq(data_fungi_sp_known, nperm = 5)
  str(res)
  length(res$LCBD)
  length(res$SCBD)
}

if (requireNamespace("adespatial")) {
  LCBD_pq(data_fungi_sp_known, nperm = 5, method = "jaccard")
}
```

`learn_idtaxa` *A wrapper of* `DECIPHER::LearnTaxa()`

Description

This function is basically a wrapper of functions `DECIPHER::LearnTaxa()`, please cite the DECIPHER package if you use this function.

Usage

```
learn_idtaxa(
  fasta_for_training,
  output_Rdata = NULL,
  output_path_only = FALSE,
  unite = FALSE,
  ...
)
```

Arguments

`fasta_for_training`	A fasta file (can be gzip) to train the trainingSet using the function `learn_idtaxa()`. Only used if trainingSet is NULL. The reference database must contain taxonomic information in the header of each sequence in the form of a string starting with ";tax=" and followed by a comma-separated list of up to nine taxonomic identifiers. The only exception is if `unite=TRUE`. In that case the UNITE taxonomy is automatically formatted.
`output_Rdata`	A vector naming the path to an output Rdata file. If left to NULL, no Rdata file is written.
`output_path_only`	(logical, default FALSE). If TRUE, the function return only the path to the output_Rdata file. Note that output_Rdata must be set.
`unite`	(logical, default FALSE). If set to TRUE, the fasta_for_training file is formatted from UNITE format to sintax one, needed in fasta_for_training. Only used if trainingSet is NULL.
`...`	Additional arguments passed on to `DECIPHER::LearnTaxa()`

Details

This function is mainly a wrapper of the work of others. Please make a reference to `DECIPHER::LearnTaxa()` if you use this function.

Value

Either a Taxa Train object (see `DECIPHER::LearnTaxa()`) or, if output_path_only is TRUE, a vector indicating the path to the output training object.

Author(s)

Adrien Taudière

See Also

`assign_idtaxa()`

Examples

```
## Not run:
training_mini_UNITE_fungi <-
  learn_idtaxa(fasta_for_training = system.file("extdata",
    "mini_UNITE_fungi.fasta.gz",
    package = "MiscMetabar"
  ))
plot(training_mini_UNITE_fungi)

training_100sp_UNITE <-
  learn_idtaxa(
    fasta_for_training = system.file("extdata",
      "100_sp_UNITE_sh_general_release_dynamic.fasta",
      package = "MiscMetabar"
    ),
    unite = TRUE
  )

plot(training_100sp_UNITE)

## End(Not run)
```

`list_fastq_files` *List fastq files*

Description

Useful for targets bioinformatic pipeline.

Usage

```
list_fastq_files(
  path,
  paired_end = TRUE,
  pattern = "fastq",
  pattern_R1 = "_R1_",
  pattern_R2 = "_R2_",
  nb_files = Inf
)
```

Arguments

`path`	path to files (required)
`paired_end`	do you have paired_end files? (default TRUE)
`pattern`	a pattern to filter files (passed on to list.files function).
`pattern_R1`	a pattern to filter R1 files (default "*R1*")
`pattern_R2`	a pattern to filter R2 files (default "*R2*")
`nb_files`	the number of fastq files to list (default FALSE)

Value

a list of one (single end) or two (paired end) list of files files are sorted by names (default behavior of `list.files()`)

Author(s)

Adrien Taudière

Examples

```
list_fastq_files(system.file("extdata", package = "MiscMetabar"))
list_fastq_files(system.file("extdata", package = "MiscMetabar"),
  paired_end = FALSE, pattern_R1 = ""
)
```

`lulu` *Post Clustering Curation of Amplicon Data.*

Description

The original function and documentation was written by Tobias Guldberg Frøslev in the lulu package.

This algorithm `lulu` consumes an OTU table and a matchlist, and evaluates cooccurence of 'daughters' (potential analytical artefacts) and their 'parents' (~= real biological species/OTUs). The algorithm requires an OTU table (species/site matrix), and a match list. The OTU table can be made with various r-packages (e.g. `DADA2`) or external pipelines (`VSEARCH`, `USEARCH`, `QIIME`, etc.), and the match-list can be made with external bioinformatic tools like `VSEARCH`, `USEARCH`, `BLASTN` or another algorithm for pair-wise sequence matching.

Usage

```
lulu(
  otu_table,
  matchlist,
  minimum_ratio_type = "min",
  minimum_ratio = 1,
  minimum_match = 84,
  minimum_relative_cooccurence = 0.95,
  progress_bar = TRUE,
  log_conserved = FALSE
)
```

Arguments

`otu_table`	a data.frame with with an OTU table that has sites/samples as columns and OTUs (unique OTU id's) as rows, and observations as read counts.
`matchlist`	a data.frame containing three columns: (1) OTU id of potential child, (2) OTU id of potential parent, (3) match - % identiti between the sequences of the potential parent and potential child OTUs. **NB: The matchlist is the product of a mapping of OTU sequences against each other. This is currently carried out by an external script in e.g. Blastn or VSEARCH, prior to running lulu!**
`minimum_ratio_type`	sets whether a potential error must have lower abundance than the parent in all samples `min` (default), or if an error just needs to have lower abundance on average `avg`. Choosing lower abundance on average over globally lower abundance will greatly increase the number of designated errors. This option was introduced to make it possible to account for non-sufficiently clustered intraspecific variation, but is not generally recommended, as it will also increase the potential of cluster well-separated, but co-occuring, sequence similar species.
`minimum_ratio`	sets the minimim abundance ratio between a potential error and a potential parent to be identified as an error. If the `minimum_ratio_type` is set to `min` (default), the `minimum_ratio` applies to the lowest observed ration across the samples. If the `minimum_ratio_type` is set to `avg` (default), the `minimum_ratio` applies to the mean of observed ration across the samples.`avg`. (default is 1).
`minimum_match`	minimum threshold of sequence similarity for considering any OTU as an error of another can be set (default 84%).
`minimum_relative_cooccurence`	minimum co-occurrence rate, i.e. the lower rate of occurrence of the potential error explained by co-occurrence with the potential parent for considering error state.
`progress_bar`	(Logical, default TRUE) print progress during the calculation or not.
`log_conserved`	(Logical, default FALSE) conserved log files writed in the disk

Details

Please cite the lulu original paper: https://www.nature.com/articles/s41467-017-01312-x

Value

Function `lulu` returns a list of results based on the input OTU table and match list.

- `curated_table` - a curated OTU table with daughters merged with their matching parents.
- `curated_count` - number of curated (parent) OTUs.
- `curated_otus` - ids of the OTUs that were accepted as valid OTUs.
- `discarded_count` - number of discarded (merged with parent) OTUs.
- `discarded_otus` - ids of the OTUs that were identified as errors (daughters) and merged with respective parents.
- `runtime` - time used by the script.
- `minimum_match` - the id threshold (minimum match \ by user).
- `minimum_relative_cooccurence` - minimum ratio of daughter-occurences explained by co-occurence with parent (set by user).
- `otu_map` - information of which daughters were mapped to which parents.
- `original_table` - original OTU table.

The matchlist is the product of a mapping of OTU sequences against each other. This is currently carried out by an external script in e.g. BLASTN or VSEARCH, prior to running `lulu`! Producing the match list requires a file with all the OTU sequences (centroids) - e.g. `OTUcentroids.fasta`. The matchlist can be produced by mapping all OTUs against each other with an external algorithm like VSEARCH or BLASTN. In `VSEARCH` a matchlist can be produced e.g. with the following command: `vsearch --usearch_global OTUcentroids.fasta --db OTUcentroids.fasta --strand plus --self --id .80 --iddef 1 --userout matchlist.txt --userfields query+target+id --maxaccepts 0 --query_cov .9 --maxhits 10`. In `BLASTN` a matchlist can be produces e.g. with the following commands. First we produce a blast-database from the fasta file: `makeblastdb -in OTUcentroids.fasta -parse_seqids -dbtype nucl`, then we match the centroids against that database: `blastn -db OTUcentoids.fasta -num_threads 10 -outfmt'6 qseqid sseqid pident' -out matchlist.txt -qcov_hsp_perc .90 -perc_identity .84 -query OTUcentroids.fasta`

Author(s)

Tobias Guldberg Frøslev (orcid: 0000-0002-3530-013X), modified by Adrien Taudière

---

`lulu_pq` *Lulu reclustering of class* `physeq`

Description

See https://www.nature.com/articles/s41467-017-01312-x for more information on the method.

Usage

```
lulu_pq(
  physeq,
  nproc = 1,
  id = 0.84,
  vsearchpath = "vsearch",
  verbose = FALSE,
  clean_pq = FALSE,
  keep_temporary_files = FALSE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`nproc`	(default 1) Set to number of cpus/processors to use for the clustering
`id`	(default: 0.84) id for –usearch_global.
`vsearchpath`	(default: vsearch) path to vsearch.
`verbose`	(logical) If true, print some additional messages.
`clean_pq`	(logical) If true, empty samples and empty ASV are discarded before clustering.
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files
`...`	Additional arguments passed on to function `lulu()`

Details

The version of LULU is a fork of Adrien Taudière (`https://github.com/adrientaudiere/lulu`) from `https://github.com/tobiasgf/lulu`

Value

a list of for object

- "new_physeq": The new phyloseq object (class physeq)
- "discrepancy_vector": A vector of discrepancy showing for each taxonomic level the proportion of identic value before and after lulu reclustering. A value of 0.6 stands for 60% of ASV before re-clustering have identical value after re-clustering. In other word, 40% of ASV are assigned to a different taxonomic value. NA value are not counted as discrepancy.
- "res_lulu": A list of the result from the lulu function
- "merged_ASV": the data.frame used to merged ASV

Author(s)

Tobias Guldberg Frøslev <tobiasgf@snm.ku.dk> & Adrien Taudière <adrien.taudiere@zaclys.net>

References

- LULU : https://github.com/adrientaudiere/lulu forked from https://github.com/tobiasgf/lulu.
- VSEARCH can be downloaded from https://github.com/torognes/vsearch.

See Also

`mumu_pq()`

Examples

```
lulu_pq(data_fungi_sp_known)
```

`merge_krona` *Merge Krona files using* R*hrefhttps://github.com/marbl/Krona/wikiKronaTools.*

Description

Need the installation of kronatools on the computer (installation instruction).

Function merge_krona allows merging multiple html files in one interactive krona file

Note that you need to use the name args in `krona()` function before `merge_krona()` in order to give good name to each krona pie in the output.

Usage

```
merge_krona(files = NULL, output = "mergeKrona.html")
```

Arguments

`files`	(required) path to html files to merged
`output`	path to the output file

Details

This function is mainly a wrapper of the work of others. Please cite Krona if you use this function.

Value

A html file

Author(s)

Adrien Taudière

See Also

`krona`

Examples

```
## Not run:
data("GlobalPatterns", package = "phyloseq")
GA <- subset_taxa(GlobalPatterns, Phylum == "Acidobacteria")
krona(GA, "Number.of.sequences.html", name = "Nb_seq_GP_acidobacteria")
krona(GA, "Number.of.ASVs.html", nb_seq = FALSE, name = "Nb_asv_GP_acidobacteria")
merge_krona(c("Number.of.sequences.html", "Number.of.ASVs.html"), "mergeKrona.html")
unlink(c("Number.of.sequences.html", "Number.of.ASVs.html", "mergeKrona.html"))

## End(Not run)
```

`merge_samples2` *Merge samples by a sample variable or factor*

Description

Firstly release in the speedyseq R package by Michael R. McLaren.

This function provides an alternative to `phyloseq::merge_samples()` that better handles sample variables of different types, especially categorical sample variables. It combines the samples in `x` defined by the sample variable or factor `group` by summing the abundances in `otu_table(x)` and combines sample variables by the summary functions in `funs`. The default summary function, `unique_or_na()`, collapses the values within a group to a single unique value if it exists and otherwise returns NA. The new (merged) samples are named by the values in `group`.

Usage

```
merge_samples2(
  x,
  group,
  fun_otu = sum,
  funs = list(),
  reorder = FALSE,
  default_fun = unique_or_na
)

## S4 method for signature 'phyloseq'
merge_samples2(
  x,
  group,
  fun_otu = sum,
  funs = list(),
  reorder = FALSE,
```

```
  default_fun = unique_or_na
)

## S4 method for signature 'otu_table'
merge_samples2(
  x,
  group,
  fun_otu = sum,
  reorder = FALSE,
  default_fun = unique_or_na
)

## S4 method for signature 'sample_data'
merge_samples2(
  x,
  group,
  funs = list(),
  reorder = FALSE,
  default_fun = unique_or_na
)
```

Arguments

`x`	A phyloseq, `otu_table`, or `sample_data` object
`group`	A sample variable or a vector of length `nsamples(x)` defining the sample grouping. A vector must be supplied if x is an otu_table
`fun_otu`	Function for combining abundances in the otu_table; default is `sum`. Can be a formula to be converted to a function by `purrr::as_mapper()`
`funs`	Named list of merge functions for sample variables; default is `unique_or_na`
`reorder`	Logical specifying whether to reorder the new (merged) samples by name
`default_fun`	Default functions if funs is not set. Per default the function unique_or_na is used. See `diff_fct_diff_class()` for a useful alternative.

Value

A new phyloseq-class, otu_table or sam_data object depending on the class of the x param

Author(s)

Michael R. McLaren (orcid: 0000-0003-1575-473X) modified by Adrien Taudiere

Examples

```
data(enterotype)

# Merge samples with the same project and clinical status
ps <- enterotype
sample_data(ps) <- sample_data(ps) %>%
```

```
  transform(Project.ClinicalStatus = Project:ClinicalStatus)
sample_data(ps) %>% head()
ps0 <- merge_samples2(ps, "Project.ClinicalStatus",
  fun_otu = mean,
  funs = list(Age = mean)
)
sample_data(ps0) %>% head()
```

---

merge_taxa_vec *Merge taxa in groups (vectorized version)*

Description

Firstly release in the speedyseq R package by Michael R. McLaren.

Merge taxa in `x` into a smaller set of taxa defined by the vector `group`. Taxa whose value in `group` is NA will be dropped. New taxa will be named according to the most abundant taxon in each group (`phyloseq` and `otu_table` objects) or the first taxon in each group (all other phyloseq component objects).

If `x` is a phyloseq object with a phylogenetic tree, then the new taxa will be ordered as they are in the tree. Otherwise, the taxa order can be controlled by the `reorder` argument, which behaves like the `reorder` argument in `base::rowsum()`. `reorder = FALSE` will keep taxa in the original order determined by when the member of each group first appears in `taxa_names(x)`; `reorder = TRUE` will order new taxa according to their corresponding value in `group`.

The `tax_adjust` argument controls the handling of taxonomic disagreements within groups. Setting `tax_adjust == 0` causes no adjustment; the taxonomy of the new group is set to the archetype taxon (see below). Otherwise, disagreements within a group at a given rank cause the values at lower ranks to be set to `NA`. If `tax_adjust == 1` (the default), then a rank where all taxa in the group are already NA is not counted as a disagreement, and lower ranks may be kept if the taxa agree. This corresponds to the original phyloseq behavior. If `tax_adjust == 2`, then these NAs are treated as a disagreement; all ranks are set to NA after the first disagreement or NA.

Usage

```
merge_taxa_vec(x, group, reorder = FALSE, tax_adjust = 1L)

## S4 method for signature 'phyloseq'
merge_taxa_vec(x, group, reorder = FALSE, tax_adjust = 1L)

## S4 method for signature 'otu_table'
merge_taxa_vec(x, group, reorder = FALSE)

## S4 method for signature 'taxonomyTable'
merge_taxa_vec(x, group, reorder = FALSE, tax_adjust = 1L)

## S4 method for signature 'phylo'
merge_taxa_vec(x, group)
```

```
## S4 method for signature 'XStringSet'
merge_taxa_vec(x, group, reorder = FALSE)
```

Arguments

`x`	A phyloseq object or component object
`group`	A vector with one element for each taxon in physeq that defines the new groups. see `base::rowsum()`.
`reorder`	Logical specifying whether to reorder the taxa by their `group` values. Ignored if x has (or is) a phylogenetic tree.
`tax_adjust`	0: no adjustment; 1: phyloseq-compatible adjustment; 2: conservative adjustment

Value

A new phyloseq-class, otu_table, tax_table, XStringset or sam_data object depending on the class of the x param

Author(s)

Michael R. McLaren (orcid: 0000-0003-1575-473X) modified by Adrien Taudiere

See Also

Function in MiscMetabar that use this function: `postcluster_pq()`

`base::rowsum()`

`phyloseq::merge_taxa()`

`MiscMetabar-deprecated` *Deprecated function(s) in the MiscMetabar package*

Description

These functions are provided for compatibility with older version of the MiscMetabar package. They may eventually be completely removed.

Usage

```
physeq_graph_test(...)
```

Arguments

`...`	Parameters to be passed on to the modern version of the function

Value

Depend on the functions.

Details

graph_test_pq	now a synonym for `physeq_graph_test`
adonis_pq	now a synonym for `adonis_phyloseq`
clean_pq	now a synonym for `clean_physeq`
lulu_pq	now a synonym for `lulu_phyloseq`
circle_pq	now a synonym for `otu_circle`
biplot_pq	now a synonym for `biplot_physeq`
read_pq	now a synonym for `read_phyloseq`
write_pq	now a synonym for `write_phyloseq`
sankey_pq	now a synonym for `sankey_phyloseq`
summary_plot_pq	now a synonym for `summary_plot_phyloseq`
plot_edgeR_pq	now a synonym for `plot_edgeR_phyloseq`
plot_deseq2_pq	now a synonym for `plot_deseq2_phyloseq`
venn_pq	now a synonym for `venn_phyloseq`
ggvenn_pq	now a synonym for `ggVenn_phyloseq`
hill_tuckey_pq	now a synonym for `hill_tuckey_phyloseq`
hill_pq	now a synonym for `hill_phyloseq`
heat_tree_pq	now a synonym for `physeq_heat_tree`
compare_pairs_pq	now a synonym for `multiple_share_bisamples`

`multipatt_pq` *Test and plot multipatt result*

Description

A wrapper for the `indicspecies::multipatt()` function in the case of `physeq` object.

Usage

```
multipatt_pq(
  physeq,
  fact,
  p_adjust_method = "BH",
  pval = 0.05,
  control = permute::how(nperm = 999),
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required) Name of the factor in `physeq@sam_data` used to plot different lines
`p_adjust_method`	(chr, default "BH"): the method used to adjust p-value
`pval`	(int, default 0.05): the value to determine the significance of LCBD
`control`	see `?indicspecies::multipatt()`
`...`	Additional arguments passed on to `indicspecies::multipatt()` function

Details

This function is mainly a wrapper of the work of others. Please make a reference to `indicspecies::multipatt()` if you use this function.

Value

A ggplot2 object

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("indicspecies")) {
  data(data_fungi)
  data_fungi_ab <- subset_taxa_pq(data_fungi, taxa_sums(data_fungi) > 10000)
  multipatt_pq(subset_samples(data_fungi_ab, !is.na(Time)), fact = "Time")
}

if (requireNamespace("indicspecies")) {
  multipatt_pq(subset_samples(data_fungi_ab, !is.na(Time)),
    fact = "Time",
    max.order = 1, control = permute::how(nperm = 99)
  )
}
```

`multiplot` *Multiple plot function*

Description

ggplot objects can be passed in ..., or to plotlist (as a list of ggplot objects)

If the layout is something like matrix(c(1,2,3,3), nrow=2, byrow=TRUE), then plot 1 will go in the upper left, 2 will go in the upper right, and 3 will go all the way across the bottom.

Usage

```
multiplot(..., plotlist = NULL, cols = 1, layout = NULL)
```

Arguments

`...`	list of ggplot objects
`plotlist`	list of ggplot objects
`cols`	number of columns
`layout`	A matrix specifying the layout. If present, 'cols' is ignored.

Value

Nothing. Print the list of ggplot objects

`multitax_bar_pq` *Plot taxonomic distribution across 3 taxonomic levels and optionally one sample factor*

Description

Note that lvl3 need to be nested in lvl2 which need to be nested in lvl1

Usage

```
multitax_bar_pq(
  physeq,
  lvl1,
  lvl2,
  lvl3,
  fact = NULL,
  nb_seq = TRUE,
  log10trans = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`lvl1`	(required) Name of the first (higher) taxonomic rank of interest
`lvl2`	(required) Name of the second (middle) taxonomic rank of interest
`lvl3`	(required) Name of the first (lower) taxonomic rank of interest
`fact`	Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`. If not set, the taxonomic distribution is plot for all samples together.
`nb_seq`	(logical; default TRUE) If set to FALSE, only the number of ASV is count. Concretely, physeq otu_table is transformed in a binary otu_table (each value different from zero is set to one)
`log10trans`	(logical, default TRUE) If TRUE, the number of sequences (or ASV if nb_seq = FALSE) is log10 transformed.

Value

A ggplot2 object

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("ggh4x")) {
  multitax_bar_pq(data_fungi_sp_known, "Phylum", "Class", "Order", "Time")
  multitax_bar_pq(data_fungi_sp_known, "Phylum", "Class", "Order")
  multitax_bar_pq(data_fungi_sp_known, "Phylum", "Class", "Order",
    nb_seq = FALSE, log10trans = FALSE
  )
}
```

`multi_biplot_pq` *Visualization of a collection of couples of samples for comparison*

Description

This allow to plot all the possible `biplot_pq()` combination using one factor.

Usage

```
multi_biplot_pq(physeq, split_by = NULL, pairs = NULL, na_remove = TRUE, ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`split_by`	(required if pairs is NULL) the name of the factor to make all combination of couples of values
`pairs`	(required if split_by is NULL) the name of the factor in physeq@sam_data‘ slot to make plot by pairs of samples. Each level must be present only two times. Note that if you set pairs, you also must set fact arguments to passed on to `biplot_pq()`.
`na_remove`	(logical, default TRUE) if TRUE remove all the samples with NA in the `split_by` variable of the `physeq@sam_data` slot
`...`	Other parameters passed on to `biplot_pq()`

Value

a list of ggplot object

Author(s)

Adrien Taudière

Examples

```
data_fungi_abun <- subset_taxa_pq(data_fungi, taxa_sums(data_fungi) > 10000)
p <- multi_biplot_pq(data_fungi_abun, "Height")
lapply(p, print)
```

`mumu_pq` *MUMU reclustering of class* `physeq`

Description

See https://www.nature.com/articles/s41467-017-01312-x for more information on the original method LULU. This is a wrapper of mumu a C++ re-implementation of LULU by Frédéric Mahé

Usage

```
mumu_pq(
  physeq,
  nproc = 1,
  id = 0.84,
  vsearchpath = "vsearch",
  mumupath = "mumu",
  verbose = FALSE,
  clean_pq = TRUE,
  keep_temporary_files = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`nproc`	(default 1) Set to number of cpus/processors to use for the clustering
`id`	(default: 0.84) id for –usearch_global.
`vsearchpath`	(default: vsearch) path to vsearch.
`mumupath`	path to mumu. See mumu for installation instruction
`verbose`	(logical) If true, print some additional messages.
`clean_pq`	(logical) If true, empty samples and empty ASV are discarded before clustering.
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files

Details

This function is mainly a wrapper of the work of others. Please cite mumu and lulu if you use this function for your work.

Value

a list of for object

- "new_physeq": The new phyloseq object (class physeq)
- "mumu_results": The log file of the mumu software. Run `man mumu` into bash to obtain details about columns' signification.

Author(s)

Frédéric Mahé & Adrien Taudière <`adrien.taudiere@zaclys.net`>

References

- MUMU: `https://github.com/frederic-mahe/mumu`
- VSEARCH can be downloaded from `https://github.com/torognes/vsearch`.

See Also

`lulu_pq()`

Examples

```
## Not run:
mumu_pq(data_fungi_sp_known)

## End(Not run)
```

`normalize_prop_pq` *Normalize OTU table using samples depth*

Description

This function implement the method proposed by McKnight et al. 2018 (doi:10.5061/dryad.tn8qs35)

Usage

```
normalize_prop_pq(physeq, base_log = 2, constante = 10000, digits = 4)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`base_log`	(integer, default 2) the base for log-transformation. If set to NULL or NA, no log-transformation is compute after normalization.
`constante`	a constante to multiply the otu_table values
`digits`	(default = 2) integer indicating the number of decimal places to be used (see `?round` for more information)

Value

A new `phyloseq-class` object with otu_table count normalize and log transformed (if base_log is an integer)

Author(s)

Adrien Taudière

Examples

```
taxa_sums(data_fungi_mini)
data_f_norm <- normalize_prop_pq(data_fungi_mini)
taxa_sums(data_f_norm)
ggplot(data.frame(
  "norm" = scale(taxa_sums(data_f_norm)),
  "raw" = scale(taxa_sums(data_fungi_mini)),
  "name_otu" = taxa_names(data_f_norm)
)) +
  geom_point(aes(x = raw, y = norm))

data_f_norm <- normalize_prop_pq(data_fungi_mini, base_log = NULL)
```

`no_legend` *Discard legend in ggplot2*

Description

A more memorable shortcut for theme(legend.position = "none").

Usage

```
no_legend()
```

Value

A ggplot2 object

Author(s)

Adrien Taudière

Examples

```
plot_refseq_pq(data_fungi)
plot_refseq_pq(data_fungi) + no_legend()
```

`perc` *Convert a value (or a fraction x/y) in percentage*

Description

Mostly for internal use.

Usage

```
perc(x, y = NULL, accuracy = 0, add_symbol = FALSE)
```

Arguments

`x`	(required): value
`y`	if y is set, compute the division of x by y
`accuracy`	number of digits (number of digits after zero)
`add_symbol`	if set to TRUE add the % symbol to the value

Value

The percentage value (number or character if add_symbol is set to TRUE)

Author(s)

Adrien Taudière

`phyloseq_to_edgeR` *Convert phyloseq OTU count data into DGEList for edgeR package*

Description

Convert phyloseq OTU count data into DGEList for edgeR package

Usage

```
phyloseq_to_edgeR(physeq, group, method = "RLE", ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`group`	(required) A character vector or factor giving the experimental group/condition for each sample/library. Alternatively, you may provide the name of a sample variable. This name should be among the output of `sample_variables(physeq)`, in which case `get_variable(physeq, group)` would return either a character vector or factor. This is passed on to `DGEList`, and you may find further details or examples in its documentation.
`method`	The label of the edgeR-implemented normalization to use. See `calcNormFactors` for supported options and details. The default option is `"RLE"`, which is a scaling factor method proposed by Anders and Huber (2010). At time of writing, the edgeR package supported the following options to the `method` argument: `c("TMM", "RLE", "upperquartile", "none")`.
`...`	Additional arguments passed on to `DGEList`

Value

A DGEList object. See `edgeR::estimateTagwiseDisp()` for more details.

`physeq_or_string_to_dna` *Return a DNAStringSet object from either a character vector of DNA sequences or the* `refseq` *slot of a phyloseq-class object*

Description

Internally used in `vsearch_clustering()`, `swarm_clustering()` and `postcluster_pq()`.

Usage

```
physeq_or_string_to_dna(physeq = NULL, dna_seq = NULL)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`dna_seq`	You may directly use a character vector of DNA sequences in place of physeq args. When physeq is set, dna sequences take the value of `physeq@refseq`

Value

An object of class DNAStringSet (see the `Biostrings::DNAStringSet()` function)

Author(s)

Adrien Taudière

See Also

`Biostrings::DNAStringSet()`

Examples

```
dna <- physeq_or_string_to_dna(data_fungi)
dna

sequences_ex <- c(
  "TACCTATGTTGCCTTGGCGGCTAAACCTACCCGGGATTTGATGGGGCGAATTAATAACGAATTCATTGAATCA",
  "TACCTATGTTGCCTTGGCGGCTAAACCTACCCGGGATTTGATGGGGCGAATTACCTGGTAAGGCCCACTT",
  "TACCTATGTTGCCTTGGCGGCTAAACCTACCCGGGATTTGATGGGGCGAATTACCTGGTAGAGGTG",
  "TACCTATGTTGCCTTGGCGGCTAAACCTACC",
  "CGGGATTTGATGGCGAATTACCTGGTATTTTAGCCCACTTACCCGGTACCATGAGGTG",
  "GCGGCTAAACCTACCCGGGATTTGATGGCGAATTACCTGG",
  "GCGGCTAAACCTACCCGGGATTTGATGGCGAATTACAAAG",
  "GCGGCTAAACCTACCCGGGATTTGATGGCGAATTACAAAG",
  "GCGGCTAAACCTACCCGGGATTTGATGGCGAATTACAAAG"
)
dna2 <- physeq_or_string_to_dna(dna_seq = sequences_ex)
dna2
```

`plot_ancombc_pq` *Plot ANCOMBC2 result for phyloseq object*

Description

Graphical representation of ANCOMBC2 result.

Usage

```
plot_ancombc_pq(
  physeq,
  ancombc_res,
  filter_passed = TRUE,
  filter_diff = TRUE,
  min_abs_lfc = 0,
  tax_col = "Genus",
  tax_label = "Species",
  add_marginal_vioplot = TRUE,
  add_label = TRUE,
  add_hline_cut_lfc = NULL
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`ancombc_res`	(required) the result of the ancombc_pq function For the moment only bimodal factors are possible.
`filter_passed`	(logical, default TRUE) Do we filter using the column passed_ss? The passed_ss value is TRUE if the taxon passed the sensitivity analysis, i.e., adding different pseudo-counts to 0s would not change the results.
`filter_diff`	(logical, default TRUE) Do we filter using the column diff? The diff value is TRUE if the taxon is significant (has q less than alpha)
`min_abs_lfc`	(integer, default 0) Minimum absolute value to filter results based on Log Fold Change. For ex. a value of 1 filter out taxa for which the abundance in a given level of the modalty is not at least the double of the abundance in the other level.
`tax_col`	The taxonomic level (must be present in `tax_table` slot) to color the points
`tax_label`	The taxonomic level (must be present in `tax_table` slot) to add label
`add_marginal_vioplot`	(logical, default TRUE) Do we add a marginal vioplot representing all the taxa lfc from ancombc_res.
`add_label`	(logical, default TRUE) Do we add a label?
`add_hline_cut_lfc`	(logical, default NULL) Do we add two horizontal lines when min_abs_lfc is set (different from zero)?

Details

This function is mainly a wrapper of the work of others. Please make a reference to `ANCOMBC::ancombc2()` if you use this function.

Value

A ggplot2 object. If add_marginal_vioplot is TRUE, this is a patchworks of plot made using `patchwork::plot_layout()`.

Author(s)

Adrien Taudière

Examples

```
## Not run:
if (requireNamespace("mia")) {
  data_fungi_mini@tax_table <- phyloseq::tax_table(cbind(
    data_fungi_mini@tax_table,
    "taxon" = taxa_names(data_fungi_mini)
  ))

  res_time <- ancombc_pq(
    data_fungi_mini,
    fact = "Time",
    levels_fact = c("0", "15"),
    tax_level = "taxon",
    verbose = TRUE
  )

  plot_ancombc_pq(data_fungi_mini, res_time,
    filter_passed = FALSE,
    tax_label = "Genus", tax_col = "Order"
  )
  plot_ancombc_pq(data_fungi_mini, res_time, tax_col = "Genus")
  plot_ancombc_pq(data_fungi_mini, res_time,
    filter_passed = FALSE,
    filter_diff = FALSE, tax_col = "Family", add_label = FALSE
  )
}

## End(Not run)
```

`plot_complexity_pq` *Plot kmer complexity of references sequences of a phyloseq object*

Description

Basically a wrapper of `dada2::seqComplexity()`

Usage

```
plot_complexity_pq(
  physeq,
  kmer_size = 2,
  window = NULL,
  by = 5,
  bins = 100,
  aggregate = FALSE,
```

```
  vline_random_kmer = TRUE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`kmer_size`	int (default 2) The size of the kmers (or "oligonucleotides" or "words") to use.
`window`	(int, default NULL) The width in nucleotides of the moving window. If NULL the whole sequence is used.
`by`	(int, default 5) The step size in nucleotides between each moving window tested.
`bins`	(int, default 100). The number of bins to use for the histogram.
`aggregate`	(logical, default FALSE) If TRUE, compute an aggregate quality profile for all samples
`vline_random_kmer`	(logical, default TRUE) If TRUE, add a vertical line at the value for random kmer (equal to 4^kmerSize))
`...`	Arguments passed on to geom_histogram.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `dada2::seqComplexity()`

Value

A ggplot2 object

Author(s)

Adrien Taudière

See Also

`dada2::seqComplexity()`, `dada2::plotComplexity()`

Examples

```
plot_complexity_pq(subset_samples(data_fungi_mini, Height == "High"),
  vline_random_kmer = FALSE
)
# plot_complexity_pq(subset_samples(data_fungi_mini, Height == "Low"),
#  aggregate = FALSE, kmer_size = 4
# )
# plot_complexity_pq(subset_samples(data_fungi, Height == "Low"),
#  kmer_size = 4)
```

`plot_deseq2_pq` *Plot DESeq2 results for a phyloseq or a DESeq2 object.*

Description

Graphical representation of DESeq2 analysis.

Usage

```
plot_deseq2_pq(
  data,
  contrast = NULL,
  tax_table = NULL,
  pval = 0.05,
  taxolev = "Genus",
  select_taxa = NULL,
  color_tax = "Phylum",
  tax_depth = NULL,
  verbose = TRUE,
  jitter_width = 0.1,
  ...
)
```

Arguments

`data`	(required) a `phyloseq-class` or a `DESeqDataSet-class` object.
`contrast`	(required) contrast specifies what comparison to extract from the object to build a results table. See `results` man page for more details.
`tax_table`	Required if data is a `DESeqDataSet-class` object. The taxonomic table used to find the `taxa` and `color_taxa` arguments. If data is a `phyloseq-class` object, data@tax_table is used.
`pval`	(default: 0.05) the significance cutoff used for optimizing the independent filtering. If the adjusted p-value cutoff (FDR) will be a value other than 0.05, pval should be set to that value.
`taxolev`	taxonomic level of interest
`select_taxa`	Either the name of the taxa (in the form of `DESeq2::results()`) or a logical vector (length of the results from `DESeq2::results()`) to select taxa to plot.
`color_tax`	taxonomic level used for color or a color vector.
`tax_depth`	Taxonomic depth to test for differential distribution among contrast. If Null the analysis is done at the OTU (i.e. Species) level. If not Null, data need to be a column name in the `tax_table` slot of the `phyloseq-class` object.
`verbose`	whether the function print some information during the computation
`jitter_width`	width for the jitter positioning
`...`	Additional arguments passed on to `DESeq` or `ggplot`

Details

Please cite `DESeq2` package if you use chis function.

Value

A `ggplot`2 plot representing DESeq2 results

Author(s)

Adrien Taudière

See Also

`DESeq`

`results`

`plot_edgeR_pq`

Examples

```
data("GlobalPatterns", package = "phyloseq")
GP <- subset_taxa(GlobalPatterns, GlobalPatterns@tax_table[, 1] == "Archaea")
GP <- subset_samples(GP, SampleType %in% c("Soil", "Skin"))
if (requireNamespace("DESeq2")) {
  res <- DESeq2::DESeq(phyloseq_to_deseq2(GP, ~SampleType),
    test = "Wald", fitType = "local"
  )
  plot_deseq2_pq(res, c("SampleType", "Soil", "Skin"),
    tax_table = GP@tax_table, color_tax = "Kingdom"
  )
  plot_deseq2_pq(res, c("SampleType", "Soil", "Skin"),
    tax_table = GP@tax_table, color_tax = "Kingdom",
    pval = 0.7
  )
  plot_deseq2_pq(res, c("SampleType", "Soil", "Skin"),
    tax_table = GP@tax_table, color_tax = "Class",
    select_taxa = c("522457", "271582")
  )
}
```

`plot_edgeR_pq` *Plot edgeR results for a phyloseq or a edgeR object.*

Description

Graphical representation of edgeR result.

Usage

```
plot_edgeR_pq(
  physeq,
  contrast = NULL,
  pval = 0.05,
  taxolev = "Genus",
  color_tax = "Phylum",
  verbose = TRUE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`contrast`	(required):This argument specifies what comparison to extract from the object to build a results table. See `results` man page for more details.
`pval`	(default: 0.05): the significance cutoff used for optimizing the independent filtering. If the adjusted p-value cutoff (FDR) will be a value other than 0.05, pval should be set to that value.
`taxolev`	taxonomic level of interest
`color_tax`	taxonomic level used for color assignation
`verbose`	(logical): whether the function print some information during the computation
`...`	Additional arguments passed on to `exactTest` or `ggplot`

Value

A `ggplot`2 plot representing edgeR results

Author(s)

Adrien Taudière

See Also

`exactTest`

`plot_deseq2_pq`

Examples

```
data("GlobalPatterns", package = "phyloseq")
GP_archae <- subset_taxa(GlobalPatterns, GlobalPatterns@tax_table[, 1] == "Archaea")

if (requireNamespace("edgeR")) {
  plot_edgeR_pq(GP_archae, c("SampleType", "Soil", "Feces"),
    color_tax = "Kingdom"
  )

  plot_edgeR_pq(GP_archae, c("SampleType", "Soil", "Feces"),
```

```
    taxolev = "Class", color_tax = "Kingdom"
  )
}
```

---

plot_guild_pq   *Plot information about Guild from tax_table slot previously created with `add_funguild_info()`*

Description

Graphical function.

Usage

```
plot_guild_pq(physeq, levels_order = NULL, clean_pq = TRUE, ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`levels_order`	(Default NULL) A character vector to reorder the levels of guild. See examples.
`clean_pq`	(logical, default TRUE): Does the phyloseq object is cleaned using the `clean_pq()` function?
`...`	Other params for be passed on to `clean_pq()` function

Value

A ggplot2 object

Author(s)

Adrien Taudière

See Also

`add_funguild_info()`

Examples

```
## Not run:
# to avoid bug in CRAN when internet is not available
if (requireNamespace("httr")) {
  d_fung_mini <- add_funguild_info(data_fungi_mini,
    taxLevels = c(
      "Domain",
      "Phylum",
      "Class",
```

```
        "Order",
        "Family",
        "Genus",
        "Species"
      )
    )
    sort(table(d_fung_mini@tax_table[, "guild"]), decreasing = TRUE)

    p <- plot_guild_pq(d_fung_mini)
    if (requireNamespace("patchwork")) {
      (plot_guild_pq(subset_samples(d_fung_mini, Height == "Low"),
        levels_order = p$data$Guild[order(p$data$nb_seq)]
      ) + theme(legend.position = "none")) +
        (plot_guild_pq(subset_samples(d_fung_mini, Height == "High"),
          levels_order = p$data$Guild[order(p$data$nb_seq)]
        ) + ylab("") + theme(axis.text.y = element_blank()))
    }
  }

  ## End(Not run)
```

`plot_LCBD_pq` *Plot and test local contributions to beta diversity (LCBD) of samples*

Description

A wrapper for the `adespatial::beta.div()` function in the case of physeq object.

Usage

```
plot_LCBD_pq(
  physeq,
  p_adjust_method = "BH",
  pval = 0.05,
  sam_variables = NULL,
  only_plot_significant = TRUE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`p_adjust_method`	(chr, default "BH"): the method used to adjust p-value
`pval`	(int, default 0.05): the value to determine the significance of LCBD
`sam_variables`	A vector of variable names present in the `sam_data` slot to plot alongside the LCBD value
`only_plot_significant`	(logical, default TRUE) Do we plot all LCBD values or only the significant ones
`...`	Additional arguments passed on to `adespatial::beta.div()` function

Details

This function is mainly a wrapper of the work of others. Please make a reference to `vegan::beta.div()` if you use this function.

Value

A ggplot2 object build with the package patchwork

Author(s)

Adrien Taudière

See Also

LCBD_pq, `adespatial::beta.div()`

Examples

```
data(data_fungi)
if (requireNamespace("adespatial")) {
  plot_LCBD_pq(data_fungi_mini,
    nperm = 100, only_plot_significant = FALSE,
    pval = 0.2
  )
}

if (requireNamespace("adespatial")) {
  plot_LCBD_pq(data_fungi_mini,
    nperm = 100, only_plot_significant = TRUE,
    pval = 0.2
  )
  if (requireNamespace("patchwork")) {
    plot_LCBD_pq(data_fungi_mini,
      nperm = 100, only_plot_significant = FALSE,
      sam_variables = c("Time", "Height")
    )
    plot_LCBD_pq(data_fungi_mini,
      nperm = 100, only_plot_significant = TRUE, pval = 0.2,
      sam_variables = c("Time", "Height", "Tree_name")
    ) &
      theme(
        legend.key.size = unit(0.4, "cm"),
        legend.text = element_text(size = 10),
        axis.title.x = element_text(size = 6)
      )
  }
}
```

---

plot_mt *Plot the result of a mt test* `phyloseq::mt()`

Description

Graphical representation of mt test.

Usage

```
plot_mt(mt = NULL, alpha = 0.05, color_tax = "Class", taxa = "Species")
```

Arguments

`mt`	(required) Result of a mt test from the function `phyloseq::mt()`.
`alpha`	(default: 0.05) Choose the cut off p-value to plot taxa.
`color_tax`	(default: "Class") A taxonomic level to color the points.
`taxa`	(default: "Species") The taxonomic level you choose for x-positioning.

Value

a `ggplot`2 plot of result of a mt test

Author(s)

Adrien Taudière

See Also

`phyloseq::mt()`

Examples

```
data_fungi_mini2 <- subset_samples(data_fungi_mini, !is.na(Time))
res <- mt(data_fungi_mini2, "Time", method = "fdr", test = "f", B = 300)
plot_mt(res)
plot_mt(res, taxa = "Genus", color_tax = "Order")
```

`plot_refseq_extremity_pq` *Plot the nucleotide proportion at both extremity of the sequences*

Description

It is a useful function to check for the absence of unwanted patterns caused for example by Illumina adaptator or bad removal of primers.

If `hill_scale` is not null, Hill diversity number are used to represent the distribution of the diversity (equitability) along the sequences.

Usage

```
plot_refseq_extremity_pq(
  physeq,
  first_n = 10,
  last_n = 10,
  hill_scales = c(1, 2),
  min_width = 0
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`first_n`	(int, default 10) The number of nucleotides to plot the 5' extremity.
`last_n`	(int, default 10) The number of nucleotides to plot the 3' extremity.
`hill_scales`	(vector) A vector defining the Hill number wanted. Set to NULL if you don't want to plot Hill diversity metrics.
`min_width`	(int, default 0) Select only the sequences from physeq@refseq with using a minimum length threshold. If `first_n` is superior to the minimum length of the references sequences, you must use min_width to filter out the narrower sequences

Value

A list of 4 objects

- p_start and p_last are the ggplot object representing respectively the start and the end of the sequences.
- df_start and df_last are the data.frame corresponding to the ggplot object.

Author(s)

Adrien Taudière

Examples

```
res1 <- plot_refseq_extremity_pq(data_fungi)
names(res1)
res1$plot_start
res1$plot_last

res2 <- plot_refseq_extremity_pq(data_fungi, first_n = 200, last_n = 100)
res2$plot_start
res2$plot_last

plot_refseq_extremity_pq(data_fungi,
  first_n = 400,
  min_width = 400,
  hill_scales = NULL
)$plot_start +
  geom_line(aes(y = value, x = seq_id, color = name), alpha = 0.4, linewidth = 0.2)

plot_refseq_extremity_pq(data_fungi,
  first_n = NULL,
  last_n = 400,
  min_width = 400,
  hill_scales = c(3)
)$plot_last
```

`plot_refseq_pq` *Plot the nucleotide proportion of references sequences*

Description

It is a wrapper of the function `plot_refseq_extremity_pq()`. See ?plot_refseq_extremity_pq for more examples.

If `hill_scale` is not null, Hill diversity number are used to represent the distribution of the diversity (equitability) along the sequences.

Usage

```
plot_refseq_pq(
  physeq,
  hill_scales = NULL,
  first_n = min(Biostrings::width(physeq@refseq)),
  last_n = NULL,
  min_width = first_n
)
```

Arguments

physeq	(required): a `phyloseq-class` object obtained using the `phyloseq` package.

`hill_scales`	(vector) A vector defining the Hill number wanted. Set to NULL if you don't want to plot Hill diversity metrics.
`first_n`	(int, default 10) The number of nucleotides to plot the 5' extremity.
`last_n`	(int, default 10) The number of nucleotides to plot the 3' extremity.
`min_width`	(int, default 0) Select only the sequences from physeq@refseq with using a minimum length threshold. If `first_n` is superior to the minimum length of the references sequences, you must use min_width to filter out the narrower sequences

Value

A ggplot2 object

Author(s)

Adrien Taudière

Examples

```
plot_refseq_pq(data_fungi)
plot_refseq_pq(data_fungi, hill_scales = c(2), first_n = 300)
```

`plot_SCBD_pq` *Plot species contributions to beta diversity (SCBD) of samples*

Description

A wrapper for the `adespatial::beta.div()` function in the case of physeq object.

Usage

```
plot_SCBD_pq(
  physeq,
  tax_level = "Taxa",
  tax_col = "Order",
  min_SCBD = 0.01,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`tax_level`	Taxonomic level to used in y axis
`tax_col`	Taxonomic level to colored points
`min_SCBD`	(default 0.01) the minimum SCBD value to plot the taxa
`...`	Additional arguments passed on to `adespatial::beta.div()` function

Details

This function is mainly a wrapper of the work of others. Please make a reference to `vegan::beta.div()` if you use this function.

Value

A ggplot2 object build with the package patchwork

Author(s)

Adrien Taudière

See Also

LCBD_pq, `adespatial::beta.div()`

Examples

```
data(data_fungi)
if (requireNamespace("adespatial")) {
  plot_SCBD_pq(data_fungi) +
    geom_text(aes(label = paste(Genus, Species)), hjust = 1, vjust = 2) +
    xlim(c(0, NA))
}

if (requireNamespace("adespatial")) {
  plot_SCBD_pq(data_fungi, tax_level = "Class", tax_col = "Phylum", min_SCBD = 0) +
    geom_jitter()
}
```

`plot_tax_pq` *Plot taxonomic distribution in function of a factor with stacked bar in %*

Description

An alternative to `phyloseq::plot_bar()` function.

Usage

```
plot_tax_pq(
  physeq,
  fact = NULL,
  merge_sample_by = NULL,
  type = "nb_seq",
  taxa_fill = "Order",
```

```
  print_values = TRUE,
  color_border = "lightgrey",
  linewidth = 0.1,
  prop_print_value = 0.01,
  nb_print_value = NULL,
  add_info = TRUE,
  na_remove = TRUE,
  clean_pq = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`fact`	(required) Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`.
`merge_sample_by`	a vector to determine which samples to merge using the `merge_samples2()` function. Need to be in `physeq@sam_data`
`type`	If "nb_seq" (default), the number of sequences is used in plot. If "nb_taxa", the number of ASV is plotted. If both, return a list of two plots, one for nbSeq and one for ASV.
`taxa_fill`	(default: 'Order'): Name of the taxonomic rank of interest
`print_values`	(logical, default TRUE): Do we print some values on plot?
`color_border`	color for the border
`linewidth`	The line width of geom_bar
`prop_print_value`	minimal proportion to print value (default 0.01)
`nb_print_value`	number of higher values to print (replace prop_print_value if both are set).
`add_info`	(logical, default TRUE) Do we add title and subtitle with information about the total number of sequences and the number of samples per modality.
`na_remove`	(logical, default TRUE) if TRUE remove all the samples with NA in the `split_by` variable of the `physeq@sam_data` slot
`clean_pq`	(logical) If set to TRUE, empty samples are discarded after subsetting ASV

Value

A ggplot2 object

Author(s)

Adrien Taudière

See Also

`tax_bar_pq()` and `multitax_bar_pq()`

Examples

```
data(data_fungi_sp_known)
plot_tax_pq(data_fungi_sp_known,
  "Time",
  merge_sample_by = "Time",
  taxa_fill = "Class"
)

plot_tax_pq(data_fungi_sp_known,
  "Height",
  merge_sample_by = "Height",
  taxa_fill = "Class",
  na_remove = TRUE,
  color_border = rgb(0, 0, 0, 0)
)

plot_tax_pq(data_fungi_sp_known,
  "Height",
  merge_sample_by = "Height",
  taxa_fill = "Class",
  na_remove = FALSE,
  clean_pq = FALSE
)
```

`plot_tsne_pq` *Plot a tsne low dimensional representation of a phyloseq object*

Description

Partially inspired by `phylosmith::tsne_phyloseq()` function developed by Schuyler D. Smith.

Usage

```
plot_tsne_pq(
  physeq,
  method = "bray",
  dims = 2,
  theta = 0,
  perplexity = 30,
  fact = NA,
  ellipse_level = 0.95,
  plot_dims = c(1, 2),
  na_remove = TRUE,
  force_factor = TRUE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`method`	A method to calculate distance using `vegan::vegdist()` function (default: "bray")
`dims`	(Int) Output dimensionality (default: 2)
`theta`	(Numeric) Speed/accuracy trade-off (increase for less accuracy), set to 0.0 for exact TSNE (default: 0.0 see details in the man page of `Rtsne::Rtsne`).
`perplexity`	(Numeric) Perplexity parameter (should not be bigger than 3 * perplexity < nrow(X) - 1, see details in the man page of `Rtsne::Rtsne`)
`fact`	Name of the column in `physeq@sam_data` used to color points and compute ellipses.
`ellipse_level`	The level used in stat_ellipse. Set to NULL to discard ellipse (default = 0.95)
`plot_dims`	A vector of 2 values defining the rank of dimension to plot (default: c(1,2))
`na_remove`	(logical, default TRUE) Does the samples with NA values in fact are removed? (default: true)
`force_factor`	(logical, default TRUE) Force the fact column to be a factor.
`...`	Additional arguments passed on to `Rtsne::Rtsne()`

Value

A ggplot object

Author(s)

Adrien Taudière

Examples

```
data(data_fungi)
if (requireNamespace("Rtsne")) {
  plot_tsne_pq(data_fungi, fact = "Height", perplexity = 15)
}

if (requireNamespace("Rtsne")) {
  plot_tsne_pq(data_fungi, fact = "Time") + geom_label(aes(label = Sample_id, fill = Time))
  plot_tsne_pq(data_fungi, fact = "Time", na_remove = FALSE, force_factor = FALSE)
}
```

`plot_var_part_pq` *Plot the partition the variation of a phyloseq object*

Description

Graphical representation of the partition of variation obtain with `var_par_pq()`.

Usage

```
plot_var_part_pq(
  res_varpart,
  cutoff = 0,
  digits = 1,
  digits_quantile = 2,
  fill_bg = c("seagreen3", "mediumpurple", "blue", "orange"),
  show_quantiles = FALSE,
  filter_quantile_zero = TRUE,
  show_dbrda_signif = FALSE,
  show_dbrda_signif_pval = 0.05,
  alpha = 63,
  id.size = 1.2,
  min_prop_pval_signif_dbrda = 0.95
)
```

Arguments

`res_varpart`	(required) the result of the functions `var_par_pq()` or `var_par_rarperm_pq()`
`cutoff`	The values below cutoff will not be displayed.
`digits`	The number of significant digits.
`digits_quantile`	The number of significant digits for quantile.
`fill_bg`	Fill colours of ellipses.
`show_quantiles`	Do quantiles are printed ?
`filter_quantile_zero`	Do we filter out value with quantile encompassing the zero value?
`show_dbrda_signif`	Do dbrda significance for each component is printed using *?
`show_dbrda_signif_pval`	(float, `[0:1]`) The value under which the dbrda is considered significant.
`alpha`	(int, `[0:255]`) Transparency of the fill colour.
`id.size`	A numerical value giving the character expansion factor for the names of circles or ellipses.
`min_prop_pval_signif_dbrda`	(float, `[0:1]`) Only used if using the result of `var_par_rarperm_pq()` function. The * for dbrda_signif is only add if at least `min_prop_pval_signif_dbrda` of permutations show significance.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `vegan::varpart()` if you use this function.

Value

A plot

Author(s)

Adrien Taudière

See Also

`var_par_rarperm_pq()`, `var_par_pq()`

Examples

```
if (requireNamespace("vegan")) {
  data_fungi_woNA <- subset_samples(data_fungi, !is.na(Time) & !is.na(Height))
  res_var_9 <- var_par_rarperm_pq(
    data_fungi_woNA,
    list_component = list(
      "Time" = c("Time"),
      "Size" = c("Height", "Diameter")
    ),
    nperm = 9,
    dbrda_computation = TRUE
  )
  res_var_2 <- var_par_rarperm_pq(
    data_fungi_woNA,
    list_component = list(
      "Time" = c("Time"),
      "Size" = c("Height", "Diameter")
    ),
    nperm = 2,
    dbrda_computation = TRUE
  )
  res_var0 <- var_par_pq(data_fungi_woNA,
    list_component = list(
      "Time" = c("Time"),
      "Size" = c("Height", "Diameter")
    ),
    dbrda_computation = TRUE
  )
  plot_var_part_pq(res_var0, digits_quantile = 2, show_dbrda_signif = TRUE)
  plot_var_part_pq(res_var_9,
    digits_quantile = 2, show_quantiles = TRUE,
    show_dbrda_signif = TRUE
  )
  plot_var_part_pq(
    res_var_2,
```

```
    digits = 5,
    digits_quantile = 2,
    cutoff = 0,
    show_quantiles = TRUE
  )
}
```

---

postcluster_pq *Recluster sequences of an object of class* physeq *or a list of DNA sequences*

Description

This function use the `merge_taxa_vec` function to merge taxa into clusters.

Usage

```
postcluster_pq(
  physeq = NULL,
  dna_seq = NULL,
  nproc = 1,
  method = "clusterize",
  id = 0.97,
  vsearchpath = "vsearch",
  tax_adjust = 0,
  vsearch_cluster_method = "--cluster_size",
  vsearch_args = "--strand both",
  keep_temporary_files = FALSE,
  swarmpath = "swarm",
  d = 1,
  swarm_args = "--fastidious",
  method_clusterize = "overlap",
  ...
)

asv2otu(
  physeq = NULL,
  dna_seq = NULL,
  nproc = 1,
  method = "clusterize",
  id = 0.97,
  vsearchpath = "vsearch",
  tax_adjust = 0,
  vsearch_cluster_method = "--cluster_size",
  vsearch_args = "--strand both",
  keep_temporary_files = FALSE,
```

```
    swarmpath = "swarm",
    d = 1,
    swarm_args = "--fastidious",
    method_clusterize = "overlap",
    ...
  )
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`dna_seq`	You may directly use a character vector of DNA sequences in place of physeq args. When physeq is set, dna sequences take the value of `physeq@refseq`
`nproc`	(default: 1) Set to number of cpus/processors to use for the clustering
`method`	(default: clusterize) Set the clustering method. • `clusterize` use the `DECIPHER::Clusterize()` fonction, • `vsearch` use the vsearch software (https://github.com/torognes/vsearch) with arguments `--cluster_size` by default (see args `vsearch_cluster_method`) and `-strand both` (see args `vsearch_args`) • `swarm` use the swarm
`id`	(default: 0.97) level of identity to cluster
`vsearchpath`	(default: vsearch) path to vsearch
`tax_adjust`	(Default 0) See the man page of `merge_taxa_vec()` for more details. To conserved the taxonomic rank of the most abundant taxa (ASV, OTU,...), set tax_adjust to 0 (default). For the moment only tax_adjust = 0 is robust
`vsearch_cluster_method`	(default: "–cluster_size) See other possible methods in the vsearch manual (e.g. `--cluster_size` or `--cluster_smallmem`) • `--cluster_fast` : Clusterize the fasta sequences in filename, automatically sort by decreasing sequence length beforehand. • `--cluster_size` : Clusterize the fasta sequences in filename, automatically sort by decreasing sequence abundance beforehand. • `--cluster_smallmem` : Clusterize the fasta sequences in filename without automatically modifying their order beforehand. Sequence are expected to be sorted by decreasing sequence length, unless *–usersort* is used. In that case you may set `vsearch_args` to vsearch_args = "–strand both –usersort"
`vsearch_args`	(default : "–strand both") a one length character element defining other parameters to passed on to vsearch.
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files • temp.fasta (refseq in fasta or dna_seq sequences) • cluster.fasta (centroid if method = "vsearch") • temp.uc (clusters if method = "vsearch")
`swarmpath`	(default: swarm) path to swarm

`d`	(default: 1) maximum number of differences allowed between two amplicons, meaning that two amplicons will be grouped if they have `d` (or less) differences
`swarm_args`	(default : "–fastidious") a one length character element defining other parameters to passed on to swarm See other possible methods in the SWARM pdf manual
`method_clusterize`	(default "overlap") the method for the `DECIPHER::Clusterize()` method
`...`	Additional arguments passed on to `DECIPHER::Clusterize()`

Details

This function use the `merge_taxa_vec` function to merge taxa into clusters. By default tax_adjust = 0. See the man page of `merge_taxa_vec()`.

Value

A new object of class physeq or a list of cluster if dna_seq args was used.

Author(s)

Adrien Taudière

References

VSEARCH can be downloaded from `https://github.com/torognes/vsearch`. More information in the associated publication `https://pubmed.ncbi.nlm.nih.gov/27781170`.

See Also

`vsearch_clustering()` and `swarm_clustering()`

Examples

```
if (requireNamespace("DECIPHER")) {
  postcluster_pq(data_fungi_mini)
}

if (requireNamespace("DECIPHER")) {
  postcluster_pq(data_fungi_mini, method_clusterize = "longest")

  if (MiscMetabar::is_swarm_installed()) {
    d_swarm <- postcluster_pq(data_fungi_mini, method = "swarm")
  }
  if (MiscMetabar::is_vsearch_installed()) {
    d_vs <- postcluster_pq(data_fungi_mini, method = "vsearch")
  }
}
```

---

psmelt_samples_pq *Build a sample information tibble from physeq object*

Description

Hill numbers are the number of equiprobable species giving the same diversity value as the observed distribution.

Note that contrary to `hill_pq()`, this function does not take into account for difference in the number of sequences per samples/modalities. You may use rarefy_by_sample = TRUE if the mean number of sequences per samples differs among modalities.

Usage

```
psmelt_samples_pq(
  physeq,
  hill_scales = c(0, 1, 2),
  filter_zero = TRUE,
  rarefy_by_sample = FALSE,
  taxa_ranks = NULL
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`hill_scales`	(a vector of integer) The list of q values to compute the hill number H^q. If Null, no hill number are computed. Default value compute the Hill number 0 (Species richness), the Hill number 1 (exponential of Shannon Index) and the Hill number 2 (inverse of Simpson Index).
`filter_zero`	(logical, default TRUE) Do we filter non present OTU from samples ? For the moment, this has no effect on the result because the dataframe is grouped by samples with abundance summed across OTU.
`rarefy_by_sample`	(logical, default FALSE) If TRUE, rarefy samples using `phyloseq::rarefy_even_depth()` function.
`taxa_ranks`	A vector of taxonomic ranks. For examples c("Family","Genus"). If taxa ranks is not set (default value = NULL), taxonomic information are not present in the resulting tibble.

Value

A tibble with a row for each sample. Columns provide information from `sam_data` slot as well as hill numbers, Abundance (nb of sequences), and Abundance_log10 (*log10(1+Abundance)*).

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("ggstatsplot")) {
  psm_tib <- psmelt_samples_pq(data_fungi_mini, hill_scales = c(0, 2, 7))
  ggstatsplot::ggbetweenstats(psm_tib, Height, Hill_0)
  ggstatsplot::ggbetweenstats(psm_tib, Height, Hill_7)

  psm_tib_tax <- psmelt_samples_pq(data_fungi_mini, taxa_ranks = c("Class", "Family"))
 ggplot(filter(psm_tib_tax, Abundance > 2000), aes(y = Family, x = Abundance, fill = Time)) +
    geom_bar(stat = "identity") +
    facet_wrap(~Height)
}
```

`rarefy_sample_count_by_modality`

Rarefy (equalize) the number of samples per modality of a factor

---

Description

This function randomly draw the same number of samples for each modality of factor. It is usefull to dissentangle the effect of different number of samples per modality on diversity. Internally used in `accu_plot_balanced_modality()`.

Usage

```
rarefy_sample_count_by_modality(physeq, fact, rngseed = FALSE, verbose = TRUE)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required): The variable to rarefy. Must be present in the `sam_data` slot of the physeq object.
`rngseed`	(Optional). A single integer value passed to set.seed, which is used to fix a seed for reproducibly random number generation (in this case, reproducibly random subsampling). If set to FALSE, then no iddling with the RNG seed is performed, and it is up to the user to appropriately call
`verbose`	(logical). If TRUE, print additional information.

Value

A new `phyloseq-class` object.

Author(s)

Adrien Taudière

See Also

`accu_plot_balanced_modality()`

Examples

```
table(data_fungi_mini@sam_data$Height)
data_fungi_mini2 <- rarefy_sample_count_by_modality(data_fungi_mini, "Height")
table(data_fungi_mini2@sam_data$Height)
if (requireNamespace("patchwork")) {
  ggvenn_pq(data_fungi_mini, "Height") + ggvenn_pq(data_fungi_mini2, "Height")
}
```

`read_pq` *Read phyloseq object from multiple csv tables and a phylogenetic tree in Newick format.*

Description

This is the reverse function of `write_pq()`.

Usage

```
read_pq(
  path = NULL,
  taxa_are_rows = FALSE,
  sam_names = NULL,
  sep_csv = "\t",
  ...
)
```

Arguments

`path`	(required) a path to the folder to read the phyloseq object
`taxa_are_rows`	(default to FALSE) see ?phyloseq for details
`sam_names`	The name of the variable (column) in sam_data.csv to rename samples. Note that if you use `write_phyloseq()` function to save your physeq object, you may use sam_names = "X" to rename the samples names as before.
`sep_csv`	(default tabulation) separator for column
`...`	Additional arguments passed on to `utils::write.table()` function.

Value

One to four csv tables (refseq.csv, otu_table.csv, tax_table.csv, sam_data.csv) and if present a phy_tree in Newick format. At least the otu_table.csv need to be present.

Author(s)

Adrien Taudière

Examples

```
write_pq(data_fungi, path = paste0(tempdir(), "/phyloseq"))
read_pq(path = paste0(tempdir(), "/phyloseq"))
unlink(paste0(tempdir(), "/phyloseq"), recursive = TRUE)
```

`rename_samples` *Rename the samples of a phyloseq slot*

Description

Useful for targets bioinformatic pipeline.

Usage

```
rename_samples(phyloseq_component, names_of_samples, taxa_are_rows = FALSE)
```

Arguments

`phyloseq_component`	(required) one of otu_table or sam_data slot of a phyloseq-class object
`names_of_samples`	(required) A vector of samples names
`taxa_are_rows`	(default to FALSE) see ?phyloseq for details

Value

The otu_table or the sam_data slot with new samples names

Author(s)

Adrien Taudière

Examples

```
otutab <- rename_samples(
  data_fungi@otu_table,
  paste0("data_f", sample_names(data_fungi))
)
otutab2 <- rename_samples(
  clean_pq(data_fungi,
    force_taxa_as_rows = TRUE
  )@otu_table,
  paste0("data_f", sample_names(data_fungi))
)
samda <- rename_samples(
  data_fungi@sam_data,
  paste0("data_f", sample_names(data_fungi))
)
```

`rename_samples_otu_table` *Rename samples of an otu_table*

Description

Useful for targets bioinformatic pipeline.

Usage

```
rename_samples_otu_table(physeq, names_of_samples)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`names_of_samples`	(required) The new names of the samples

Value

the matrix with new colnames (or rownames if `taxa_are_rows` is true)

Author(s)

Adrien Taudière

Examples

```
rename_samples_otu_table(data_fungi, as.character(seq_along(sample_names(data_fungi))))
```

`reorder_taxa_pq` *Reorder taxa in otu_table/tax_table/refseq slot of a phyloseq object*

Description

Note that the taxa order in a physeq object with a tree is locked by the order of leaf in the phylogenetic tree.

Usage

```
reorder_taxa_pq(physeq, names_ordered, remove_phy_tree = FALSE)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`names_ordered`	(required): Names of the taxa (must be the same as taxa in `taxa_names(physeq)`) in a given order
`remove_phy_tree`	(logical, default FALSE) If TRUE, the phylogenetic tree is removed. It is

Value

A phyloseq object

Author(s)

Adrien Taudière

Examples

```
data_fungi_ordered_by_genus <- reorder_taxa_pq(
  data_fungi,
  taxa_names(data_fungi)[order(as.vector(data_fungi@tax_table[, "Genus"]))]
)

data_fungi_mini_asc_ordered_by_abundance <- reorder_taxa_pq(
  data_fungi_mini,
  taxa_names(data_fungi_mini)[order(taxa_sums(data_fungi_mini))]
)
```

`resolve_vector_ranks` *Resolve conflict in a vector of taxonomy values*

Description

Internally used in the function `assign_blastn()` with method="vote" and `assign_vsearch_lca()` if `top_hits_only` is FALSE and `vote_algorithm` is not NULL.

Usage

```
resolve_vector_ranks(
  vec,
 method = c("consensus", "rel_majority", "abs_majority", "preference", "unanimity"),
  strict = FALSE,
  second_method = c("consensus", "rel_majority", "abs_majority", "unanimity"),
  nb_agree_threshold = 1,
  preference_index = NULL,
  collapse_string = "/",
  replace_collapsed_rank_by_NA = FALSE
)
```

Arguments

`vec`	(required) A vector of (taxonomic) values
`method`	One of "consensus", "rel_majority", "abs_majority", "preference" or "unanimity". See details.
`strict`	(logical, default FALSE). If TRUE, NA are considered as informative in resolving conflict (i.e. NA are taking into account in vote). See details for more informations.
`second_method`	One of "consensus", "rel_majority", "abs_majority", or "unanimity". Only used if method = "preference". See details.
`nb_agree_threshold`	(Int, default 1) The minimum number of times a value must arise to be selected using vote. If 2, we only kept taxonomic value present at least 2 times in the vector.
`preference_index`	(Int. default NULL). Required if method="preference". Useless for other method. The preference index is the index of the value in vec for which we have a preference.
`collapse_string`	(default '/'). The character to collapse taxonomic names when multiple assignment is done.
`replace_collapsed_rank_by_NA`	(logical, default FALSE). If set to TRUE, all multiple assignments (all taxonomic rank including the 'collapse_string' parameter) are replaced by NA.

Details

- `unanimity`: Only keep taxonomic value when all methods are agree
  - By default, the value with NA are not taking into account (strict=FALSE)
  - If `strict` , one NA in the row is sufficient to return a NA
- `consensus`: Keep all taxonomic values separated by a '/' (separation can be modify using param `collapse_string`)
  - If `strict` is TRUE, NA are also added to taxonomic vector such as 'Tiger/Cat/NA' instead of 'Tiger/Cat'
- `abs_majority`: Keep the most found taxonomic value if it represent at least half of all taxonomic values.
  - If `strict` is TRUE, NA values are also count to determine the majority. For example, a vector of taxonomic rank c("A", "A", "A", "B", NA, NA) will give a value of 'A' if `strict` is FALSE (default) but a value of NA if `strict` is TRUE.
- `rel_majority`: Keep the most found taxonomic value. In case of equality, apply a consensus strategy (collapse values separated by a '/') across the most found taxonomic values.
  - If `strict` is TRUE, NA are considered as a rank and can win the relative majority vote. If `strict` is FALSE (default), NA are removed before ranking the taxonomic values.
  - `nb_agree_threshold`: Only keep return value when at least x methods agreed with x is set by parameter `nb_agree_threshold`. By default, (`nb_agree_threshold` = 1): a majority of one is enough.

- `preference`: Keep the value from a preferred column.
    - when the value is NA in the preferred column, apply a second strategy (by default `consensus`) to the other column (parameter `second_method`). Note that the parameters `strict` and `nb_agree_threshold` are used for the second_method consensus.

Value

a vector of length 1 (one character value)

Author(s)

Adrien Taudière

Examples

```
resolve_vector_ranks(c("A"))
resolve_vector_ranks(c("A"),
  method = "preference",
  preference_index = 1
)
resolve_vector_ranks(c("A"), method = "abs_majority")
resolve_vector_ranks(c("A"), method = "rel_majority")
resolve_vector_ranks(c("A"),
  method = "rel_majority",
  nb_agree_threshold = 2
)
resolve_vector_ranks(c("A"), method = "unanimity")

resolve_vector_ranks(c("A", "A", "A"))
resolve_vector_ranks(c("A", "A", "A"),
  method = "preference",
  preference_index = 1
)
resolve_vector_ranks(c("A", "A", "A"), method = "abs_majority")
resolve_vector_ranks(c("A", "A", "A"), method = "rel_majority")
resolve_vector_ranks(c("A", "A", "A"), method = "unanimity")

resolve_vector_ranks(c(NA, NA, NA))
resolve_vector_ranks(c(NA, NA, NA),
  method = "preference",
  preference_index = 1
)
resolve_vector_ranks(c(NA, NA, NA), method = "abs_majority")
resolve_vector_ranks(c(NA, NA, NA), method = "rel_majority")
resolve_vector_ranks(c(NA, NA, NA), method = "unanimity")

resolve_vector_ranks(c("A", "A", NA))
resolve_vector_ranks(c("A", "A", NA),
  method = "preference",
  preference_index = 1
)
resolve_vector_ranks(c("A", "A", NA), method = "rel_majority")
```

```
resolve_vector_ranks(c("A", "A", NA), method = "abs_majority")
resolve_vector_ranks(c("A", "A", NA, NA),
  method = "abs_majority",
  strict = FALSE
)
resolve_vector_ranks(c("A", "A", NA, NA),
  method = "abs_majority",
  strict = TRUE
)
resolve_vector_ranks(c("A", "A", NA), method = "unanimity")
resolve_vector_ranks(c("A", "A", NA),
  method = "unanimity",
  strict = TRUE
)

resolve_vector_ranks(c("A", "B", NA))
resolve_vector_ranks(c("A", "B", NA), strict = TRUE)
resolve_vector_ranks(c("A", "B", NA),
  method = "preference",
  preference_index = 1
)
resolve_vector_ranks(c("A", "B", NA), method = "abs_majority")
resolve_vector_ranks(c("A", "B", NA), method = "rel_majority")
resolve_vector_ranks(c("A", "B", NA),
  method = "rel_majority",
  strict = TRUE
)
resolve_vector_ranks(c("A", "B", NA),
  method = "rel_majority",
  nb_agree_threshold = 2
)
resolve_vector_ranks(c("A", "B", NA), method = "unanimity")

resolve_vector_ranks(c("A", NA, NA))
resolve_vector_ranks(c("A", NA, NA), method = "rel_majority")
resolve_vector_ranks(c("A", NA, NA), method = "unanimity")
resolve_vector_ranks(c("A", NA, NA),
  method = "preference",
  preference_index = 1
)
resolve_vector_ranks(c("A", NA, NA),
  method = "preference",
  preference_index = 2
)
resolve_vector_ranks(c("A", NA, "B"),
  method = "preference",
  preference_index = 2
)
resolve_vector_ranks(c("A", NA, "B"),
  method = "preference",
  preference_index = 2, second_method = "abs_majority"
)
```

```
resolve_vector_ranks(c("A", "B", "B"))
resolve_vector_ranks(c("A", "B", "B"),
  method = "preference",
  preference_index = 1
)
resolve_vector_ranks(c("A", "B", "B"), method = "abs_majority")
resolve_vector_ranks(c("A", "B", "B"), method = "rel_majority")
resolve_vector_ranks(c("A", "B", "B"), method = "unanimity")


resolve_vector_ranks(c("A", "A", "A", "B", NA, NA))
resolve_vector_ranks(c("A", "A", "A", "B", NA, NA),
  strict = TRUE
)
resolve_vector_ranks(c("A", "A", "A", "B", NA, NA),
  method = "abs_majority"
)
resolve_vector_ranks(c("A", "A", "A", "B", NA, NA),
  method = "abs_majority",
  strict = TRUE
)
resolve_vector_ranks(c("A", "A", "A", "B", NA, NA),
  method = "preference", preference_index = 6, second_method = "abs_majority"
)
resolve_vector_ranks(c("A", "A", "A", "B", NA, NA, NA),
  method = "preference", preference_index = 6, second_method = "abs_majority"
)
resolve_vector_ranks(c("A", "A", "A", "B", NA, NA, NA),
  method = "preference", preference_index = 6, second_method = "abs_majority",
  strict = TRUE
)
```

`ridges_pq` *Ridge plot of a phyloseq object*

Description

Graphical representation of distribution of taxa across a factor using ridges.

Usage

```
ridges_pq(
  physeq,
  fact,
  nb_seq = TRUE,
  log10trans = TRUE,
  tax_level = "Class",
  type = "density",
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required) Name of the factor in `physeq@sam_data` used to plot different lines
`nb_seq`	(logical; default TRUE) If set to FALSE, only the number of ASV is count. Concretely, physeq `otu_table` is transformed in a binary `otu_table` (each value different from zero is set to one)
`log10trans`	(logical, default TRUE) If TRUE, the number of sequences (or ASV if nb_seq = FALSE) is log10 transformed.
`tax_level`	The taxonomic level to fill ridges
`type`	Either "density" (the default) or "ecdf" to plot a plot a cumulative version using `ggplot2::stat_ecdf()`
`...`	Other params passed on to `ggridges::geom_density_ridges()`

Value

A `ggplot`2 plot with bar representing the number of sequence en each taxonomic groups

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("ggridges")) {
  ridges_pq(data_fungi_mini, "Time", alpha = 0.5, log10trans = FALSE) + xlim(c(0, 1000))
}

if (requireNamespace("ggridges")) {
  ridges_pq(data_fungi_mini, "Time", alpha = 0.5, scale = 0.9)
  ridges_pq(data_fungi_mini, "Time", alpha = 0.5, scale = 0.9, type = "ecdf")
  ridges_pq(data_fungi_mini, "Sample_names", log10trans = TRUE) + facet_wrap("~Height")

  ridges_pq(data_fungi_mini,
    "Time",
    jittered_points = TRUE,
    position = ggridges::position_points_jitter(width = 0.05, height = 0),
    point_shape = "|", point_size = 3, point_alpha = 1, alpha = 0.7,
    scale = 0.8
  )
}
```

`rotl_pq` *rotl wrapper for phyloseq data*

Description

Make a phylogenetic tree using the ASV names of a physeq object and the Open Tree of Life tree.

Usage

```
rotl_pq(physeq, species_colnames = "Genus_species", context_name = "All life")
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`species_colnames`	(default: "Genus_species"): the name of the column where the species binominal name is stored in `@tax_table` slot. Can also be a vector of two columns names e.g. `c("Genus", "Species")`
`context_name`	: can bue used to select only a part of the Open Tree of Life. See `?rotl::tnrs_contexts()` for available values

Details

This function is mainly a wrapper of the work of others. Please make a reference to `rotl` package if you use this function.

Value

A plot

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("rotl")) {
  tr <- rotl_pq(data_fungi_mini, species_colnames = "Genus_species")
  plot(tr)

 tr_Asco <- rotl_pq(data_fungi, species_colnames = "Genus_species", context_name = "Ascomycetes")
  plot(tr_Asco)
}
```

`sample_data_with_new_names`

Load sample data from file and rename samples using names of samples and an optional order

---

Description

Useful for targets bioinformatic pipeline.

Usage

```
sample_data_with_new_names(
  file_path,
  names_of_samples,
  samples_order = NULL,
  ...
)
```

Arguments

`file_path`	(required) a path to the sample_data file
`names_of_samples`	(required) a vector of sample names
`samples_order`	Optional numeric vector to sort sample names
`...`	Additional arguments passed on to `utils::read.delim()` function.

Value

A data.frame from file_path and new names

Author(s)

Adrien Taudière

See Also

`rename_samples()`

Examples

```
sam_file <- system.file("extdata", "sam_data.csv", package = "MiscMetabar")
sample_data_with_new_names(sam_file, paste0("Samples_", seq(1, 185)))
```

`sam_data_matching_names`

Match sample names from sam_data and fastq files

Description

Useful for targets bioinformatic pipeline.

Usage

```
sam_data_matching_names(
  path_sam_data,
  sample_col_name,
  path_raw_seq,
  pattern_remove_sam_data = NULL,
  pattern_remove_fastq_files = NULL,
  verbose = TRUE,
  remove_undocumented_fastq_files = FALSE,
  prefix = NULL,
  ...
)
```

Arguments

`path_sam_data`	(Required) Path to sample data file.
`sample_col_name`	(Required) The name of the column defining sample names in the sample data file.
`path_raw_seq`	(Required) Path to the folder containing fastq files
`pattern_remove_sam_data`	If not null, describe the pattern that will be deleted from sam_data samples names.
`pattern_remove_fastq_files`	If not null, describe the pattern that will be deleted from fastq files names.
`verbose`	(logical, default TRUE) If TRUE, print some additional messages.
`remove_undocumented_fastq_files`	(logical, default FALSE) If set to TRUE fastq files not present in sam_data are removed from your folder. Keep a copy of those files somewhere before.
`prefix`	Add a prefix to new samples names (ex. prefix = "samp")
`...`	Other parameters passed on to `utils::read.csv()` function.

Value

A list of two objects :

- $sam_names_matching is a tibble of corresponding samples names
- $sam_data is a sample data files including only matching sample names

Author(s)

Adrien Taudière

`sankey_pq` *Sankey plot of* `phyloseq-class` *object*

Description

Graphical representation of distribution of taxa across Taxonomy and (optionnaly a factor).

Usage

```
sankey_pq(
  physeq = NULL,
  fact = NULL,
  taxa = 1:4,
  add_nb_seq = FALSE,
  min_prop_tax = 0,
  tax2remove = NULL,
  units = NULL,
  symbol2sub = c("\\.", "-"),
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`.
`taxa`	a vector of taxonomic rank to plot
`add_nb_seq`	Represent the number of sequences or the number of OTUs (add_nb_seq = FALSE). Note that plotting the number of sequences is slower.
`min_prop_tax`	(default: 0) The minimum proportion for taxa to be plotted. EXPERIMENTAL. For the moment each links below the min.prop. tax is discard from the sankey network resulting in sometimes weird plot.
`tax2remove`	a vector of taxonomic groups to remove from the analysis (e.g. `c('Incertae sedis', 'unidentified')`)
`units`	character string describing physical units (if any) for Value
`symbol2sub`	(default: c('\.', '-')) vector of symbol to delete in the taxonomy
`...`	Additional arguments passed on to `sankeyNetwork`

Value

A `sankeyNetwork` plot representing the taxonomic distribution of OTUs or sequences. If `fact` is set, represent the distribution of the last taxonomic level in the modalities of `fact`

Author(s)

Adrien Taudière

See Also

`sankeyNetwork`, `ggaluv_pq()`

Examples

```
data("GlobalPatterns", package = "phyloseq")
GP <- subset_taxa(GlobalPatterns, GlobalPatterns@tax_table[, 1] == "Archaea")
if (requireNamespace("networkD3")) {
  sankey_pq(GP, fact = "SampleType")
}

if (requireNamespace("networkD3")) {
  sankey_pq(GP, taxa = 1:4, min_prop_tax = 0.01)
  sankey_pq(GP, taxa = 1:4, min_prop_tax = 0.01, add_nb_seq = TRUE)
}
```

`save_pq` *A wrapper of write_pq to save in all three possible formats*

Description

A wrapper of write_pq to save in all three possible formats

Usage

```
save_pq(physeq, path = NULL, ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`path`	a path to the folder to save the phyloseq object
`...`	Additional arguments passed on to `write_pq()` or `utils::write.table()` function.

Details

Write :

- 4 separate tables
- 1 table version
- 1 RData file

Value

Build a folder (in path) with four csv tables (`refseq.csv`, `otu_table.csv`, `tax_table.csv`, `sam_data.csv`) + one table with all tables together + a rdata file (`physeq.RData`) that can be loaded using `base::load()` function + if present a phylogenetic tree in Newick format (`phy_tree.txt`)

Author(s)

Adrien Taudière

See Also

`write_pq()`

Examples

```
save_pq(data_fungi, path = paste0(tempdir(), "/phyloseq"))
unlink(paste0(tempdir(), "/phyloseq"), recursive = TRUE)
```

`search_exact_seq_pq` *Search for exact matching of sequences*

Description

Search for exact matching of sequences using complement, reverse and reverse-complement. It is useful to check for primers issues after cutadapt step.

Usage

```
search_exact_seq_pq(physeq, seq2search)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`seq2search`	A DNAStringSet object of sequences to search for.

Value

A list of data-frames for each input sequences with the name, the sequences and the number of occurrences of the original sequence, the complement sequence, the reverse sequence and the reverse-complement sequence.

Author(s)

Adrien Taudière

Examples

```
data("data_fungi")
search_primers <- search_exact_seq_pq(data_fungi,
  seq2search = Biostrings::DNAStringSet(c("TTGAACGCACATTGCGCC", "ATCCCTACCTGATCCGAG"))
)
```

`select_one_sample` *Select one sample from a physeq object*

Description

Mostly for internal used, for example in function `track_wkflow_samples()`.

Usage

```
select_one_sample(physeq, sam_name, silent = FALSE)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`sam_name`	(required) The sample name to select
`silent`	(logical) If true, no message are printing.

Value

A new `phyloseq-class` object with one sample

Author(s)

Adrien Taudière

Examples

```
A8_005 <- select_one_sample(data_fungi, "A8-005_S4_MERGED.fastq.gz")
A8_005
```

`select_taxa` *Select a subset of taxa in a specified order where possible*

Description

Select (a subset of) taxa; if `x` allows taxa to be reordered, then taxa are given in the specified order.

Usage

```
select_taxa(x, taxa, reorder = TRUE)

## S4 method for signature 'sample_data,character'
select_taxa(x, taxa)

## S4 method for signature 'otu_table,character'
select_taxa(x, taxa, reorder = TRUE)

## S4 method for signature 'taxonomyTable,character'
select_taxa(x, taxa, reorder = TRUE)

## S4 method for signature 'XStringSet,character'
select_taxa(x, taxa, reorder = TRUE)

## S4 method for signature 'phylo,character'
select_taxa(x, taxa)

## S4 method for signature 'phyloseq,character'
select_taxa(x, taxa, reorder = TRUE)
```

Arguments

`x`	A phyloseq object or phyloseq component object
`taxa`	Character vector of taxa to select, in requested order
`reorder`	Logical specifying whether to use the order in `taxa` (TRUE) or keep the order in `taxa_names(x)` (FALSE)

Details

This is a simple selector function that is like `prune_taxa(taxa, x)` when `taxa` is a character vector but always gives the taxa in the order `taxa` if possible (that is, except for phy_tree's and phyloseq objects that contain phy_tree's).

Author(s)

Michael R. McLaren (orcid: 0000-0003-1575-473X)

`signif_ancombc` *Filter ancombc_pq results*

Description

Internally used in `plot_ancombc_pq()`.

Usage

```
signif_ancombc(
  ancombc_res,
  filter_passed = TRUE,
  filter_diff = TRUE,
  min_abs_lfc = 0
)
```

Arguments

`ancombc_res`	(required) the result of the ancombc_pq function For the moment only bimodal factors are possible.
`filter_passed`	(logical, default TRUE) Do we filter using the column passed_ss? The passed_ss value is TRUE if the taxon passed the sensitivity analysis, i.e., adding different pseudo-counts to 0s would not change the results.
`filter_diff`	(logical, default TRUE) Do we filter using the column diff? The diff value is TRUE if the taxon is significant (has q less than alpha)
`min_abs_lfc`	(integer, default0) Minimum absolute value to filter results based on Log Fold Change. For ex. a value of 1 filter out taxa for which the abundance in a given level of the modality is not at least the double of the abundance in the other level.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `ANCOMBC::ancombc2()` if you use this function.

Value

A data.frame with the same number of columns than the `ancombc_res` param but with less (or equal) numbers of rows

See Also

`ancombc_pq()`, `plot_ancombc_pq()`

Examples

```
## Not run:
if (requireNamespace("mia")) {
  data_fungi_mini@tax_table <- phyloseq::tax_table(cbind(
    data_fungi_mini@tax_table,
    "taxon" = taxa_names(data_fungi_mini)
  ))

  res_time <- ancombc_pq(
    data_fungi_mini,
    fact = "Time",
    levels_fact = c("0", "15"),
    tax_level = "taxon",
    verbose = TRUE
  )

  signif_ancombc(res_time)
}

## End(Not run)
```

`simplify_taxo` *Simplify taxonomy by removing some unused characters such as "k__"*

Description

Internally used in `clean_pq()`

Usage

```
simplify_taxo(
  physeq,
  pattern_to_remove = c(".__", ".*:"),
  remove_space = TRUE,
  remove_NA = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`pattern_to_remove`	(a vector of character) the pattern to remove using `base::gsub()` function.
`remove_space`	(logical; default TRUE): do we remove space?
`remove_NA`	(logical; default FALSE): do we remove NA (in majuscule)?

Value

A `phyloseq-class` object with simplified taxonomy

Author(s)

Adrien Taudière

Examples

```
d_fm <- data_fungi_mini
d_fm@tax_table[, "Species"] <- paste0(rep(
  c("s__", "s:"),
  ntaxa(d_fm) / 2
), d_fm@tax_table[, "Species"])

# First column is the new vector of Species,
# second column is the column before simplification
cbind(
  simplify_taxo(d_fm)@tax_table[, "Species"],
  d_fm@tax_table[, "Species"]
)
cbind(
  simplify_taxo(d_fm, remove_NA = TRUE)@tax_table[, "Species"],
  d_fm@tax_table[, "Species"]
)
```

`SRS_curve_pq` *Scaling with ranked subsampling (SRS) curve of phyloseq object*

Description

A wraper of `SRS::SRScurve()` function.

Usage

```
SRS_curve_pq(physeq, clean_pq = FALSE, ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`clean_pq`	(logical): Does the phyloseq object is cleaned using the `clean_pq()` function?
`...`	Additional arguments passed on to `SRS::SRScurve()`

Value

A plot

Examples

```
if (requireNamespace("SRS")) {
  SRS_curve_pq(data_fungi_mini,
    max.sample.size = 200,
    rarefy.comparison = TRUE, rarefy.repeats = 3
  )
  SRS_curve_pq(data_fungi_mini, max.sample.size = 500, metric = "shannon")
}
```

`subsample_fastq` *Subsample a fastq file copying the n_seq first sequences in a given folder*

Description

Useful to test a pipeline on small fastq files.

Usage

```
subsample_fastq(fastq_files, folder_output = "subsample", nb_seq = 1000)
```

Arguments

`fastq_files`	The path to one fastq file or a list of fastq files (see examples)
`folder_output`	The path to a folder for output files
`nb_seq`	(int; default 1000) : Number of sequences kept (every sequence spread across 4 lines)

Value

Nothing, create subsampled fastq files in a folder

Author(s)

Adrien Taudière

Examples

```
ex_file <- system.file("extdata", "ex_R1_001.fastq.gz",
  package = "MiscMetabar",
  mustWork = TRUE
)
subsample_fastq(ex_file, paste0(tempdir(), "/output_fastq"))
subsample_fastq(list_fastq_files(system.file("extdata", package = "MiscMetabar")),
  paste0(tempdir(), "/output_fastq"),
  n = 10
)
unlink(paste0(tempdir(), "/output_fastq"), recursive = TRUE)
```

`subset_samples_pq` *Subset samples using a conditional boolean vector.*

Description

The main objective of this function is to complete the `phyloseq::subset_samples()` function by propose a more easy (but more prone to error) way of subset_samples. It replace the subsetting expression which used the name of the variable in the sam_data by a boolean vector.

Warnings: you must verify the result of this function as the boolean condition must match the order of samples in the `sam_data` slot.

This function is robust when you use the sam_data slot of the phyloseq object used in physeq (see examples)

Usage

```
subset_samples_pq(physeq, condition)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`condition`	A boolean vector to subset samples. Length must fit the number of samples

Value

a new phyloseq object

Author(s)

Adrien Taudière

Examples

```
cond_samp <- grepl("A1", data_fungi@sam_data[["Sample_names"]])
subset_samples_pq(data_fungi, cond_samp)

subset_samples_pq(data_fungi, data_fungi@sam_data[["Height"]] == "Low")
```

`subset_taxa_pq` *Subset taxa using a conditional named boolean vector.*

Description

The main objective of this function is to complete the `phyloseq::subset_taxa()` function by propose a more easy way of subset_taxa using a named boolean vector. Names must match taxa_names.

Usage

```
subset_taxa_pq(
  physeq,
  condition,
  verbose = TRUE,
  clean_pq = TRUE,
  taxa_names_from_physeq = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`condition`	A named boolean vector to subset taxa. Length must fit the number of taxa and names must match taxa_names. Can also be a condition using a column of the tax_table slot (see examples). If the order of condition is the same as taxa_names(physeq), you can use the parameter `taxa_names_from_physeq = TRUE`.
`verbose`	(logical) Informations are printed
`clean_pq`	(logical) If set to TRUE, empty samples are discarded after subsetting ASV
`taxa_names_from_physeq`	(logical) If set to TRUE, rename the condition vector using taxa_names(physeq). Carefully check the result of this function if you use this parameter. No effect if the condition is of class `tax_table`.

Value

a new phyloseq object

Author(s)

Adrien Taudière

Examples

```
subset_taxa_pq(data_fungi, data_fungi@tax_table[, "Phylum"] == "Ascomycota")

cond_taxa <- grepl("Endophyte", data_fungi@tax_table[, "Guild"])
names(cond_taxa) <- taxa_names(data_fungi)
```

```
subset_taxa_pq(data_fungi, cond_taxa)

subset_taxa_pq(data_fungi, grepl("mycor", data_fungi@tax_table[, "Guild"]),
  taxa_names_from_physeq = TRUE
)
```

`subset_taxa_tax_control` *Subset taxa using a taxa control or distribution based method*

Description

There is 3 main methods : discard taxa (i) using a control taxa (e.g. truffle root tips), (ii) using a mixture models to detect bimodality in pseudo-abundance distribution or (iii) using a minimum difference threshold pseudo-abundance. Each cutoff is defined at the sample level.

Usage

```
subset_taxa_tax_control(
  physeq,
  taxa_distri,
  method = "mean",
  min_diff_for_cutoff = NULL
)
```

Arguments

`physeq` (required): a `phyloseq-class` object obtained using the `phyloseq` package.

`taxa_distri` (required) a vector of length equal to the number of samples with the number of sequences per sample for the taxa control

`method` (default: "mean") a method to calculate the cut-off value. There are 6 available methods:

1. `cutoff_seq`: discard taxa with less than the number of sequence than taxa control,
2. `cutoff_mixt`: using mixture models,
3. `cutoff_diff`: using a minimum difference threshold (need the argument min_diff_for_cutoff)
4. `min`: the minimum of the three firsts methods
5. `max`: the maximum of the three firsts methods
6. `mean`: the mean of the three firsts methods

`min_diff_for_cutoff` (int) argument for method `cutoff_diff`. Required if method is `cutoff_diff`, `min`, `max` or `mean`

Value

A new `phyloseq-class` object.

Author(s)

Adrien Taudière

Examples

```
subset_taxa_tax_control(data_fungi,
  as.numeric(data_fungi@otu_table[, 300]),
  min_diff_for_cutoff = 2
)
```

`summary_plot_pq` *Summarize a* `phyloseq-class` *object using a plot.*

Description

Graphical representation of a phyloseq object.

Usage

```
summary_plot_pq(
  physeq,
  add_info = TRUE,
  min_seq_samples = 500,
  clean_pq = TRUE,
  text_size = 1,
  text_size_info = 1
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`add_info`	Does the bottom down corner contain extra informations?
`min_seq_samples`	(int): Used only when add_info is set to true to print the number of samples with less sequences than this number.
`clean_pq`	(logical): Does the phyloseq object is cleaned using the `clean_pq()` function?
`text_size`	(Num, default 1) A size factor to expand or minimize text size.
`text_size_info`	(Num, default 1) A size factor to expand or minimize text size for extra informations.

Value

A ggplot2 object

Examples

```
summary_plot_pq(data_fungi)
summary_plot_pq(data_fungi, add_info = FALSE) + scale_fill_viridis_d()
if (requireNamespace("patchwork")) {
  (summary_plot_pq(data_fungi, text_size = 0.5, text_size_info = 0.6) +
    summary_plot_pq(data_fungi_mini, text_size = 0.5, text_size_info = 0.6)) /
    (summary_plot_pq(data_fungi_sp_known, text_size = 0.5, text_size_info = 0.6) +
      summary_plot_pq(subset_taxa(data_fungi_sp_known, Phylum == "Ascomycota"),
        text_size = 0.5, text_size_info = 0.6
      ))
}
```

`swarm_clustering` *Re-cluster sequences of an object of class* `physeq` *or cluster a list of DNA sequences using SWARM*

Description

A wrapper of SWARM software.

Usage

```
swarm_clustering(
  physeq = NULL,
  dna_seq = NULL,
  d = 1,
  swarmpath = "swarm",
  vsearch_path = "vsearch",
  nproc = 1,
  swarm_args = "--fastidious",
  tax_adjust = 0,
  keep_temporary_files = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`dna_seq`	NOT WORKING FOR THE MOMENT You may directly use a character vector of DNA sequences in place of physeq args. When physeq is set, dna sequences take the value of `physeq@refseq`
`d`	(default: 1) maximum number of differences allowed between two amplicons, meaning that two amplicons will be grouped if they have `d` (or less) differences
`swarmpath`	(default: swarm) path to swarm

`vsearch_path`	(default: vsearch) path to vsearch, used only if physeq is NULL and dna_seq is provided.
`nproc`	(default: 1) Set to number of cpus/processors to use for the clustering
`swarm_args`	(default : "–fastidious") a one length character element defining other parameters to passed on to swarm See other possible methods in the SWARM pdf manual
`tax_adjust`	(Default 0) See the man page of `merge_taxa_vec()` for more details. To conserved the taxonomic rank of the most abundant ASV,
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files ? • temp.fasta (refseq in fasta or dna_seq sequences) • temp_output (classical output of SWARM) • temp_uclust (clusters output of SWARM)

Details

This function use the `merge_taxa_vec` function to merge taxa into clusters. By default tax_adjust = 0. See the man page of `merge_taxa_vec()`.

This function is mainly a wrapper of the work of others. Please cite SWARM.

Value

A new object of class physeq or a list of cluster if dna_seq args was used.

References

SWARM can be downloaded from `https://github.com/torognes/swarm/`.

SWARM can be downloaded from `https://github.com/torognes/swarm`. More information in the associated publications doi:10.1093/bioinformatics/btab493 and doi:10.7717/peerj.593

See Also

`postcluster_pq()`, `vsearch_clustering()`

Examples

```
summary_plot_pq(data_fungi)
system2("swarm", "-h")

data_fungi_swarm <- swarm_clustering(data_fungi)
summary_plot_pq(data_fungi_swarm)

sequences_ex <- c(
  "TACCTATGTTGCCTTGGCGGCTAAACCTACCCGGGATTTGATGGGGCGAATTAATAACGAATTCATTGAATCA",
  "TACCTATGTTGCCTTGGCGGCTAAACCTACCCGGGATTTGATGGGGCGAATTACCTGGTAAGGCCCACTT",
  "TACCTATGTTGCCTTGGCGGCTAAACCTACCCGGGATTTGATGGGGCGAATTACCTGGTAGAGGTG",
  "TACCTATGTTGCCTTGGCGGCTAAACCTACC",
  "CGGGATTTGATGGCGAATTACCTGGTATTTTAGCCCACTTACCCGGTACCATGAGGTG",
```

```
    "GCGGCTAAACCTACCCGGGATTTGATGGCGAATTACCTGG",
    "GCGGCTAAACCTACCCGGGATTTGATGGCGAATTACAAAG",
    "GCGGCTAAACCTACCCGGGATTTGATGGCGAATTACAAAG",
    "GCGGCTAAACCTACCCGGGATTTGATGGCGAATTACAAAG"
  )

  sequences_ex_swarm <- swarm_clustering(
    dna_seq = sequences_ex
  )
```

`taxa_as_columns` *Force taxa to be in columns in the otu_table of a physeq object*

Description

Mainly for internal use. It is a special case of clean_pq function.

Usage

```
taxa_as_columns(physeq)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.

Value

A new `phyloseq-class` object

Author(s)

Adrien Taudière

`taxa_as_rows` *Force taxa to be in columns in the otu_table of a physeq object*

Description

Mainly for internal use. It is a special case of clean_pq function.

Usage

```
taxa_as_rows(physeq)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.

Value

A new phyloseq-class object

Author(s)

Adrien Taudière

`taxa_only_in_one_level`

Show taxa which are present in only one given level of a modality

---

Description

Given one modality name in sam_data and one level of the modality, return the taxa strictly specific of this level.

Usage

```
taxa_only_in_one_level(
  physeq,
  modality,
  level,
  min_nb_seq_taxa = 0,
  min_nb_samples_taxa = 0
)

taxa_only_in_one_level(
  physeq,
  modality,
  level,
  min_nb_seq_taxa = 0,
  min_nb_samples_taxa = 0
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`modality`	(required) The name of a column present in the `@sam_data` slot of the physeq object. Must be a character vector or a factor.
`level`	(required) The level (must be present in modality) of interest
`min_nb_seq_taxa`	(default 0 = no filter) The minimum number of sequences per taxa
`min_nb_samples_taxa`	(default 0 = no filter) The minimum number of samples per taxa

Value

A vector of taxa names

A vector of taxa names

Author(s)

Adrien Taudière

Examples

```
data_fungi_mini_woNA4height <- subset_samples(
  data_fungi_mini,
  !is.na(data_fungi_mini@sam_data$Height)
)
taxa_only_in_one_level(data_fungi_mini_woNA4height, "Height", "High")
#' # Taxa present only in low height samples
suppressMessages(suppressWarnings(
  taxa_only_in_one_level(data_fungi, "Height", "Low")
))
# Number of taxa present only in sample of time equal to 15
suppressMessages(suppressWarnings(
  length(taxa_only_in_one_level(data_fungi, "Time", "15"))
))
data_fungi_mini_woNA4height <- subset_samples(
  data_fungi_mini,
  !is.na(data_fungi_mini@sam_data$Height)
)
taxa_only_in_one_level(data_fungi_mini_woNA4height, "Height", "High")
#' # Taxa present only in low height samples
suppressMessages(suppressWarnings(
  taxa_only_in_one_level(data_fungi, "Height", "Low")
))
# Number of taxa present only in sample of time equal to 15
suppressMessages(suppressWarnings(
  length(taxa_only_in_one_level(data_fungi, "Time", "15"))
))
```

`tax_bar_pq` *Plot the distribution of sequences or ASV in one taxonomic levels*

Description

Graphical representation of distribution of taxonomy, optionnaly across a factor.

Usage

```
tax_bar_pq(
  physeq,
  fact = "Sample",
```

```
  taxa = "Order",
  percent_bar = FALSE,
  nb_seq = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`.
`taxa`	(default: 'Order') Name of the taxonomic rank of interest
`percent_bar`	(default FALSE) If TRUE, the stacked bar fill all the space between 0 and 1. It just set position = "fill" in the `ggplot2::geom_bar()` function
`nb_seq`	(logical; default TRUE) If set to FALSE, only the number of ASV is count. Concretely, physeq otu_table is transformed in a binary otu_table (each value different from zero is set to one)

Value

A `ggplot`2 plot with bar representing the number of sequence en each taxonomic groups

Author(s)

Adrien Taudière

See Also

`plot_tax_pq()` and `multitax_bar_pq()`

Examples

```
data_fungi_ab <- subset_taxa_pq(data_fungi, taxa_sums(data_fungi) > 10000)
tax_bar_pq(data_fungi_ab) + theme(legend.position = "none")

tax_bar_pq(data_fungi_ab, taxa = "Class")
tax_bar_pq(data_fungi_ab, taxa = "Class", percent_bar = TRUE)
tax_bar_pq(data_fungi_ab, taxa = "Class", fact = "Time")
```

`tax_datatable` *Make a datatable with the taxonomy of a* `phyloseq-class` *object*

Description

An interactive table for phyloseq taxonomy.

Usage

```
tax_datatable(
  physeq,
  abundance = TRUE,
  taxonomic_level = NULL,
  modality = NULL,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`abundance`	(default: TRUE) Does the number of sequences is print
`taxonomic_level`	(default: NULL) a vector of selected taxonomic level using their column numbers (e.g. taxonomic_level = 1:7)
`modality`	(default: NULL) A sample modality to split OTU abundancy by level of the modality
`...`	Other argument for the datatable function

Value

A datatable

Author(s)

Adrien Taudière

Examples

```
data("GlobalPatterns", package = "phyloseq")
if (requireNamespace("DT")) {
  tax_datatable(subset_taxa(
    GlobalPatterns,
    rowSums(GlobalPatterns@otu_table) > 10000
  ))

  # Using modality
  tax_datatable(GlobalPatterns,
    modality = GlobalPatterns@sam_data$SampleType
  )
}
```

`tbl_sum_samdata` *Summarize information from sample data in a table*

Description

A wrapper for the `gtsummary::tbl_summary()` function in the case of physeq object.

Usage

```
tbl_sum_samdata(physeq, remove_col_unique_value = TRUE, ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`remove_col_unique_value`	(logical, default TRUE) Do we remove informative columns (categorical column with one value per samples), e.g. samples names ?
`...`	Additional arguments passed on to `gtsummary::tbl_summary()`.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `gtsummary::tbl_summary()` if you use this function.

Value

A new `phyloseq-class` object with a larger slot tax_table

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("gtsummary")) {
  tbl_sum_samdata(data_fungi) %>%
    gtsummary::as_kable()

  summary_samdata <- tbl_sum_samdata(data_fungi,
    include = c("Time", "Height"),
    type = list(Time ~ "continuous2", Height ~ "categorical"),
    statistic = list(Time ~ c("{median} ({p25}, {p75})", "{min}, {max}"))
  )
}

data(enterotype)
if (requireNamespace("gtsummary")) {
  summary_samdata <- tbl_sum_samdata(enterotype)
```

```
  summary_samdata <- tbl_sum_samdata(enterotype, include = !contains("SampleId"))
}
```

`tbl_sum_taxtable` *Summarize a tax_table (taxonomic slot of phyloseq object) using gtsummary*

Description

Mainly a wrapper for the `gtsummary::tbl_summary()` function in the case of physeq object.

Usage

```
tbl_sum_taxtable(physeq, taxonomic_ranks = NULL, ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`taxonomic_ranks`	A list of taxonomic ranks we want to summarized.
`...`	Additional arguments passed on to `gtsummary::tbl_summary()`

Value

A table of class c('tbl_summary', 'gtsummary')

Author(s)

Adrien Taudière

Examples

```
tbl_sum_taxtable(data_fungi_mini)
data_fungi_mini |>
  filt_taxa_pq(min_occurence = 2) |>
  tbl_sum_taxtable(taxonomic_rank = c("Species", "Genus"))
```

`Tengeler2020_pq` *This tutorial explores the dataset from Tengeler et al. (2020) available in the* `mia` *package. obtained using* `mia::makePhyloseqFromTreeSE(Tengeler2020)`

Description

This is a phyloseq version of the Tengeler2020 dataset.

Usage

```
data(Tengeler2020_pq)
```

Format

A phyloseq object

Details

Tengeler2020 includes gut microbiota profiles of 27 persons with ADHD. A standard bioinformatic and statistical analysis done to demonstrate that altered microbial composition could be a driver of altered brain structure and function and concomitant changes in the animals behavior. This was investigated by colonizing young, male, germ-free C57BL/6JOlaHsd mice with microbiota from individuals with and without ADHD.

Tengeler, A.C., Dam, S.A., Wiesmann, M. et al. Gut microbiota from persons with attention-deficit/hyperactivity disorder affects the brain in mice. Microbiome 8, 44 (2020). https://microbiomejournal.biomedcentral.co
020-00816-x

`track_wkflow` *Track the number of reads (= sequences), samples and cluster (e.g. ASV) from various objects including dada-class and derep-class.*

Description

- List of fastq and fastg.gz files -> nb of reads and samples
- List of dada-class -> nb of reads, clusters (ASV) and samples
- List of derep-class -> nb of reads, clusters (unique sequences) and samples
- Matrix of samples x clusters (e.g. `otu_table`) -> nb of reads, clusters and samples
- Phyloseq-class -> nb of reads, clusters and samples

Usage

```
track_wkflow(
  list_of_objects,
  obj_names = NULL,
  clean_pq = FALSE,
  taxonomy_rank = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

`list_of_objects`	(required) a list of objects
`obj_names`	A list of names corresponding to the list of objects
`clean_pq`	(logical) If set to TRUE, empty samples and empty ASV are discarded before clustering.
`taxonomy_rank`	A vector of int. Define the column number of taxonomic rank `in physeq@tax_table` to compute the number of unique value. Default is NULL and do not compute values for any taxonomic rank
`verbose`	(logical) If true, print some additional messages.
`...`	Additional arguments passed on to `clean_pq()` function.

Value

The number of sequences, clusters (e.g. OTUs, ASVs) and samples for each object.

Author(s)

Adrien Taudière

See Also

`track_wkflow_samples()`

Examples

```
data(enterotype)
if (requireNamespace("pbapply")) {
  track_wkflow(list(data_fungi, enterotype), taxonomy_rank = c(3, 5))
  track_wkflow(list(
    "data FUNGI" = data_fungi,
    "fastq files forward" =
      unlist(list_fastq_files(system.file("extdata", package = "MiscMetabar"),
        paired_end = FALSE
      ))
  ))
}
```

`track_wkflow_samples` *Track the number of reads (= sequences), samples and cluster (e.g. ASV) for each sample*

Description

Contrary to `track_wkflow()`, only phyloseq object are possible. More information are available in the manual of the function `track_wkflow()`

Usage

```
track_wkflow_samples(list_pq_obj, ...)
```

Arguments

`list_pq_obj`	(required): a list of object passed on to `track_wkflow()` Only phyloseq object will return value because information of sample is needed
`...`	Other args passed on to `track_wkflow()`

Value

A list of dataframe. cf `track_wkflow()` for more information

Author(s)

Adrien Taudière

Examples

```
tree_A10_005 <- subset_samples(data_fungi, Tree_name == "A10-005")
if (requireNamespace("pbapply")) {
  track_wkflow_samples(tree_A10_005)
}
```

`transp` *Adds transparency to a vector of colors*

Description

Adds transparency to a vector of colors

Usage

```
transp(col, alpha = 0.5)
```

Arguments

`col`	a vector of colors
`alpha`	(default 0.5) a numeric value between 0 and 1 representing the alpha coefficient; 0: total transparency; 1: no transparency.

Value

a color vector

Author(s)

Thibaut Jombart in `adegenet` package

See Also

The R package RColorBrewer, proposing a nice selection of color palettes. The viridis package, with many excellent palettes

---

`treemap_pq` *Plot treemap of 2 taxonomic levels*

Description

Note that lvl2need to be nested in lvl1

Usage

```
treemap_pq(
  physeq,
  lvl1,
  lvl2,
  nb_seq = TRUE,
  log10trans = TRUE,
  plot_legend = FALSE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`lvl1`	(required) Name of the first (higher) taxonomic rank of interest
`lvl2`	(required) Name of the second (lower) taxonomic rank of interest
`nb_seq`	(logical; default TRUE) If set to FALSE, only the number of ASV is count. Concretely, physeq otu_table is transformed in a binary otu_table (each value different from zero is set to one)

`log10trans`	(logical, default TRUE) If TRUE, the number of sequences (or ASV if nb_seq = FALSE) is log10 transformed.
`plot_legend`	(logical, default FALSE) If TRUE, plot che legend of color for lvl 1
`...`	Additional arguments passed on to `treemapify::geom_treemap()` function.

Value

A ggplot2 object

Author(s)

Adrien Taudière

Examples

```
data(data_fungi_sp_known)
if (requireNamespace("treemapify")) {
  treemap_pq(
    clean_pq(subset_taxa(
      data_fungi_sp_known,
      Phylum == "Basidiomycota"
    )),
    "Order", "Class",
    plot_legend = TRUE
  )
}

if (requireNamespace("treemapify")) {
  treemap_pq(
    clean_pq(subset_taxa(
      data_fungi_sp_known,
      Phylum == "Basidiomycota"
    )),
    "Order", "Class",
    log10trans = FALSE
  )
  treemap_pq(
    clean_pq(subset_taxa(
      data_fungi_sp_known,
      Phylum == "Basidiomycota"
    )),
    "Order", "Class",
    nb_seq = FALSE, log10trans = FALSE
  )
}
```

`tsne_pq` *Compute tSNE position of samples from a phyloseq object*

Description

Compute tSNE position of samples from a phyloseq object

Usage

```
tsne_pq(physeq, method = "bray", dims = 2, theta = 0, perplexity = 30, ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`method`	A method to calculate distance using `vegan::vegdist()` function
`dims`	(Int) Output dimensionality (default: 2)
`theta`	(Numeric) Speed/accuracy trade-off (increase for less accuracy), set to 0.0 for exact TSNE (default: 0.0 see details in the man page of `Rtsne::Rtsne`).
`perplexity`	(Numeric) Perplexity parameter (should not be bigger than 3 * perplexity < nrow(X) - 1, see details in the man page of `Rtsne::Rtsne`)
`...`	Additional arguments passed on to `Rtsne::Rtsne()`

Value

A list of element including the matrix Y containing the new representations for the objects. See ?Rtsne::Rtsne() for more information

Examples

```
if (requireNamespace("Rtsne")) {
  res_tsne <- tsne_pq(data_fungi)
}
```

`umap_pq` *Computes a manifold approximation and projection (UMAP) for phyloseq object*

Description

https://journals.asm.org/doi/full/10.1128/msystems.00691-21

Usage

```
umap_pq(physeq, pkg = "umap", ...)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`pkg`	Which R packages to use, either "umap" or "uwot".
`...`	Additional arguments passed on to `umap::umap()` or `uwot::umap2()` function. For example `n_neighbors` set the number of nearest neighbors (Default 15). See `umap::umap.defaults()` or `uwot::umap2()` for the list of parameters and default values.

Details

This function is mainly a wrapper of the work of others. Please make a reference to `umap::umap()` if you use this function.

Value

A dataframe with samples informations and the x_umap and y_umap position

Author(s)

Adrien Taudière

See Also

`umap::umap()`, `tsne_pq()`, `phyloseq::plot_ordination()`

Examples

```
library("umap")
df_umap <- umap_pq(data_fungi_mini)
ggplot(df_umap, aes(x = x_umap, y = y_umap, col = Height)) +
  geom_point(size = 2)

# library(patchwork)
# physeq <- data_fungi_mini
# res_tsne <- tsne_pq(data_fungi_mini)
# df_umap_tsne <- df_umap
# df_umap_tsne$x_tsne <- res_tsne$Y[, 1]
# df_umap_tsne$y_tsne <- res_tsne$Y[, 2]
# ((ggplot(df_umap, aes(x = x_umap, y = y_umap, col = Height)) +
#  geom_point(size = 2) +
#  ggtitle("UMAP")) + (plot_ordination(physeq,
#  ordination =
#    ordinate(physeq, method = "PCoA", distance = "bray"), color = "Height"
# )) +
#  ggtitle("PCoA")) /
#  ((ggplot(df_umap_tsne, aes(x = x_tsne, y = y_tsne, col = Height)) +
#    geom_point(size = 2) +
#    ggtitle("tsne")) +
#    (plot_ordination(physeq,
#      ordination = ordinate(physeq, method = "NMDS", distance = "bray"),
```

```
#      color = "Height"
#    ) +
#      ggtitle("NMDS"))) +
#   patchwork::plot_layout(guides = "collect")

# df_uwot <- umap_pq(data_fungi_mini, pkg = "uwot")

#   (ggplot(df_umap, aes(x = x_umap, y = y_umap, col = Height)) +
#     geom_point(size = 2) +
#    ggtitle("umap::umap")) /
#     (ggplot(df_uwot, aes(x = x_umap, y = y_umap, col = Height)) +
#       geom_point(size = 2) +
#       ggtitle("uwot::umap2"))
```

`unique_or_na` *Get the unique value in x or NA if none*

Description

If `unique(x)` is a single value, return it; otherwise, return an NA of the same type as `x`. If `x` is a factor, then the levels and ordered status will be kept in either case. If `x` is a non-atomic vector (i.e. a list), then the logical `NA` will be used.

Usage

```
unique_or_na(x)
```

Arguments

`x`	A vector

Value

Either a single value (if `unique(x)` return a single value) or a NA

Author(s)

Michael R. McLaren (orcid: 0000-0003-1575-473X)

Examples

```
f <- factor(c("a", "a", "b", "c"), ordered = TRUE)
unique_or_na(f)
unique_or_na(f[1:2])

x <- c("a", "b", "a")
unique_or_na(x[c(1, 3)])
unique_or_na(x)
unique_or_na(x) %>% typeof()
```

`upset_pq` *Make upset plot for phyloseq object.*

Description

Alternative to venn plot.

Usage

```
upset_pq(
  physeq,
  fact,
  taxa_fill = NULL,
  min_nb_seq = 0,
  na_remove = TRUE,
  numeric_fonction = sum,
  rarefy_after_merging = FALSE,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required): Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`.
`taxa_fill`	(default NULL) fill the ASV upset using a column in `tax_table` slot.
`min_nb_seq`	minimum number of sequences by OTUs by samples to take into count this OTUs in this sample. For example, if min_nb_seq=2,each value of 2 or less in the OTU table will not count in the venn diagram
`na_remove`	: if TRUE (the default), NA values in fact are removed if FALSE, NA values are set to "NA"
`numeric_fonction`	(default : sum) the function for numeric vector useful only for complex plot (see examples)
`rarefy_after_merging`	Rarefy each sample after merging by the modalities of `fact` parameter
`...`	Additional arguments passed on to the `ComplexUpset::upset()`

Value

A `ggplot`2 plot

Author(s)

Adrien Taudière

See Also

`ggvenn_pq()`

Examples

```
if (requireNamespace("ComplexUpset")) {
  upset_pq(data_fungi_mini,
    fact = "Height", width_ratio = 0.2,
    taxa_fill = "Class"
  )
}

if (requireNamespace("ComplexUpset")) {
  upset_pq(data_fungi_mini, fact = "Height", min_nb_seq = 1000)
  upset_pq(data_fungi_mini, fact = "Height", na_remove = FALSE)

 upset_pq(data_fungi_mini, fact = "Time", width_ratio = 0.2, rarefy_after_merging = TRUE)

  upset_pq(
    data_fungi_mini,
    fact = "Time",
    width_ratio = 0.2,
    annotations = list(
      "Sequences per ASV \n (log10)" = (
        ggplot(mapping = aes(y = log10(Abundance)))
        +
          geom_jitter(aes(
            color =
              Abundance
          ), na.rm = TRUE)
          +
          geom_violin(alpha = 0.5, na.rm = TRUE) +
          theme(legend.key.size = unit(0.2, "cm")) +
          theme(axis.text = element_text(size = 12))
      ),
      "ASV per phylum" = (
        ggplot(mapping = aes(fill = Phylum))
        +
          geom_bar() +
          ylab("ASV per phylum") +
          theme(legend.key.size = unit(0.2, "cm")) +
          theme(axis.text = element_text(size = 12))
      )
    )
  )

  upset_pq(
    data_fungi_mini,
    fact = "Time",
    width_ratio = 0.2,
    numeric_fonction = mean,
    annotations = list(
```

```
    "Sequences per ASV \n (log10)" = (
      ggplot(mapping = aes(y = log10(Abundance)))
      +
        geom_jitter(aes(
          color =
            Abundance
        ), na.rm = TRUE)
        +
        geom_violin(alpha = 0.5, na.rm = TRUE) +
        theme(legend.key.size = unit(0.2, "cm")) +
        theme(axis.text = element_text(size = 12))
    ),
    "ASV per phylum" = (
      ggplot(mapping = aes(fill = Phylum))
      +
        geom_bar() +
        ylab("ASV per phylum") +
        theme(legend.key.size = unit(0.2, "cm")) +
        theme(axis.text = element_text(size = 12))
    )
  )
)

upset_pq(
  subset_taxa(data_fungi_mini, Phylum == "Basidiomycota"),
  fact = "Time",
  width_ratio = 0.2,
  base_annotations = list(),
  annotations = list(
    "Sequences per ASV \n (log10)" = (
      ggplot(mapping = aes(y = log10(Abundance)))
      +
        geom_jitter(aes(
          color =
            Abundance
        ), na.rm = TRUE)
        +
        geom_violin(alpha = 0.5, na.rm = TRUE) +
        theme(legend.key.size = unit(0.2, "cm")) +
        theme(axis.text = element_text(size = 12))
    ),
    "ASV per phylum" = (
      ggplot(mapping = aes(fill = Class))
      +
        geom_bar() +
        ylab("ASV per Class") +
        theme(legend.key.size = unit(0.2, "cm")) +
        theme(axis.text = element_text(size = 12))
    )
  )
)

data_fungi2 <- data_fungi_mini
```

```
  data_fungi2@sam_data[["Time_0"]] <- data_fungi2@sam_data$Time == 0
  data_fungi2@sam_data[["Height__Time_0"]] <-
    paste0(data_fungi2@sam_data[["Height"]], "__", data_fungi2@sam_data[["Time_0"]])
 data_fungi2@sam_data[["Height__Time_0"]][grepl("NA", data_fungi2@sam_data[["Height__Time_0"]])] <-
    NA
  upset_pq(data_fungi2, fact = "Height__Time_0", width_ratio = 0.2, min_size = 2)
}
```

`upset_test_pq` *Test for differences between intersections*

Description

See `upset_pq()` to plot upset.

Usage

```
upset_test_pq(
  physeq,
  fact,
  var_to_test = "OTU",
  min_nb_seq = 0,
  na_remove = TRUE,
  numeric_fonction = sum,
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`fact`	(required): Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`.
`var_to_test`	(default c("OTU")) : a vector of column present in the tax_table slot from the physeq object
`min_nb_seq`	minimum number of sequences by OTUs by samples to take into count this OTUs in this sample. For example, if min_nb_seq=2,each value of 2 or less in the OTU table will not count in the venn diagram
`na_remove`	: if TRUE (the default), NA values in fact are removed if FALSE, NA values are set to "NA"
`numeric_fonction`	(default : sum) the function for numeric vector useful only for complex plot (see examples)
`...`	Additional arguments passed on to the `ComplexUpset::upset_test()`

Value

A ggplot2 plot

Author(s)

Adrien Taudière

See Also

`upset_pq()`

Examples

```
data(data_fungi)
if (requireNamespace("ComplexUpset")) {
  upset_test_pq(data_fungi, "Height", var_to_test = c("OTU", "Class", "Guild"))
  upset_test_pq(data_fungi, "Time")
}
```

`var_par_pq` *Partition the Variation of a phyloseq object by 2, 3, or 4 Explanatory Matrices*

Description

The function partitions the variation in otu_table using distance (Bray per default) with respect to two, three, or four explanatory tables, using adjusted R^2 in redundancy analysis ordination (RDA) or distance-based redundancy analysis. If response is a single vector, partitioning is by partial regression. Collinear variables in the explanatory tables do NOT have to be removed prior to partitioning. See `vegan::varpart()` for more information.

Usage

```
var_par_pq(
  physeq,
  list_component,
  dist_method = "bray",
  dbrda_computation = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`list_component`	(required) A named list of 2, 3 or four vectors with names from the `@sam_data` slot.
`dist_method`	(default "bray") the distance used. See `phyloseq::distance()` for all available distances or run `phyloseq::distanceMethodList()`. For "aitchison" and "robust.aitchison" distance, `vegan::vegdist()` function is directly used.

`dbrda_computation`
(logical) Do dbrda computations are runned for each individual component (each name of the list component) ?

Details

This function is mainly a wrapper of the work of others. Please make a reference to `vegan::varpart()` if you use this function.

Value

an object of class "varpart", see `vegan::varpart()`

Author(s)

Adrien Taudière

See Also

`var_par_rarperm_pq()`, `vegan::varpart()`, `plot_var_part_pq()`

Examples

```
if (requireNamespace("vegan")) {
  data_fungi_woNA <-
    subset_samples(data_fungi, !is.na(Time) & !is.na(Height))
  res_var <- var_par_pq(data_fungi_woNA,
    list_component = list(
      "Time" = c("Time"),
      "Size" = c("Height", "Diameter")
    ),
    dbrda_computation = TRUE
  )
}
```

`var_par_rarperm_pq` *Partition the Variation of a phyloseq object with rarefaction permutations*

Description

This is an extension of the function `var_par_pq()`. The main addition is the computation of nperm permutations with rarefaction even depth by sample. The return object

Usage

```
var_par_rarperm_pq(
  physeq,
  list_component,
  dist_method = "bray",
  nperm = 99,
  quantile_prob = 0.975,
  dbrda_computation = FALSE,
  dbrda_signif_pval = 0.05,
  sample.size = min(sample_sums(physeq)),
  verbose = FALSE,
  progress_bar = TRUE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`list_component`	(required) A named list of 2, 3 or four vectors with names from the `@sam_data` slot.
`dist_method`	(default "bray") the distance used. See `phyloseq::distance()` for all available distances or run `phyloseq::distanceMethodList()`. For aitchison and robust.aitchison distance, `vegan::vegdist()` function is directly used.#' @param fill_bg
`nperm`	(int) The number of permutations to perform.
`quantile_prob`	(float, `[0:1]`) the value to compute the quantile. Minimum quantile is compute using 1-quantile_prob.
`dbrda_computation`	(logical) Do dbrda computations are runned for each individual component (each name of the list component) ?
`dbrda_signif_pval`	(float, `[0:1]`) The value under which the dbrda is considered significant.
`sample.size`	(int) A single integer value equal to the number of reads being simulated, also known as the depth. See `phyloseq::rarefy_even_depth()`.
`verbose`	(logical). If TRUE, print additional information.
`progress_bar`	(logical, default TRUE) Do we print progress during the calculation?

Details

This function is mainly a wrapper of the work of others. Please make a reference to `vegan::varpart()` if you use this function.

Value

A list of class varpart with additional information in the `$part$indfract` part. Adj.R.square is the mean across permutation. Adj.R.squared_quantil_min and Adj.R.squared_quantil_max represent the quantile values of adjusted R squared

Author(s)

Adrien Taudière

See Also

`var_par_pq()`, `vegan::varpart()`, `plot_var_part_pq()`

Examples

```
if (requireNamespace("vegan")) {
  data_fungi_woNA <- subset_samples(data_fungi, !is.na(Time) & !is.na(Height))
  res_var_9 <- var_par_rarperm_pq(
    data_fungi_woNA,
    list_component = list(
      "Time" = c("Time"),
      "Size" = c("Height", "Diameter")
    ),
    nperm = 9,
    dbrda_computation = TRUE
  )
  res_var_2 <- var_par_rarperm_pq(
    data_fungi_woNA,
    list_component = list(
      "Time" = c("Time"),
      "Size" = c("Height", "Diameter")
    ),
    nperm = 2,
    dbrda_computation = TRUE
  )
}
```

`venn_pq` *Venn diagram of* `phyloseq-class` *object*

Description

Graphical representation of distribution of taxa across combined modality of a factor.

Usage

```
venn_pq(physeq, fact, min_nb_seq = 0, print_values = TRUE)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`fact`	(required): Name of the factor to cluster samples by modalities. Need to be in `physeq@sam_data`.

`min_nb_seq`	(default: 0)): minimum number of sequences by OTUs by samples to take into count this OTUs in this sample. For example, if min_nb_seq=2,each value of 2 or less in the OTU table will be change into 0 for the analysis
`print_values`	(logical) Print (or not) the table of number of OTUs for each combination. If print_values is TRUE the object is not a ggplot object. Please use print_values = FALSE if you want to add ggplot function (cf example).

Value

A `ggplot`2 plot representing Venn diagram of modalities of the argument `factor`

Author(s)

Adrien Taudière

See Also

`venneuler`

Examples

```
if (requireNamespace("venneuler")) {
  data("enterotype")
  venn_pq(enterotype, fact = "SeqTech")
}

if (requireNamespace("venneuler")) {
  venn_pq(enterotype, fact = "ClinicalStatus")
  venn_pq(enterotype, fact = "Nationality", print_values = FALSE)
  venn_pq(enterotype, fact = "ClinicalStatus", print_values = FALSE) +
    scale_fill_hue()
  venn_pq(enterotype, fact = "ClinicalStatus", print_values = FALSE) +
    scale_fill_hue()
}
```

`verify_pq` *Verify the validity of a phyloseq object*

Description

Mostly for internal use in MiscMetabar functions.

Usage

```
verify_pq(
  physeq,
  verbose = FALSE,
  min_nb_seq_sample = 500,
  min_nb_seq_taxa = 1
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`verbose`	(logical, default FALSE) If TRUE, prompt some warnings.
`min_nb_seq_sample`	(numeric) Only used if verbose = TRUE. Minimum number of sequences per samples to not show warning.
`min_nb_seq_taxa`	(numeric) Only used if verbose = TRUE. Minimum number of sequences per taxa to not show warning.

Value

Nothing if the phyloseq object is valid. An error in the other case. Warnings if verbose = TRUE

Author(s)

Adrien Taudière

`vsearch_clustering` *Recluster sequences of an object of class* `physeq` *or cluster a list of DNA sequences using vsearch software*

Description

A wrapper of VSEARCH software.

Usage

```
vsearch_clustering(
  physeq = NULL,
  dna_seq = NULL,
  nproc = 1,
  id = 0.97,
  vsearchpath = "vsearch",
  tax_adjust = 0,
  vsearch_cluster_method = "--cluster_size",
  vsearch_args = "--strand both",
  keep_temporary_files = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`dna_seq`	You may directly use a character vector of DNA sequences in place of physeq args. When physeq is set, dna sequences take the value of `physeq@refseq`
`nproc`	(default: 1) Set to number of cpus/processors to use for the clustering
`id`	(default: 0.97) level of identity to cluster
`vsearchpath`	(default: "vsearch") path to vsearch
`tax_adjust`	(Default 0) See the man page of `merge_taxa_vec()` for more details. To conserved the taxonomic rank of the most abundant ASV, set tax_adjust to 0 (default). For the moment only tax_adjust = 0 is robust
`vsearch_cluster_method`	(default: "–cluster_size) See other possible methods in the vsearch manual (e.g. `--cluster_size` or `--cluster_smallmem`) • `--cluster_fast` : Clusterize the fasta sequences in filename, automatically sort by decreasing sequence length beforehand. • `--cluster_size` : Clusterize the fasta sequences in filename, automatically sort by decreasing sequence abundance beforehand. • `--cluster_smallmem` : Clusterize the fasta sequences in filename without automatically modifying their order beforehand. Sequence are expected to be sorted by decreasing sequence length, unless *–usersort* is used. In that case you may set `vsearch_args` to vsearch_args = "–strand both –usersort"
`vsearch_args`	(default : "–strand both") a one length character element defining other parameters to passed on to vsearch.
`keep_temporary_files`	(logical, default: FALSE) Do we keep temporary files ? • temp.fasta (refseq in fasta or dna_seq sequences) • cluster.fasta (centroid if method = "vsearch") • temp.uc (clusters if method = "vsearch")

Details

This function use the `merge_taxa_vec()` function to merge taxa into clusters. By default tax_adjust = 0. See the man page of `merge_taxa_vec()`.

This function is mainly a wrapper of the work of others. Please cite vsearch.

Value

A new object of class `physeq` or a list of cluster if dna_seq args was used.

Author(s)

Adrien Taudière

References

VSEARCH can be downloaded from https://github.com/torognes/vsearch. More information in the associated publication https://pubmed.ncbi.nlm.nih.gov/27781170.

See Also

`postcluster_pq()`, `swarm_clustering()`

Examples

```
summary_plot_pq(data_fungi)
d_vs <- vsearch_clustering(data_fungi)
summary_plot_pq(d_vs)
```

`vs_search_global` *Search for a list of sequence in a fasta file against physeq reference sequences using Rhrefhttps://github.com/torognes/vsearchvsearch*

Description

Use of VSEARCH software.

Usage

```
vs_search_global(
  physeq,
  path_to_fasta = NULL,
  seq2search = NULL,
  vsearchpath = "vsearch",
  id = 0.8,
  iddef = 0,
  keep_temporary_files = FALSE
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the phyloseq package.
`path_to_fasta`	(required if seq2search is NULL) a path to fasta file if seq2search is est to NULL.
`seq2search`	(required if path_to_fasta is NULL) Either (i) a DNAStringSet object or (ii) a character vector that will be convert to DNAStringSet using `Biostrings::DNAStringSet()`
`vsearchpath`	(default: "vsearch") path to vsearch
`id`	(default: 0.8) id for the option `--usearch_global` of the vsearch software
`iddef`	(default: 0) iddef for the option `--usearch_global` of the vsearch software

`keep_temporary_files`
(logical, default: FALSE) Do we keep temporary files

- temp.fasta (refseq in fasta)
- cluster.fasta (centroid)
- temp.uc (clusters)

Details

This function is mainly a wrapper of the work of others. Please cite vsearch.

Value

A dataframe with uc results (invisible)

Author(s)

Adrien Taudière

Examples

```
if (requireNamespace("seqinr")) {
  file_dna <- tempfile("dna.fa")
  seqinr::write.fasta("GCCCATTAGTATTCTAGTGGGCATGCCTGTTCGAGCGTCATTTTCAACC",
    file = file_dna, names = "seq1"
  )

  res <- vs_search_global(data_fungi, path_to_fasta = file_dna)
  unlink(file_dna)

  res[res$identity != "*", ]

  clean_pq(subset_taxa(data_fungi, res$identity != "*"))
}
```

`write_pq` *Save phyloseq object in the form of multiple csv tables.*

Description

This is the reverse function of `read_pq()`.

Usage

```
write_pq(
  physeq,
  path = NULL,
  rdata = FALSE,
  one_file = FALSE,
  write_sam_data = TRUE,
  sam_data_first = FALSE,
  clean_pq = TRUE,
  reorder_taxa = FALSE,
  rename_taxa = FALSE,
  remove_empty_samples = TRUE,
  remove_empty_taxa = TRUE,
  clean_samples_names = TRUE,
  silent = FALSE,
  verbose = FALSE,
  quote = FALSE,
  sep_csv = "\t",
  ...
)
```

Arguments

`physeq`	(required): a `phyloseq-class` object obtained using the `phyloseq` package.
`path`	a path to the folder to save the phyloseq object
`rdata`	(logical) does the phyloseq object is also saved in Rdata format?
`one_file`	(logical) if TRUE, combine all data in one file only
`write_sam_data`	(logical) does the samples data are add to the file. Only used if `one_file` is TRUE. Note that these option result in a lot of NA values.
`sam_data_first`	(logical) if TRUE, put the sample data at the top of the table Only used if `one_file` and write_sam_data are both TRUE.
`clean_pq`	(logical) If set to TRUE, empty samples are discarded after subsetting taxa (ASV, OTU, ...)
`reorder_taxa`	(logical) if TRUE the otu_table is ordered by the number of sequences of taxa (ASV, OTU, ...) (descending order). Default to TRUE. Only possible if clean_pq is set to TRUE.
`rename_taxa`	reorder_taxa (logical) if TRUE, taxa (ASV, OTU, ...) are renamed by their position in the OTU_table (asv_1, asv_2, ...). Default to FALSE. Only possible if clean_pq is set to TRUE.
`remove_empty_samples`	(logical) Do you want to remove samples without sequences (this is done after removing empty taxa)
`remove_empty_taxa`	(logical) Do you want to remove taxa without sequences (this is done before removing empty samples)

`clean_samples_names`	(logical) Do you want to clean samples names?
`silent`	(logical) If true, no message are printing.
`verbose`	(logical) Additional informations in the message the verbose parameter overwrite the silent parameter.
`quote`	a logical value (default FALSE) or a numeric vector. If TRUE, any character or factor columns will be surrounded by double quotes. If a numeric vector, its elements are taken as the indices of columns to quote. In both cases, row and column names are quoted if they are written. If FALSE nothing is quoted.
`sep_csv`	(default tabulation) separator for column
`...`	Additional arguments passed on to `utils::write.table()` function.

Value

Build a folder (path) containing one to four csv tables (refseq.csv, otu_table.csv, tax_table.csv, sam_data.csv) and if present a phy_tree in Newick format

Author(s)

Adrien Taudière

See Also

`save_pq()`

Examples

```
write_pq(data_fungi, path = paste0(tempdir(), "/phyloseq"))
write_pq(data_fungi, path = paste0(tempdir(), "/phyloseq"), one_file = TRUE)
unlink(paste0(tempdir(), "/phyloseq"), recursive = TRUE)
```

Index