Package 'QsRutils'

March 31, 2026

Type Package

Title R Functions Useful for Community Ecology

Version 0.2.0

Date 2025-10-13

Author John Quensen

Maintainer John Quensen <quensenj@msu.edu>

Description Some functions I have written that simplify community analyses by vegan, phyloseq, etc.

Depends phyloseq,
ggplot2,
vegan (>= 2.4-6),
R (>= 4.1.0)

Imports agricolae,
ape,
Biostrings,
dada2,
data.table,
dplyr,
ggplot2,
insect,
multcompView,
magrittr,
ShortRead,
SRS,
stringr,
tibble

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Suggests dunn.test,
knitr,
rmarkdown,
reshape2,
testthat (>= 3.0.0)

VignetteBuilder knitr

URL https://github.com/jfq3/QsRutils

Config/testthat/edition 3

Contents

`arc_sine` *arc_sine*

Description

Arcsine of a Percentage

Usage

```
arc_sine(x)
```

Arguments

`x`	A percentage.

Value

The arcsine transformation of x.

Examples

```
arc_sine(30.1)
```

`asterix` *Indicate Significance with Stars*

Description

Indicate Significance with Stars

Usage

```
asterix(prob)
```

Arguments

`prob`	p value

Details

Returns '***; for p < 0.001, '**' for p < 0.01, '*' for p < 0.05.

Value

Character vector of asterisks indicating significance level.

Examples

```
asterix(0.039)
```

avg_alpha *Average Alpha Diversity (faster implementation)*

Description

Calculates alpha-diversity metrics from n samplings of an OTU table to a constant number of counts per sample.

Usage

```
avg_alpha(
  otu,
  sampling_depth,
  iterations = 100,
  sum_method = c("median", "mean"),
  ncores = 1
)
```

Arguments

otu	An OTU table as a data frame or matrix with samples as rows and taxa as columns.
sampling_depth	The number of counts per sample in the sampled OTU table
iterations	The number of times the OTU table should be sampled.
sum_method	Method ("median" or "mean") for summarizing replication results.
ncores	Number of cores to use for parallel execution. Default 1 (no parallelism).

Details

This implementation focuses on speed: - vectorizes the alpha-metric calculations (avoids repeated calls to vegan::diversity and vegan::specnumber) - uses base R operations (rowSums, logical ops, arithmetic) which are much faster in tight loops - optionally parallelizes replicates across cores (platform-aware).

The OTU data frame supplied must be in typical vegan format: samples as row names and taxa as column names. The minimum row sum must be greater than or equal to the sampling depth.

By default the sum_method is mean. For a similar function in QIIME2, the default sum_method is median.

Value

Returns a dataframe with Shannon, Observed, Pielou, Simpson and Inverse Simpson for each sample in an OTU table.

Examples

```
{
data(BCI)
otu <- BCI[rowSums(BCI) > 400, ]
avg_alpha(otu, sampling_depth = 400, iterations = 100)
}
```

check_primer_hits *Check Primer Hits*

Description

Determine hits of all orientations of primers to paired sequence files.

Usage

```
check_primer_hits(
  path,
  fwd_pattern = "_R1.fastq",
  rev_pattern = "_R2.fastq",
  fwd_primer = "GGAAGTAAAAGTCGTAACAAGG",
  rev_primer = "GCTGCGTTCTTCATCGATGC"
)
```

Arguments

path	Path to paired sequence files
fwd_pattern	Portion of file name that distinguishes forward read files. The default is "_R1.fastq"
rev_pattern	Portion of the file name that distinguishes the reverse file. The default is "_R2.fastq"
fwd_primer	Nucleotide sequence of the forward primer
rev_primer	Nucleotide sequence of the reverse primet

Details

This function is for checking the effectiveness of primer trimming of ITS sequences. Because the ITS region varies in length, it is possible for forward sequences to extend past the reverse primenr region and vice-versa, leadsing to what Robert Edgar calls staggered pairs fi the sequences are merged.

The fwd_pattern and rev_pattern must contain the file extension. The defaults are "_R1.fastq" and "_R2.fastq".

Default primers are ITS5 (forward) and ITS2 (reverse) from White et.al 1990.

ITS5: "GGAAGTAAAAGTCGTAACAAGG"

ITS2: "GCTGCGTTCTTCATCGATGC"

Value

A table of hits to the sequences by all primer orientations

Examples

```
# This takes a while
path <- system.file("extdata", package = "QsRutils")
check_primer_hits(
path = path,
fwd_pattern = "raw_1.fastq.gz",
  rev_pattern = "raw_2.fastq.gz",
fwd_primer = "GGAAGTAAAAGTCGTAACAAGG",
```

```
rev_primer = "GCTGCGTTCTTCATCGATGC"
)
```

---

check_var *Check Variance*

Description

Tests for Heterogeneity of Variances in make_comparisons Result

Usage

```
check_var(otu.pc.transformed, group.vector)
```

Arguments

otu.pc.transformed	An OTU matrix of transformed data. Taxa are rows.
group.vector	A vector of treatments.

Value

Prints test results to the console.

See Also

make_comparisons

Examples

```
# Transform species matrix to proportion;
# Check variances for the first three species
# in the dune data set grouped by Management.
data(dune)
data(dune.env)
dune <- vegan::decostand(dune, method = "total")
dune <- dune[, 1:3]
dune <- t(dune)
check_var(dune, dune.env$Management)
```

cld_dunn *CLDs from DUNN*

Description

Generate compact letter displays from Dunn.test results

Usage

```
cld_dunn(dunn_rslt, significance = 0.05)
```

Arguments

dunn_rslt	The result of the function dunn.test::dunn.test
significance	The alpha level for statistical significance

Value

A character vector of CLDs - groups of significantly different treatment groups

Examples

```
# Example cribbed and modified from the kruskal.test documentation
x <- c(2.9, 3.0, 2.5, 2.6, 3.2) # normal subjects
y <- c(3.8, 2.7, 4.0, 2.4)      # with obstructive airway disease
z <- c(2.8, 3.4, 3.7, 2.2, 2.0) # with asbestosis
x <- c(x, y, z)
g <- factor(rep(1:3, c(5, 4, 5)),
            labels = c("Normal",
                       "COPD",
                       "Asbestosis"))
dunn_rslt <- dunn.test::dunn.test(x, g)
cld_dunn(dunn_rslt, significance = 0.05)
```

cld_hsd *Make CLD tibble from Tukey HSD Results*

Description

Makes a tibble for adding compact letter assignments to a boxplot using HSD.test results.

Usage

```
cld_hsd(hsd_rslt, y_pos = "boxtop")
```

Arguments

hsd_rslt	The result of the HSD.test function of package agricolae
y_pos	The y-position in relation to the boxplots. Choices are at the top of the box ("boxtop", the default) or at the maximum group value ("max").

Details

hsd_rslt must be created with agricolae::HSD.test

Value

A tibble with columns for treatment groups (x), the y-positions of the treatment CLD (y), and the CLD letters indicating significantly different treatments.

Examples

```
data("iris")
model <- lm(Petal.Length ~ Species, data = iris)
hsd_rslt <- agricolae::HSD.test(model, trt="Species")
cld_hsd(hsd_rslt)
```

---

clear_warnings *Clear Warnings*

Description

Clears all warning messages from the base environment.

Usage

```
clear_warnings()
```

Details

Sometimes when working in the console R retains a list of warnings such that they keep being reported after the function call which originated them. This function removes them so that they are not a nuisance

Examples

```
## Not run:
clear_warnings()

## End(Not run)
```

comb *comb*

Description

Calculates the number of combinations of n things drawn r at a time.

Usage

```
comb(n, r, repetition = FALSE)
```

Arguments

n	The total number of items.
r	The number of items to be drawn.
repetition	A logical, whether or not repetitions are allowed. FALSE by default.

Value

An integer giving the number of ways a set of r items can be drawn from a set of n items.

Examples

```
comb(5, 3)
comb(5, 3, repetition = TRUE)
```

comp_assemble *Assemble Comparison Parts*

Description

Assembles Comparison Data Frame

Usage

```
comp_assemble(part1, part2, part3)
```

Arguments

part1	Result from comp_means_sd
part2	Result from comp_make_f_tests
part3	Result from comp_comparisons

Value

A summary data frame of differential abundances by taxon and treatment.

Examples

```
## Not run:
See vignette

## End(Not run)
```

`comp_comparisons` *Make Comparisons*

Description

Calculates the treatment comparison portion of a table comparing relative abundances of each taxon among treatments.

Usage

```
comp_comparisons(
  otu.pc,
  otu.pc.trans,
  grps,
  p.adjust.method = "BH",
  pool.sd = FALSE
)
```

Arguments

`otu.pc`	An OTU table of percentages.
`otu.pc.trans`	An OTU table of transfromed data.
`grps`	A vector of treatemnt groups for which to make comparisons.
`p.adjust.method`	Adjustment method for multiple comparisons.
`pool.sd`	A logical, whether or not to pool standard deviations.

Value

A data frame of differences in relative abundances among treatments.

Examples

```
## Not run:
See vignette

## End(Not run)
```

comp_make_f_tests *Make F Tests*

Description

Calclates omnibus F tests to be included in a table comparing relative abundances of each taxon among treatments.

Usage

```
comp_make_f_tests(otu.pc.trans, grps, var.equal = FALSE)
```

Arguments

otu.pc.trans	An OTU table of transformed data from comp_prepare_otu_table.
grps	A vector of treatemnt groups for which to make comparisons.
var.equal	Logical, whether or not to assume variances equal.

Value

A data frame of the F-test results.

Examples

```
## Not run:
See vignette

## End(Not run)
```

comp_means_sd *Calculate Means and Standard Deviations*

Description

Calculates means and standard deviation for each taxon to be included in a table comparing relative abundances of each taxon among treatments.

Usage

```
comp_means_sd(otu.pc)
```

Arguments

otu.pc	An OTU table with data as percentages.

Details

The OTU table should be created with comp_prepare_otu_table.

Value

A data frame with means and standard deviations by taxon.

Examples

```
## Not run:
See vignette

## End(Not run)
```

`comp_prepare_otu_table` *Prepare OTU Table*

Description

Make OTU tables for making comparisons of relative abundances among treatments.

Usage

```
comp_prepare_otu_table(
  expt.taxon.pc,
  grps = "Treatment",
  transformation = "sqrt_arc_sine"
)
```

Arguments

`expt.taxon.pc`	Phyloseq object from comp_prepare_phyloseq with percentages in the otu_table.
`grps`	Factor in sample data for which to make comparisons.
`transformation`	Transformation function to use.

Details

transformation may be "none" or a user-supplied function name in quotation marks or any of the built-it transformations("arc_sine", "log_arc_sine", or "sqrt_arc_sine"). The +sqrt_arc_sine" has generaally proven most effective.

Value

A list consisting of an OTU table with percentages, an OTU table with transformed data, and a vector of treatment groups.

Examples

```
## Not run:
See vignette

## End(Not run)
```

comp_prepare_phyloseq *Prepare Phyloseq*

Description

Prepares a phyloseq object for making comparisons of relative abundances among treatments.

Usage

```
comp_prepare_phyloseq(expt, taxrank = "Phylum", pc.filter = 0.01)
```

Arguments

expt	Experiment level phyloseq object.
taxrank	Taxonomic rank for which to make comparisons.
pc.filter	Minimum percentage of total counts to include rank in result.

Details

The otu_table in one of the returned objects has been transformed to percentages based on the original phyloseq object supplied. The taxa in both have been filtered to include only OTUs initially present at >= pc.filter times the original total counts. For both only taxrank is included in the tax_table.

Value

A list of two modified experiemnt level phyloseq objects

Examples

```
## Not run:
See vignette

## End(Not run)
```

deg2rad *deg2rad*

Description

Degrees to radians

Usage

```
deg2rad(x)
```

Arguments

x	Angle in degrees

Value

Angle in radians.

Examples

```
deg2rad(90)
```

`format_taxon` *Format a taxon*

Description

Formats a taxon so that it will be properly italicized in a ggpplot

Usage

```
format_taxon(x)
```

Arguments

x	A string representing a phylum, class, etc.

Details

If a taxon begins with an upper case letter followed by lower case letters and does not contain an underscore, it is wrapped in asterisks. If it begins with an upper case letter followed by lower case letters and contains an underscore, the portion before the underscore is wrapped in asterisks, the underscore removed and any letters following the underscore left alone. If a taxon contains all upper case letters or digits it is not a proper taxon and is left alone.

Value

The string with asterisks properly inserted.

Examples

```
format_taxon("UAB")
format_taxon("Pseudomonas_B")
format_taxon("Pseudomonas")
```

generate_password *Genererate a Password*

Description

Generates a random character string of specified length.

Usage

```
generate_password(n, type = "alpha_numeric")
```

Arguments

n	Numberof characters in password.
type	c("alpha_numeric", "anything_else")

Details

If type equals "alpha_numeric" (the default), only alpha-numeric characters are used to generate the password. If type does not equal "alpha_numeric" then at least one non-alpha-numeric symbol will be included in the password. In either case, the alpha characters used are both upper and lower case.

Value

A character string.

Examples

```
generate_password(8)
```

get_groups *get_groups*

Description

Assign treatment groups based on pairwise t-tests.

Usage

```
get_groups(ptt.rslt, alpha = 0.05, rm.subset = FALSE)
```

Arguments

ptt.rslt	Result from the stats function pairwise.t.test.
alpha	Confidence level.
rm.subset	A logical; remove group subsets if true.

Details

This function aids in making letter assignments as to which treatments are significantly different. Also returns a square matrix of alpha values for all pairwise differences. This square matrix can serve as input to the multcompLetters function of the multcompView package which provides letter assignments. If rm.subset is FALSE, then groups such as {A,B} and {A, B, C} may be reported. This is redundant in the sense the {A, B} is a subset of {A, B, C}. In this case if rm.subset is FALSE, the group {A, B} is not reported.

Value

A list consisting of groups of treatment groups that are not significantly different and a matrix of p values.

See Also

make_letter_assignments

Examples

```
attach(airquality)
Month <- factor(Month, labels = month.abb[5:9])
ptt.rslt <- pairwise.t.test(Ozone, Month)
detach(airquality)
get_groups(ptt.rslt, alpha = 0.05, rm.subset = FALSE)
```

---

get_plot_limits *Get ggplot Plot Limits*

Description

Gets the ranges for the width and height of a ggplot panel.

Usage

```
get_plot_limits(plot)
```

Arguments

plot	A plot created with ggplot2

Value

A list: xmin, xmax, ymin, ymax

Examples

```
library(ggplot2)
data(iris)
plt <- ggplot(data=iris, aes(x=Species, y=Petal.Length)) + geom_boxplot()
get_plot_limits(plt)
```

goods *Calculate Good's Coverage*

Description

Calculates Good's coverage from a community data matrix with samples as rows and OTUs as columns.

Usage

```
goods(com)
```

Arguments

com	a vegan compatible community data matrix.

Value

A table with the headings number of singletons, number of sequences, and Good's coverage for each sample in rows.

References

Good, I. J. 1953. The Population Frequencies of Species and the Estimation of Population Parameters. Biometrika 40:237-264.

Examples

```
## Not run:
goods(species_matrix)

## End(Not run)
```

hash_dna_seqs *Hash DNA sequences*

Description

Renames sequences in a multi-fasta file with the MD5 hash of each sequence.

Usage

```
hash_dna_seqs(seqs)
```

Arguments

seqs	A list of DNA sequences

Details

Taxa names for the ASV table returned by the R version of DADA2 are the sequences themselves. This function renames them with the MD5 hash of each sequences so that the results are directly comparable QIIME2/DADA2 results.

Value

DNA sequences renamed with their hashes.

Examples

```
seqs <- ">ASV1
AGTTTGATCATGGCTCAGATTGAACGCTGGCGGCAGGCCTAACACATGCAAGTCGAACGGTAACAGGAAG"
hash_dna_seqs(seqs)
```

---

its.root *An Experiment Level phyloseq Object*

Description

Based on ITS2 sequences amplified from corn roots.

Usage

```
its.root
```

Format

A phyloseq object with otu_table, sample_data and tax_table. The sample_data variables are:

P Phosporous level, H or L

Genotype One of three: 2, 3, and C

Label A code for treatments: 2HR, 2LR, 3HR, 3LR, CHR, CLR

---

log_arc_sine *log_arc_sine*

Description

Log of the arc-sine Transfromation of a Percentage

Usage

```
log_arc_sine(x)
```

Arguments

x	A percentage.

Value

The common logarithm of the arcsine transformation of x.

Examples

```
log_arc_sine(x = 30.1)
```

`make_comparisons` *Make Multiple Comparisons on Transformed Data*

Description

Makes multiple comparisons of the relative abundances of taxa between treatment groups using the pairwise.t.test. Data may be transformed by a user supplied function. Three are included in this package.

Usage

```
make_comparisons(
  expt,
  taxrank = "Phylum",
  grps = "Treatment",
  transformation = "none",
  pc.filter = 0.01,
  p.adjust.method = "BH",
  pool.sd = FALSE
)
```

Arguments

`expt`	Experiment level phyloseq object.
`taxrank`	Rank for which to make comparisons.
`grps`	Factor in sample data for which to make comparisons.
`transformation`	Transformation function to use.
`pc.filter`	Minimum percentage of total counts to include rank in result.
`p.adjust.method`	Adjustment method for multiple comparisons.
`pool.sd`	Logical, whether or not to pool standard deviations.

Details

transformation may be "none" or a user-supplied function name in quotation marks or any of the built-it transformations ("arc_sine", "log_arc_sine", or "sqrt_arc_sine"). The "sqrt_arc_sine" has generaally proven most effective.

Value

A data frame with taxa as rows and results in columns.

See Also

arc_sine, log_arc_sine, sqrt_arc_sine, check_var

Examples

```
## Not run:
See vignette

## End(Not run)
```

`make_letter_assignments`
Make Letter Assignments

Description

Makes letter assignments for treatment groups that are not significantly different.

Usage

```
make_letter_assignments(ptt.rslt, significance = 0.05)
```

Arguments

`ptt.rslt`	Output from the pairwise.t.test function.
`significance`	Alpha level to be declared a significant difference.

Details

Letter assignments are made using Piepho's algorithm.

Value

Lists of letter assignments.

References

Piepho, H. P. 2004. An algorithm for a letter-based representation of all-pairwise comparisons. Journal of Computational and Graphical Statistics **13**:456-466.

Examples

```
## Not run:
make_letter_assignments(ptt.rstl, significance = 0.05)

## End(Not run)
```

merge_2_frames *Merge Two Data Frames*

Description

Merge two data frames by their row names.

Usage

```
merge_2_frames(one, two)
```

Arguments

one	A data frame.
two	A second data frame.

Details

Merges data frames by common row names. This function differs from merge.data.frames in that the merged data frame returned has row names and not a new column of the row names.

Value

A merged data frame.

Examples

```
## Not run:
merge_2_frames(dataframe_1, dataframe_2)

## End(Not run)
```

ord_labels *Make Ordination Axis Labels*

Description

Makes ordination axis labels that include, if appropriate, the % total variance explained by each axis.

Usage

```
ord_labels(ord)
```

Arguments

ord	A vegan ordination object.

Details

If there are no eigenvalues in ord, or if any eigenvalues are less than 0, each element of the vector returned has the form "DIMn" where N is the axis number. Otherwise, each element of the vector returned has the form "AxisN xx.x%" where "Axis" is taken from the vector of eigenvalues in ord if they are named or simply "DIM" if they are not, N is the number of the axis, and xx.x is the % of total variance explained by the axis.

Value

A character vector, each element of which can be used to label the corresponding axis of an ordination plot.

Examples

```
## Not run:
ord_labels(nmds_ordination)
ord_labels(pca_ordination
ord_labels(pcoa_ordination))

## End(Not run)
```

---

pca_labels *Make PCA Axis Labels*

Description

Makes PCA axis labels that include the

Usage

```
pca_labels(pca)
```

Arguments

pca	Object containig the results of vegan's rda function.

Details

Each element of the vector returned has the form "PCAn xx.x

Value

A character vector, each element of which can be used to label the corresponding axis of a PCA plot.

Examples

```
## Not run:
pca_labels(pca_ordination)

## End(Not run)
```

perm *Permutations*

Description

Retuns the number of permutaions of n things taken r at a time.

Usage

```
perm(n, r, repetition = FALSE)
```

Arguments

n	Total number of items.
r	Number of items drawn.
repetition	A logical, whether or not repetitions are allowed. FALSE by default.

Value

An integer giving how many ways m things can be drawn n at a time.

Examples

```
perm(10, 5)
perm(10, 5, repetition = TRUE)
```

plot_df *A Data File in Long Format*

Description

Used in Case 3 of the vignette make_comparisons

Usage

```
plot_df
```

Format

A data file in long format used for a ggplot. The sample_data variables are:

Treatment A code for genotype (2, 3, or C), P level (H or L) and sample type (R)

Family One of the families in Gigasporaceae

Percent Percent of total counts for family and treatment combination.

plot_transition_stats *Plot DADA2 Transition Stats*

Description

Extracts a QIIME2/DADA2 transition stats file and returns a plot

Usage

```
plot_transition_stats(trans_stats.qza)
```

Arguments

trans_stats.qza	The transitions stats file output by QIIME2 DADA2

Details

Beginning with QIIME2 version 2025.7 the DADA2 plugin requires outut of a compressed (qza) file of the transition stats. This function makes a ggplot plot from the data in that file. It is useful in determining how well DADA2 has corrected read errors.

Value

A ggplot of the transition probabilities

prop_filter *Filter OTUs by Abundance*

Description

Allows subsetting of a phyloseq object according to the relative abundance of OTUs in a minimal number of samples. Returns a logical vector of OTUs that are at least n% of the sequences in at leas m samples.

Usage

```
prop_filter(x, n, m)
```

Arguments

x	A phyloseq object.
n	Minimum percentage to keep OTU.
m	Minimum number of samples.

Details

The functions creates a logical vector to be used in subsetting a phyloseq object according to the relative abundance of OTUs in a given number of samples. For example, if n = 1 and m = 2, then the OTUs to be kept must represent at least 1% of the sequences in at least 2 samples. The vector is then used as an argument to the phyloseq object 'prune_taxa'.

Value

A logical vector of OTUs to keep.

Examples

```
## Not run:
prop_filter(expt, 1, 5)

## End(Not run)
```

`QsRutils` *QsRutils: R Functions Useful for Community Ecology*

Description

The QsRutils package contains functions I have written to make some aspects of using phyloseq and vegan simpler. I originally called the package MyRutils, but that does not make much sense if I am posting it publically!

`rad2deg` *rad2deg*

Description

Radians to degrees

Usage

```
rad2deg(x)
```

Arguments

`x`	Angle in radians

Value

Angle in degrees.

Examples

```
rad2deg(pi * 0.5)
```

rda_labels *Make RDA Axis Labels*

Description

Makes RDA axis labels that include the

Usage

```
rda_labels(rda)
```

Arguments

rda	Object that contains CCA result from vegan's rda function.

Details

Each element of the vector returned has the form "RDAn xx.x

Value

A character vector, each element of which can be used to label the corresponding axis of an RDA plot.

Examples

```
## Not run:
rda_labels(rda_ordination)

## End(Not run)
```

root_phyloseq_tree *Root Tree in phyloseq Object*

Description

Roots an unrooted tree in a phyloseq object

Usage

```
root_phyloseq_tree(phylo)
```

Arguments

phylo	A phyloseq object containing an unrooted tree

Details

The tree is rooted by the longest terminal branch.

Value

The same phyloseq object with a rooted tree

Examples

```
## Not run:
expt.rooted <- root_phyloseq_tree(expt.unrooted)

## End(Not run)
```

---

se *Standard error*

Description

Calculates the standard error of a numeric vector

Usage

```
se(x)
```

Arguments

x	A numeric vector

Details

NA values are ignored.

Value

The standard error of the numeric vector

Examples

```
x <- c(1,2,3,4,5, NA)
se(x)
```

sqrt_arc_sine *sqrt_arc_sine*

Description

Square Root of the arc-sine of a Percentage

Usage

```
sqrt_arc_sine(x)
```

Arguments

x	A percentage.

Value

The square root of the arcsine transformation of x.

Examples

```
sqrt_arc_sine(30.1)
```

srs_p *srs_p*

Description

Normalize sample counts using scaling with ranked subsampling (SRS)

Usage

```
srs_p(p)
```

Arguments

p	a phyloseq object containing an OTU table

Details

This is an alternative to "rarefying" an OTU table to a constant sample size. The phyloseq object submitted must be pruned to the desired sample size before using this function.

Value

a phyloseq object including an OTU table, all sample sums equal.

Author(s)

John Quensen

References

Beule L, Karlovsky P. Improved normalization of species count data in ecology by scaling with ranked subsampling (SRS): application to microbial communities. PeerJ. 2020;8:e9593.

Examples

```
## Not run:
srs_p(p)

## End(Not run)
```

`subset_by_refseq_lengths` *Subset physeq by refseq lengths*

Description

Subset physeq by refseq lengths

Usage

```
subset_by_refseq_lengths(p, min_len = 252, max_len = 255)
```

Arguments

`p`	An experiment level phyloseq object with reference sequences
`min_len`	The minimum reference sequence length to keep
`max_len`	The maximum references sequences length to keep

Details

Sometimes, due to sequencing errors, reference sequences have lengths less than and/or greater than the expected amplicon length. This function offers an easy way of removing such extraneous reference sequences from an experiment level phyloseq object. The default values for min_len and max_len are fro the V4 region of the 16S rRNA gene.

Value

A phyloseq object filtered to have reference sequences within the length range specified.

Examples

```
## Not run:
p <- subset_by_refseq_lengths(p)

## End(Not run)
```

subset_dist *Subset Distance Matrix*

Description

Subsets a distance matrix.

Usage

```
subset_dist(physeq, d.matrix)
```

Arguments

physeq	An experiment level phyloseq object.
d.matrix	A distance matrix.

Details

Some distance matrices take a long time to calculate for large data sets. This is especially true of unifrac and generalized unifrac distances calculated by GUniFracs. If distances are first calculated from data in a large experiment level phyloseq object and then it is desired to perform PERMANOVA (with adonis) on a subset of that object, this function provides a means of sub-setting the distance matrix so that it does not have to be calculated again for the subset data. The arguments are the distance matrix for the original phyloseq object and the smaller phyloseq object subset from the original.

Value

A distance matrix of smaller dimensions.

References

Chen J, Bittinger K, Charlson ES et al. (2012) Associating microbiome composition with environmental covariates using generalized UniFrac distances. Bioinformatics, 28, 2106-2113.

Examples

```
otu <- veganotu(its.root)
d <- vegdist(otu)
d
str(d)
p <- subset_samples(its.root, P_Location == "LR")
d_sub <- QsRutils::subset_dist(p, d)
str(d_sub)
```

veganotu *Extract Vegan OTU Table*

Description

Extracts a vegan compatible OTU table from a phyloseq object.

Usage

```
veganotu(physeq)
```

Arguments

physeq	A phyloseq object contaning at least an OTU table.

Value

A matrix with samples in rows and OTUs in columns.

Examples

```
## Not run:
veganotu(physeq)

## End(Not run)
```

vegansam *Extract Sample Data Table*

Description

Extracts a sample data table from a phyloseq object.

Usage

```
vegansam(physeq)
```

Arguments

physeq	A phyloseq object containing sample_data.

Value

A data frame with samples in rows and factors and/or variables in columns.

Examples

```
## Not run:
vegansam(physeq)

## End(Not run)
```

vegan_stand *Standardize a Phyloseq OTU Table*

Description

Applies any vegan decostand standardization method to a phyloseq OTU table.

Usage

```
vegan_stand(physeq, method = "hellinger", ...)
```

Arguments

physeq	A phyloseq object containing at least an OTU table.
method	A method from vegan's decostand function.
...	Other parameters passed to vegan's decostand function.

Value

Returns a phyloseq object with transformed OTU table.

Examples

```
## Not run:
veganstand(expt, method = "hellinger")

## End(Not run)
```

%wo% *Remove elements from a vector*

Description

Removes elements present in ‘y‘ from ‘x‘.

Usage

```
x %wo% y
```

Arguments

x	A vector.
y	A vector of elements to remove from ‘x‘.

Value

A vector containing elements of ‘x‘ that are not in ‘y‘.

Examples

```
c(1, 2, 3, 4, 5) %wo% c(2, 4)
letters[1:5] %wo% c("b", "d")
```

Index