Package 'Rsearch'

October 26, 2025

Type Package

Title Processing and Analyzing Amplicon Sequence Data

Version 1.0.0

Maintainer Cassandra Stamsaas <cassandra.hjortdahl@nmbu.no>

Description Processing and analysis of targeted sequencing data. The package provides a user-friendly interface for core 'VSEARCH' (Rognes et al. (2016), <doi:10.7717/peerj.2584>) functions, in addition to tools for visualization and parameter tuning.

License GPL (>= 3)

Encoding UTF-8

Imports ape (>= 5.8.1), cowplot (>= 1.1.3), dplyr, ggExtra (>= 0.10.1), ggplot2 (>= 3.5.1), microseq (>= 2.1.6), phyloseq (>= 1.50.0), RColorBrewer (>= 1.1.3), readr (>= 2.1.5), stats, stringr (>= 1.5.1), tibble (>= 3.2.1), tidyr (>= 1.3.1)

Suggests rmarkdown, testthat (>= 3.0.0), withr (>= 3.0.2)

Config/testthat/edition 3

RoxygenNote 7.3.3

URL https://cassandrahjo.github.io/Rsearch/, https://github.com/CassandraHjo/Rsearch/

BugReports https://github.com/CassandraHjo/Rsearch/issues/

Depends R (>= 4.1)

Config/Needs/website rmarkdown

NeedsCompilation no

Author Cassandra Stamsaas [cre, aut],
Lars Snipen [aut],
Torbjørn Rognes [aut],
Hilde Vinje [aut]

Repository CRAN

Date/Publication 2025-10-26 18:40:14 UTC

Contents

`fastx_combine_files` *Combine FASTA/FASTQ files in a directory into a single file or object*

Description

`fastx_combine_files` combines all FASTA or FASTQ files within a specified directory into a single output file or a tibble object.

Usage

```
fastx_combine_files(
  files_dir,
  output_file = NULL,
  file_ext = ".fq",
  file_format = "fastq",
  tmpdir = NULL
)
```

Arguments

files_dir	(Required). A character string specifying the path to the directory containing the files to be combined. Files must be uncompressed.
output_file	(Optional). A character string specifying the name of the output file. If NULL (default), the combined data is returned as a FASTA/FASTQ object depending on file_format instead of being written to a file.
file_ext	(Optional). File extension of the files to be combined. Defaults to ".fq".
file_format	(Optional). Format of files to be combined and the desired output format: either "fasta" or "fastq" (default). See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

files_dir must contain uncompressed FASTA or FASTQ files matching the specified file_ext.

All files with the specified file_ext in files_dir are concatenated into a single output file or tibble.

A FASTA object is a tibble containing the columns Header and Sequence. A FASTQ object is a tibble containing the columns Header, Sequence, and Quality.

If output_file is specified, the combined sequences are written to this file in the format specified by file_format.

If output_file is NULL, the combined sequences are returned as a tibble in the format specified by file_format, and no file is written.

Value

A tibble or NULL.

If output_file is specified, the combined sequences are written to the specified file.

If output_file is NULL, the combined sequences are returned as a tibble in the format specified by file_format.

Examples

```
# Define arguments
files_dir <- system.file("extdata", package = "Rsearch")
output_file <- NULL
file_ext <- ".fq"
file_format <- "fastq"

# Combine files and return tibble object
combined_files <- fastx_combine_files(files_dir = files_dir,
                                      output_file = output_file,
                                      file_ext = file_ext,
                                      file_format = file_format)

# Combine files and write to output file

# Define output file name
out <- tempfile(fileext = ".fastq")

fastx_combine_files(files_dir = files_dir,
                    output_file = out,
                    file_ext = file_ext,
                    file_format = file_format)
```

`fastx_synchronize` *Synchronize FASTA and FASTQ files or objects*

Description

`fastx_synchronize` synchronizes sequences between two FASTA/FASTQ files or objects by retaining only the common sequences present in both.

Usage

```
fastx_synchronize(
  file1,
  file2 = NULL,
  file_format = "fastq",
  file1_out = NULL,
  file2_out = NULL
)
```

Arguments

`file1`	(Required). A FASTQ file path, a FASTQ tibble, or a paired-end tibble of class `"pe_df"`. See *Details*.
`file2`	(Optional). A FASTQ file path or a FASTQ tibble. Optional if `file1` is a `"pe_df"` object. See *Details*.

file_format	(Optional). Format of the input (file1 and file2) and the desired output format: "fasta" or "fastq" (default). This determines the format for both outputs.
file1_out	(Optional). Name of the output file for synchronized reads from file1. The file is in either FASTA or FASTQ format, depending on file_format. If NULL (default), no sequences are written to a file. See *Details*.
file2_out	(Optional). Name of the output file for synchronized reads from file2. The file is in either FASTA or FASTQ format, depending on file_format. If NULL (default), no sequences are written to a file. See *Details*.

Details

file1 and file2 can either be paths to FASTA/FASTQ files or tibble objects containing the sequences. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta. FASTQ objects are tibbles that contain the columns Header, Sequence, and Quality, see readFastq.

If file1 is an object of class "pe_df", the second read tibble is automatically extracted from its "reverse" attribute unless explicitly provided via the file2 argument. This allows streamlined input handling for paired-end tibbles created by vs_fastx_trim_filt.

Sequence IDs in the Header fields must be identical for each read pair in both file1 and file2 for synchronization to work correctly.

If file1_out and file2_out are specified, the synchronized sequences are written to these files in the format specified by file_format.

If file1_out and file2_out are NULL, the function returns a FASTA/FASTQ object containing synchronized reads from file1. The synchronized reads from file2 are included as an attribute named "reverse" in the returned tibble.

The returned tibble is assigned the S3 class "pe_df", indicating that it represents paired-end sequence data. Downstream functions can use this class tag to distinguish paired-end tibbles from other tibbles.

Both file1_out and file2_out must either be NULL or both must be character strings specifying the file paths.

Value

A tibble or NULL.

If both file1_out and file2_out are NULL, a tibble containing the synchronized reads from file1 is returned. The synchronized reads from file2 are accessible via the "reverse" attribute of the returned tibble.

If both file1_out and file2_out are specified, the synchronized sequences are written to the specified output files, and no tibble is returned.

Examples

```
# Define arguments
file1 <- system.file("extdata/small_R1.fq", package = "Rsearch")
file2 <- system.file("extdata/small_R1.fq", package = "Rsearch")
file_format <- "fastq"
file1_out <- NULL
```

```
file2_out <- NULL

# Synchronize files and return as a tibble
sync_seqs <- fastx_synchronize(file1 = file1,
                               file2 = file2,
                               file_format = file_format,
                               file1_out = file1_out,
                               file2_out = file2_out)

# Extract tibbles with synchronized sequences
R1_sync <- sync_seqs
R2_sync <- attr(sync_seqs, "reverse")

# Synchronize files and write to output files

# Define output file names
out1 <- tempfile(fileext = ".fastq")
out2 <- tempfile(fileext = ".fastq")

fastx_synchronize(file1 = file1,
                  file2 = file2,
                  file_format = file_format,
                  file1_out = out1,
                  file2_out = out2)
```

`make_sintax_db` *Make Sintax database*

Description

Creates a properly formatted FASTA file for the use as a Sintax database.

Usage

```
make_sintax_db(taxonomy_table, outfile)
```

Arguments

`taxonomy_table`	(Required). A data.frame with sequences and proper information for making a Sintax database, see *Details*.
`outfile`	(Required). Name of database file to create (a FASTA file).

Details

The Sintax algorithm is used by VSEARCH to assign taxonomic information to 16S sequences. It requires a database, which is nothing but a FASTA file of 16S sequences with properly formatted Header-lines.

The taxonomy_table provided as input here must have the columns:

- Header - short unique text for each sequence
- Sequence - the sequences
- Columns domain, phylum, class, order, family, genus, species. Text columns with taxon names.

In some taxonomies the domain rank is named kingdom, but here we use the word domain. You may very well have empty (NA) entries in the taxonomy columns of the table.

Value

No return in R, but a FASTA file (outfile) with properly formatted Header lines is created.

References

https://www.biorxiv.org/content/10.1101/074161v1

Examples

```
## Not run:
# First, you need a table of the same format as output by vs_sintax:
db.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "sintax_db.fasta")
fasta.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                        "small.fasta")
tax.tbl <- vs_sintax(fasta_input = fasta.file, database = db.file)

# Inspect tax.tbl to see its columns. You replace the column content with
# your desired taxonomy.
# From such a tax.tbl you create the database file:
make_sintax_db(tax.tbl, outfile = "delete_ma.fasta")

## End(Not run)
```

---

phyloseq2rsearch *Convert phyloseq object to Rsearch object*

Description

Creating an Rsearch object (list) from a phyloseq object.

Usage

```
phyloseq2rsearch(phyloseq.obj)
```

Arguments

`phyloseq.obj`	(Required). A phyloseq object, see `phyloseq`.

Details

This function converts a phyloseq object to a simple `list` with three elements as dataframes (or tibbles). The entries are named according to the structure used in `rsearch_obj`

Value

A `list` with entries as in a Rsearch object, except that the `sequence.tbl` do not contain sequences, only taxonomy.

References

https://joey711.github.io/phyloseq/

See Also

`rsearch_obj`

Examples

```
## Not run:
# Convert phyloseq object to Rsearch object
rsearch_obj <- phyloseq2rsearch(phy_obj)

# Extract read count data
rsearch_obj$readcount.mat

# Extract sample data
rsearch_obj$sampledata.df

# Extract sequence data
rsearch_obj$sequence.df

## End(Not run)
```

plot_base_quality *Plot quality scores per position for FASTQ reads*

Description

Generates a plot displaying the quality scores for each position in FASTQ reads.

Usage

```
plot_base_quality(
  fastq_input,
  reverse = NULL,
  quantile_lower = 0.25,
  quantile_upper = 0.75,
  plot_title = "Per-position quality scores: median and mean",
  show_median = TRUE,
  show_mean = TRUE,
  show_overlap_box = FALSE,
  tmpdir = NULL
)
```

Arguments

fastq_input	(Required). A FASTQ file path or FASTQ object containing (forward) reads. See *Details*.
reverse	(Optional). An optional FASTQ file path or FASTQ tibble containing reverse reads. Defaults to NULL. See *Details*.
quantile_lower	(Optional). The lower quantile threshold for the error bars in the plot. Defaults to 0.25.
quantile_upper	(Optional). The upper quantile threshold for the error bars in the plot. Defaults to 0.75.
plot_title	(Optional). The title of the plot. Defaults to "Per-position quality scores: median and mean". Set to "" for no title.
show_median	(Optional). If TRUE (default), a line representing the median quality scores is added to the plot.
show_mean	(Optional). If TRUE (default), a line representing the mean quality scores is added to the plot.
show_overlap_box	(Optional). If TRUE, a shaded box is drawn to indicate the mean overlap length that would result from merging all reads in their current state. This visualization is only applicable when reverse is specified. Defaults to FALSE.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

The mean and median quality scores for each base position over all reads are plotted as curves. The vertical bars at each base indicate the interquartile range.

`fastq_input` and `reverse` can either be file paths to FASTQ files or FASTQ objects. FASTQ objects are tibbles that contain the columns `Header`, `Sequence`, and `Quality`, see `readFastq`.

If `reverse` is provided, it is plotted together with the first plot in its own panel. Note that the x-axis in this panel is reversed.

The vertical bars represent the interquartile range (25% - 75%) in the quality scores. Custom quantile ranges can be specified via `quantile_lower` and `quantile_upper`. Additionally, the median and mean quality lines, and overlap-shading box may be turned off by setting `show_median = FALSE`, `show_mean = FALSE`, or `show_overlap_box = FALSE`, respectively.

If `fastq_input` (and `reverse`, if provided) contains more than 10 000 reads, the function will randomly select 10 000 rows for downstream calculations. This subsampling is performed to reduce computation time and improve performance on large datasets.

Value

A ggplot2 object.

Examples

```
# Define inputs
fastq_input <- system.file("extdata/small_R1.fq", package = "Rsearch")
reverse <- system.file("extdata/small_R2.fq", package = "Rsearch")

# Generate and display quality plot with both median and mean lines
qual_plots <- plot_base_quality(fastq_input = fastq_input,
                                reverse = reverse)
print(qual_plots)

# Generate and display quality plot without the plot title
qual_plots_wo_title <- plot_base_quality(fastq_input = fastq_input,
                                         reverse = reverse,
                                         plot_title = "")
print(qual_plots_wo_title)

# Generate a plot showing only the median quality line
qual_plots_median_only <- plot_base_quality(fastq_input = fastq_input,
                                            reverse = reverse,
                                            show_mean = FALSE)
print(qual_plots_median_only)

# Generate a plot showing only the mean quality line
qual_plots_mean_only <- plot_base_quality(fastq_input = fastq_input,
                                          reverse = reverse,
                                          show_median = FALSE)
print(qual_plots_mean_only)
```

`plot_ee_rate_dist` *Plot distribution of expected error (EE) rate of reads*

Description

Generates a histogram visualizing the distribution of the expected error (EE) rate for reads. The EE rate represents the cumulative probability of errors in a read, calculated from Phred quality scores.

Usage

```
plot_ee_rate_dist(
  fastq_input,
  n_bins = 30,
  plot_title = "Distribution of the expected error (EE) rate of reads"
)
```

Arguments

`fastq_input`	(Required). A FASTQ file path or FASTQ object containing reads. See *Details*.
`n_bins`	(Optional). Number of bins used in the histogram. Defaults to `30`, which is the default value in `ggplot2::geom_histogram()`.
`plot_title`	(Optional). The title of the plot. Defaults to `"Distribution of the expected error (EE) rate of reads"`. Set to `""` for no title.

Details

A histogram is plotted using ggplot2 to visualize the distribution of EE rates. The user can adjust the number of bins in the histogram using the `n_bins` parameter.

`fastq_input` can either be a file path to a FASTQ file or a FASTQ object. FASTQ objects are tibbles that contain the columns `Header`, `Sequence`, and `Quality`, see `readFastq`.

The EE rate is calculated as the sum of error probabilities per read, where the error probability for each base is computed as $10^{(-Q/10)}$ from Phred scores. A lower EE rate indicates higher sequence quality, while a higher EE rate suggests lower confidence in the read.

If `fastq_input` contains more than 10 000 reads, the function will randomly select 10 000 rows for downstream calculations. This subsampling is performed to reduce computation time and improve performance on large datasets.

Value

A ggplot2 object displaying the histogram of EE rate distribution.

Examples

```
# Define input file path
fastq_input <- system.file("extdata/small_R1.fq", package = "Rsearch")

# Generate and display histogram
ee_plot <- plot_ee_rate_dist(fastq_input = fastq_input)
print(ee_plot)
```

`plot_read_quality` *Plot read length vs. read quality*

Description

Generates a scatter plot visualizing the relationship between read length and read quality. The y-axis can display either the mean quality score per read or the expected error (EE) rate. Marginal histograms are included to show the distribution of read lengths and quality metrics.

Usage

```
plot_read_quality(
  fastq_input,
  use_ee_rate = FALSE,
  plot_title = TRUE,
  alpha = 0.5
)
```

Arguments

`fastq_input`	(Required). A FASTQ file path or FASTQ object containing reads. See *Details*.
`use_ee_rate`	(Optional). If `TRUE`, the plot will display the expected error rate (EE) on the y-axis instead of the mean quality score. Defaults to `FALSE`.
`plot_title`	(Optional). If `TRUE` (default), a title will be displayed in the plot. The title will either be "Read length vs Expected error rate (EE) of read" or "Read length vs Average quality score of read", depending on `use_ee_rate`. Set to `FALSE` for no title.
`alpha`	(Optional). The transparency level of the points in the scatter plot. Defaults to `0.5`.

Details

This function visualizes the relationship between read length and read quality. The user can choose to plot either the mean quality score per read or the expected error (EE) rate.

`fastq_input` can either be a file path to a FASTQ file or a FASTQ object. FASTQ objects are tibbles that contain the columns `Header`, `Sequence`, and `Quality`, see `readFastq`.

The EE rate is calculated as the mean of error probabilities per read, where the error probability for each base is computed as $10^{(-Q/10)}$ from Phred scores. A lower EE rate indicates higher sequence quality, while a higher EE rate suggests lower confidence in the read.

Marginal histograms are added to display the distribution of read lengths (top) and quality scores or EE rates (right).

If `fastq_input` contains more than 10 000 reads, the function will randomly select 10 000 rows for downstream calculations. This subsampling is performed to reduce computation time and improve performance on large datasets.

Value

A ggplot2 object displaying the scatter plot with marginal histograms.

Examples

```
# Define arguments
fastq_input <- system.file("extdata/small_R1.fq", package = "Rsearch")

# Generate and display scatter plot with mean quality score on y-axis
p1 <- plot_read_quality(fastq_input = fastq_input)
print(p1)

# Generate and display scatter plot with mean quality score on y-axis
p2 <- plot_read_quality(fastq_input = fastq_input,
                        use_ee_rate = TRUE)
print(p2)
```

---

plot_size_dist *Plot distribution of size values*

Description

Generates a plot representing the distribution of size values from a FASTA or FASTQ file/object.

Usage

```
plot_size_dist(
  fastx_input,
  input_format = NULL,
  cutoff = NULL,
  y_breaks = NULL,
  plot_title = "Size distribution",
  log_scale_y = TRUE,
  n_bins = 30
)
```

Arguments

fastx_input	(Required). A FASTA/FASTQ file path or FASTA/FASTQ object containing reads with size values embedded in the Header column. See *Details*.
input_format	(Optional). The format of the input file. Must be "fasta" or "fastq" if fastx_input is a file path. Defaults to NULL.
cutoff	(Optional). A numeric value specifying a size threshold. Reads with size greater than this value will be grouped into a single category labeled "> cutoff" in the plot. Defaults to NULL (no cutoff applied).
y_breaks	(Optional). A numeric vector specifying the breakpoints for the y-axis if log10 scaling is applied (log_scale_y = TRUE. Defaults to NULL.
plot_title	(Optional). The title of the plot. Defaults to "Size distribution". Set to "" for no title.
log_scale_y	(Optional). If TRUE (default), applies a log10 scale to the y-axis. If FALSE, the y-axis remains linear.
n_bins	(Optional). Number of bins used in the histogram if cutoff is unspecified. Defaults to 30, which is the default value in ggplot2::geom_histogram().

Details

fastx_input can either be a file path to FASTA/FASTQ file or a FASTA/FASTQ object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta. FASTQ objects are tibbles that contain the columns Header, Sequence, and Quality, see readFastq. The Header column must contain the size values for each read.

The Header column must contain size annotations formatted as ;size=<int>.

The y-axis of the plot can be log10-transformed to handle variations in read counts across different size values. If y_breaks is specified, the given breakpoints will be used. If y_breaks is NULL, ggplot2 will automatically determine suitable breaks.

Value

A ggplot2 object displaying a plot of size distribution.

Examples

```
# Define input file
fastx_input <- system.file("extdata/small_derep_R1.fa", package = "Rsearch")

# Generate and display plot without cutoff
size_plot <- plot_size_dist(fastx_input = fastx_input,
                            input_format = "fasta")
print(size_plot)

# Generate and display plot with a cutoff at size 100
size_plot <- plot_size_dist(fastx_input = fastx_input,
                            input_format = "fasta",
                            cutoff = 100)
print(size_plot)
```

```
# Generate and display plot with custom y-axis breaks
size_plot <- plot_size_dist(fastx_input = fastx_input,
                            input_format = "fasta",
                            y_breaks = c(1, 50, 500, 5000))
print(size_plot)

# Generate and display plot with linear y-axis
size_plot <- plot_size_dist(fastx_input = fastx_input,
                            input_format = "fasta",
                            log_scale_y = FALSE)
print(size_plot)
```

`rsearch2phyloseq` *Convert Rsearch object to phyloseq object*

Description

`rsearch2phyloseq` converts an Rsearch object to a phyloseq object.

Usage

```
rsearch2phyloseq(rsearch.obj, sample_id_col = "sample_id")
```

Arguments

`rsearch.obj`	(Required). An Rsearch object, see `rsearch_obj`.
`sample_id_col`	(Optional). A character string specifying the name of the column in `sampledata.df` that contains sample identifiers. Defaults to `"sample_id"`.

Details

This function converts an Rsearch object, which is a simple `list`, to a `phyloseq` object from the `phyloseq` R package.

Value

A `phyloseq` object.

References

https://joey711.github.io/phyloseq/

See Also

`rsearch_obj`

Examples

```
## Not run:
# Convert Rsearch object to phyloseq object
phy_obj <- rsearch2phyloseq(obj, sample_id_col = "sample_id")

## End(Not run)
```

`rsearch_obj` *Create Rsearch object*

Description

`rsearch_obj` standardizes and organizes data into an Rsearch object. An Rsearch object is a list containing three elements with data structures that can be used as input to build a `phyloseq` object in the `phyloseq` package.

Usage

```
rsearch_obj(
  readcount_data,
  sequence_data,
  sample_data,
  sample_id_col = "sample_id"
)
```

Arguments

`readcount_data`	(Required). A file path or a data frame (or tibble) containing OTU count data, typically the output from `vs_cluster_size` or similar. This must have one row per OTU and one column per sample. The first column must contain OTU identifiers corresponding to those in the first column of `sequence_data`, and the remaining columns must have names matching the sample identifiers in `sample_data`. OTUs and samples not found across all data structures are discarded.
`sequence_data`	(Required). A file path or a data frame (or tibble) containing centroid sequences representing each OTU, typically obtained from clustering (`vs_cluster_size`) or denoising (`vs_cluster_unoise`). The first column must be called `Header` and contain OTU identifiers. One of the remaining columns must be named `Sequence`, containing the actual DNA sequences. Additional columns may include taxonomic classification data, e.g. from `vs_sintax`.
`sample_data`	(Required). A file path or a data frame (or tibble) containing metadata about each sample. Samples are assumed to be in rows, and one of the columns **must** contain a unique identifier for each sample that matches the column names in `readcount_data`.
`sample_id_col`	(Optional). A character string specifying the name of the column in `sample_data` that contains the unique sample identifiers. This column will be used to match sample metadata to read count data. Defaults to `"sample_id"`.

Details

This function standardizes and organizes data into an Rsearch object: a structured three key data components used or generated during the Rsearch workflow: read count data, sequence data, and sample data.

The function accepts three datasets—read count data, sequence data, and sample metadata, and returns a streamlined input suitable for constructing a phyloseq object using the `rsearch2phyloseq` function. The implementation uses a standard `list` in R rather than a specialized class providing an open and easily accessible structure.

To convert this object into a `phyloseq` object, use `rsearch2phyloseq`.

Value

A straightforward named list with three elements:

- `readcount.mat`: A numeric matrix of OTU abundances with OTUs as rows and samples as columns.
- `sequence.df`: A data.frame with one row for each OTU sequence and
- `sampledata.df`: A data frame containing data about the samples.

See Also

rsearch2phyloseq phyloseq2rsearch

Examples

```
# Define inputs
readcount.dta <- system.file("extdata/readcount_data.tsv", package = "Rsearch")
sequence.dta <- system.file("extdata/sequence_data.tsv", package = "Rsearch")
sample.dta <- system.file("extdata/sample_data.tsv", package = "Rsearch")

# Create Rsearch object
obj <- rsearch_obj(readcount_data = readcount.dta,
                   sequence_data = sequence.dta,
                   sample_data = sample.dta,
                   sample_id_col = "sample_id")

# Convert Rsearch object to phyloseq object
phy_obj <- rsearch2phyloseq(obj, sample_id_col = "sample_id")

# Convert phyloseq object to Rsearch object
rsearch_obj <- phyloseq2rsearch(phy_obj)
```

set_vsearch_executable *Set the VSEARCH executable*

Description

set_vsearch_executable specifies the valid command to invoke VSEARCH.

Usage

```
set_vsearch_executable(vsearch_executable)
```

Arguments

vsearch_executable
: (Required). Full path to the VSEARCH executable on your computer. See *Details* for more information on how to install VSEARCH.

Details

Use this function to change the command used to invoke the external software VSEARCH on this computer. When the Rsearch package is installed this command is by default just "vsearch".

If you have a windows computer and have copied the binary vsearch.exe to the folder C:/Documents/ on your computer, you update R with this information by set_vsearch_executable("C:/Documents/vsearch").

You may use the function vsearch to test if the command is valid.

Visit https://github.com/CassandraHjo/Rsearch for more information on how to install VSEARCH.

Value

Nothing is returned, but the option Rsearch.vsearch_executable is updated. The string is also saved to a file for later R sessions, i.e. you only need to update this once (or if you change how you run/install VSEARCH).

See Also

vsearch.

taxonomy_tree *Make a taxonomy tree*

Description

Creates a phylo object based on taxonomy

Usage

```
taxonomy_tree(taxonomy_table, confidence = NULL)
```

Arguments

taxonomy_table	(Required). A data.frame with sequences and taxonomy information, see *Details*.
confidence	(Optional). A threshold value used to replace taxa with confidence scores below this to NA.

Details

In some data analyses involving OTU data a phylogenetic tree describing the relatedness of the OTUs is required. To construct such trees you typically need to make a multiple alignment of the sequences behind each OTU, which is a huge job.

An alternative is then to simply use the taxonomy, and create a 'taxonomy-tree' instead of a phylogenetic tree. This function creates such a tree from a taxonomy table of the same format as output by vs_sintax.

Distances between two OTUs reflect how high up in the taxonomy they have a common taxon, i.e if they are of the same species the distance is 0, if different species but same genus the distance is 1 etc. Note that NAs in the taxonomy are not matched, increasing the distances, i.e if two OTUs have NA as species and genus, but share family, the distance is 2.

The confidence sets a threshold for replacing low-confidence taxa to NA. For this to work the taxonomy_table must have columns with such confidence scores i.e. columns domain_score, phylum_score, ...species_score. If the species_score is below confidence the corresponding species name is set to NA, and similar for all ranks. The default is to ignore this confidence (confidence = NULL).

From these distances a Neighbor Joining tree is built using nj.

Value

A phylo object, see nj.

References

https://www.biorxiv.org/content/10.1101/074161v1

Examples

```
## Not run:
# Assign taxonomy with sintax
db.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "sintax_db.fasta")
fasta.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small.fasta")
tax.tbl <- vs_sintax(fasta_input = fasta.file, database = db.file)

# Making tree
tax.tree <- taxonomy_tree(tax.tbl)

## End(Not run)
```

`vsearch` *Test if* `VSEARCH` *can be executed*

Description

`vsearch` tests if the VSEARCH executable is a valid command.

Usage

```
vsearch()
```

Details

Use this function to test the command used to invoke the external software VSEARCH on this computer.

Value

No return value, called for side effects (prints validation message to console).

See Also

`set_vsearch_executable`.

vs_alignment_classification
Taxonomic classification with LCA

Description

vs_alignment_classification assigns taxonomy by global alignment and Last Common Ancestor (LCA) consensus of database hits using VSEARCH.

Usage

```
vs_alignment_classification(
  fastx_input,
  database,
  lcaout = NULL,
  lca_cutoff = 1,
  top_hits_only = FALSE,
  gapopen = "20I/2E",
  gapext = "2I/1E",
  id = 0.7,
  strand = "plus",
  maxaccepts = 2,
  maxrejects = 32,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fastx_input	(Required). A FASTA/FASTQ file path or FASTA/FASTQ object. See *Details*.
database	(Required). A FASTA/FASTQ file path or FASTA/FASTQ tibble object containing the target sequences.
lcaout	(Optional). A character string specifying the name of the output file. If NULL (default), no output is written to a file and the results are returned as a tibble with the columns query_id and taxonomy.
lca_cutoff	(Optional). Adjust the fraction of matching hits required for the last common ancestor (LCA). Defaults to 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 between 0.5 and 1.0.
top_hits_only	(Optional). If TRUE, only the top hits with an equally high percentage of identity between the query and database sequence sets are written to the output. Defaults to FALSE.
gapopen	(Optional). Penalties for gap opening. Defaults to "20I/2E". See *Details*.

gapext	(Optional). Penalties for gap extension. Defaults to "2I/1E". See *Details*.
id	(Optional). Pairwise identity threshold. Defines the minimum identity required for matches. Defaults to 0.7.
strand	(Optional). Specifies which strand to consider when comparing sequences. Can be either "plus" (default) or "both".
maxaccepts	(Optional). Maximum number of matching target sequences to accept before stopping the search for a given query. Defaults to 2. Must be larger than 1 for information to be useful.
maxrejects	(Optional). Maximum number of non-matching target sequences to consider before stopping the search for a given query. Defaults to 32. If maxaccepts and maxrejects are both set to 0, the complete database is searched.
threads	(Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.
vsearch_options	(Optional). Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Performs global sequence alignment against a reference database and assigns taxonomy using the Last Common Ancestor (LCA) approach, reporting the deepest taxonomic level consistently supported by the majority of hits.

fastx_input and database can either be file paths to a FASTA/FASTQ files or FASTA/FASTQ objects. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta. FASTQ objects are tibbles that contain the columns Header, Sequence, and Quality, see readFastq.

Pairwise identity (id) is calculated as the number of matching columns divided by the alignment length minus terminal gaps.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Visit the VSEARCH documentation for information about defining gapopen and gapext.

Value

A tibble or NULL.

If lcaout is specified the results are written to the specified file. If lcaout is NULL a data.frame is returned.

The data.frame contains the classification results for each query sequence. Both the Header and Sequence columns of fasta_input are copied into this table, and in addition are also the columns for each rank. The ranks depend on the database file used, but are typically domain, phylum, class, order,family, genus and species.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Example files
db.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "sintax_db.fasta")
fasta.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small.fasta")

tax.tbl <- vs_alignment_classification(fastx_input = fasta.file,
                                       database = db.file)
View(tax.tbl)

## End(Not run)
```

`vs_cluster_size` *Cluster FASTA sequences*

Description

`vs_cluster_size` clusters FASTA sequences from a given file or object using `VSEARCH´s cluster_size` method. The function automatically sorts sequences by decreasing abundance before clustering.

Usage

```
vs_cluster_size(
  fasta_input,
  centroids = NULL,
  otutabout = NULL,
  size_column = FALSE,
  id = 0.97,
  strand = "plus",
  sizein = TRUE,
  sizeout = TRUE,
  relabel = NULL,
  relabel_sha1 = FALSE,
  fasta_width = 0,
  sample = NULL,
  log_file = NULL,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fasta_input	(Required). A FASTA file path or a FASTA object containing reads to cluster. See *Details*.
centroids	(Optional). A character string specifying the name of the FASTA output file for the cluster centroid sequences. If NULL (default), no output is written to a file and the centroid sequences are returned as a FASTA object. See *Details*.
otutabout	(Optional). A character string specifying the name of the output file in an OTU table format. If NULL (default), no output is written to a file. If TRUE, the output is returned as a tibble. See *Details*.
size_column	(Optional). If TRUE, a column with the size of each centroid is added to the centroid output tibble.
id	(Optional). Pairwise identity threshold for sequence to be added to a cluster. Defaults to 0.97. See *Details*.
strand	(Optional). Specifies which strand to consider when comparing sequences. Can be either "plus" (default) or "both".
sizein	(Optional). If TRUE (default), abundance annotations present in sequence headers are taken into account.
sizeout	(Optional). If TRUE (default), abundance annotations are added to FASTA headers.
relabel	(Optional). Relabel sequences using the given prefix and a ticker to construct new headers. Defaults to NULL.
relabel_sha1	(Optional). If TRUE (default), relabel sequences using the SHA1 message digest algorithm. Defaults to FALSE.
fasta_width	(Optional). Number of characters per line in the output FASTA file. Defaults to 0, which eliminates wrapping.
sample	(Optional). Add the given sample identifier string to sequence headers. For instance, if the given string is "ABC", the text ";sample=ABC" will be added to the header. This option is only applicable when the output format is FASTA (centroids). If NULL (default), no identifier is added.
log_file	(Optional). Name of the log file to capture messages from VSEARCH. If NULL (default), no log file is created.
threads	(Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.
vsearch_options	(Optional). Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Sequences are clustered based on the pairwise identity threshold specified by id. Sequences are sorted by decreasing abundance before clustering. The centroid of each cluster is the first sequence added to the cluster.

fasta_input can either be a file path to a FASTA file or a FASTA object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta.

If neither centroids nor otutabout is specified (default), the function returns the centroid sequences as a FASTA object with an additional column otu_id. This column contains the identifier extracted from each sequence header.

If centroids is specified, centroid sequences are written to the specified file in FASTA format.

otutabout gives the option to output the results in an OTU table format with tab-separated columns. When writing to a file, the first line starts with the string "#OTU ID", followed by a tab-separated list of all sample identifiers (formatted as "sample=X"). Each subsequent line, corresponding to an OTU, begins with the OTU identifier and is followed by tab-separated abundances for that OTU in each sample. If otutabout is a character string, the output is written to the specified file. If otutabout is TRUE, the function returns the OTU table as a tibble, where the first column is named otu_id instead of "#OTU ID".

id is a value between 0 and 1 that defines the minimum pairwise identity required for a sequence to be added to a cluster. A sequence is not added to a cluster if its pairwise identity with the centroid is below the id threshold. Pairwise identity is calculated as the number of matching columns divided by the alignment length minus terminal gaps.

If log_file is NULL and centroids is specified, clustering statistics from VSEARCH will not be captured.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If centroids is specified the centroid sequences are written to the specified file, and no tibble is returned.

If otutabout is TRUE, an OTU table is returned as a tibble. If otutabout is a character string, the output is written to the file, and no tibble is returned.

If neither centroids nor otutabout is specified, a FASTA object with the centroid sequences and additional column otu_id is returned. The clustering statistics are included as an attribute named "statistics".

The "statistics" attribute of the returned tibble (when centroids is NULL) is a tibble with the following columns:

- num_nucleotides: Total number of nucleotides used as input for clustering.
- min_length_input_seq: Length of the shortest sequence used as input for clustering.
- max_length_input_seq: Length of the longest sequence used as input for clustering.
- avg_length_input_seq: Average length of the sequences used as input for clustering.
- num_clusters: Number of clusters generated.
- min_size_cluster: Size of the smallest cluster.
- max_size_cluster: Size of the largest cluster.
- avg_size_cluster: Average size of the clusters.
- num_singletons: Number of singletons after clustering.
- input: Name of the input file/object for the clustering.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Define arguments
fasta_input <- file.path(file.path(path.package("Rsearch"), "extdata"),
                                    "small.fasta")
centroids <- NULL

# Cluster sequences and return a FASTA tibble
cluster_seqs <- vs_cluster_size(fasta_input = fasta_input,
                                centroids = centroids)

# Extract clustering statistics
statistics <- attr(cluster_seqs, "statistics")

# Cluster sequences and write centroids to a file
vs_cluster_size(fasta_input = fasta_input,
                centroids = "centroids_sequences.fa")

## End(Not run)
```

---

vs_cluster_subseq *Cluster FASTA sequences*

Description

vs_cluster_subseq clusters FASTA sequences from a given file or object using VSEARCH´s cluster_fast method and 100 identity. The function automatically sorts sequences by decreasing length before clustering.

Usage

```
vs_cluster_subseq(
  fasta_input,
  centroids = NULL,
  strand = "plus",
  sizein = TRUE,
  fasta_width = 0,
  log_file = NULL,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fasta_input	(Required). A FASTA file path or a FASTA object containing reads to cluster. See *Details*.
centroids	(Optional). A character string specifying the name of the FASTA output file for the cluster centroid sequences. If NULL (default), no output is written to a file and the centroid sequences are returned as a FASTA object. See *Details*.
strand	(Optional). Specifies which strand to consider when comparing sequences. Can be either "plus" (default) or "both".
sizein	(Optional). If TRUE (default), abundance annotations present in sequence headers are taken into account.
fasta_width	(Optional). Number of characters per line in the output FASTA file. Defaults to 0, which eliminates wrapping.
log_file	(Optional). Name of the log file to capture messages from VSEARCH. If NULL (default), no log file is created.
threads	(Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.
vsearch_options	(Optional). Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

After merging/dereplication some sequences may be sub-sequences of longer sequences. This function will cluster such sequences at 100 (terminal gaps ignored), and keep the longest in each cluster as the centroid.

fasta_input can either be a file path to a FASTA file or a FASTA object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta.

If sizein = TRUE (default) the FASTA headers must contain text matching the regular expression "size=[0-9]+" indicating the copy number (=size) of each input sequence. This is then summed for each cluster and added to the output. This text is typically added by de-replication, see vs_fastx_uniques.

The number of distinct sequences in each cluster is output as members.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If centroids is specified the centroid sequences are written to the specified file, and no tibble is returned.

If centroids is not specified, a FASTA object is returned. This is a tibble with columns Header and Sequence, and also the additional column(s) members and, if sizein = TRUE, size.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Define arguments
fasta_input <- file.path(file.path(path.package("Rsearch"), "extdata"),
                                    "small.fasta")

# De-replicating
derep.tbl <- vs_fastx_uniques(fasta_input, output_format = "fasta")

# Clustering subsequences
cluster.tbl <- vs_cluster_subseq(fasta_input = derep.tbl)

# Cluster sequences and write centroids to a file
vs_cluster_subseq(fasta_input = derep.tbl,
                  centroids = "distinct.fa")

## End(Not run)
```

---

vs_cluster_unoise *Denoising FASTA sequences*

Description

vs_cluster_unoise performs denoising of FASTA sequences from a given file or object using VSEARCH´s cluster_unoise method.

Usage

```
vs_cluster_unoise(
  fasta_input,
  otutabout = NULL,
  minsize = 8,
  unoise_alpha = 2,
  relabel = NULL,
  relabel_sha1 = FALSE,
  log_file = NULL,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fasta_input	(Required). A FASTA file path or a FASTA object containing reads to denoise. See *Details*.
otutabout	(Optional). A character string specifying the name of the output file in an OTU table format. If NULL (default), the output is returned as a tibble in R. See *Details*.
minsize	(Optional). Minimum abundance of cluster centroids. Defaults to 8.
unoise_alpha	(Optional). Alpha value for the UNOISE algorithm. Defaults to 2.
relabel	(Optional). Relabel sequences using the given prefix and a ticker to construct new headers. Defaults to NULL.
relabel_sha1	(Optional). If TRUE (default), relabel sequences using the SHA1 message digest algorithm. Defaults to FALSE.
log_file	(Optional). Name of the log file to capture messages from VSEARCH. If NULL (default), no log file is created.
threads	(Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.
vsearch_options	(Optional). Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Sequences are denoised according to the UNOISE version 3 algorithm by Robert Edgar, but without the de novo chimera removal step. In this algorithm, clustering of sequences depends both on their similarity and their abundances. The abundance ratio (skew) is the abundance of a new sequence divided by the abundance of the centroid sequence. This skew must not be larger than beta if the sequences should be clustered together. Beta is calculated as 2 raised to the power of minus 1 minus alpha times the sequence distance. The sequence distance used is the number of mismatches in the alignment, ignoring gaps. This means that the abundance must be exponentially lower as the distance increases from the centroid for a new sequence to be included in the cluster.

The argument minsize will affect the total number of clusters, specifying the minimum copy number required for any centroid. A larger value means (in general) fewer clusters.

fasta_input can either be a file path to a FASTA file or a FASTA object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta.

The Header column **must** contain the size (copy number) for each read. The size information must have the format ";size=X", where X is the count for the given sequence. This is obtained by running all reads through vs_fastx_uniques with sizeout = TRUE.

You may use reads for a single sample or all reads from all samples as input. In the latter case the Header must also contain sample information on the format ";sample=xxx" where "xxx" is a unique sample identifier text. Again, this is obtained by using vs_fastx_uniques on the reads for each sample prior to this step. Use the sample = "xxx" argument, where "xxx" is replaced with some unique text for each sample.

If log_file is NULL and centroids is specified, clustering statistics from VSEARCH will not be captured.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A read count table with one row for each cluster and one column for each sample. If otutabout is a text it is assumed to be a file name, and the results are written to this file. If no such text is supplied (default), it is returned as a tibble.

The first two columns of this tibble lists the Header and Sequence of the centroid sequences for each cluster.

The clustering statistics are included as an attribute named "statistics" with the following columns:

- num_nucleotides: Total number of nucleotides used as input for clustering.
- min_length_input_seq: Length of the shortest sequence used as input for clustering.
- max_length_input_seq: Length of the longest sequence used as input for clustering.
- avg_length_input_seq: Average length of the sequences used as input for clustering.
- num_clusters: Number of clusters generated.
- min_size_cluster: Size of the smallest cluster.
- max_size_cluster: Size of the largest cluster.
- avg_size_cluster: Average size of the clusters.
- num_singletons: Number of singletons after clustering.
- input: Name of the input file/object for the clustering.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# A small fasta file
fasta_input <- file.path(file.path(path.package("Rsearch"), "extdata"), "small.fasta")

# Denoise sequences and read counts
denoised.tbl <- vs_cluster_unoise(fasta_input = fasta_input)
head(denoised.tbl)

# Extract clustering statistics
statistics <- attr(denoised.tbl, "statistics")

# Cluster sequences and write results to a file
vs_cluster_unoise(fasta_input = fasta_input,
                  otutabout = "otutable.tsv")

## End(Not run)
```

vs_fastq_join *Join paired-end sequence reads*

Description

vs_fastq_join joins paired-end sequence reads into a single sequence with a specified gap between them using VSEARCH.

Usage

```
vs_fastq_join(
  fastq_input,
  reverse = NULL,
  output_format = "fastq",
  fastaout = NULL,
  fastqout = NULL,
  join_padgap = "NNNNNNNN",
  join_padgapq = "IIIIIIII",
  fasta_width = 0,
  log_file = NULL,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fastq_input	(Required). A FASTQ file path, a FASTQ tibble object (forward reads), or a paired-end tibble of class "pe_df". See *Details*.
reverse	(Optional). A FASTQ file path or a FASTQ tibble object (reverse reads). Optional if fastq_input is a "pe_df" object. See *Details*.
output_format	(Optional). Desired output format of the file or tibble: "fasta" or "fastq" (default).
fastaout	(Optional). Name of the FASTA output file with the joined reads. If NULL (default), no output is written to a file. See *Details*.
fastqout	(Optional). Name of the FASTQ output file with the joined reads. If NULL (default), no output is written to a file. See *Details*.
join_padgap	(Optional). Padding sequence to use in the gap between the sequences. Defaults to "NNNNNNNN".
join_padgapq	(Optional). Quality of the padding sequence. Defaults to "IIIIIIII", corresponding to a base quality score of 40 (a very high quality score with error probability 0.0001).
fasta_width	(Optional). Number of characters per line in the output FASTA file. Only applies if the output file is in FASTA format. Defaults to 0, which eliminates wrapping.

log_file (Optional). Name of the log file to capture messages from VSEARCH. If NULL (default), no log file is created.

threads (Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.

vsearch_options (Optional). Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.

tmpdir (Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Read pairs from the input FASTQ files (fastq_input and reverse) are joined into a single sequence by adding a gap with a specified padding sequence. The resulting sequences consist of the forward read, the padding sequence, and the reverse complement of the reverse read.

fastq_input and reverse can either be file paths to FASTQ files or FASTQ objects. FASTQ objects are tibbles that contain the columns Header, Sequence, and Quality, see readFastq. Forward and reverse reads must appear in the same order and have the same total number of reads in both files.

If fastq_input is an object of class "pe_df", the reverse reads are automatically extracted from its "reverse" attribute unless explicitly provided via the reverse argument. This simplifies function calls when using paired-end tibbles created by functions such as fastx_synchronize or vs_fastx_trim_filt.

If fastaout or fastqout is specified, the joined reads are written to the respective file in either FASTA or FASTQ format.

If both fastaout or fastqout are NULL, the results are returned as a FASTA or FASTQ object, and no file is written. output_format must match the desired output files/objects.

Any input sequence with fewer bases than the value set in minlen is discarded. By default, minlen is set to 0, which means that no sequences are removed. However, using the default value may allow empty sequences to remain in the results.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If fastaout or fastqout is specified, the joined sequences are written to the specified output file, and no tibble is returned.

If fastaout or fastqout is NULL, a tibble containing the joined reads in the format specified by output_format is returned.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Define arguments
fastq_input <- file.path(file.path(path.package("Rsearch"), "extdata"),
                         "small_R1.fq")
reverse <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small_R2.fq")
output_format <- "fastq"

# Execute joining and return a FASTQ tibble
join_seqs <- vs_fastq_join(fastq_input = fastq_input,
                           reverse = reverse,
                           output_format = output_format)

# Execute joining and write joined sequences to file
vs_fastq_join(fastq_input = fastq_input,
              reverse = reverse,
              fastqout = "joined_sequences.fq",
              output_format = output_format)


## End(Not run)
```

---

vs_fastq_mergepairs *Merge paired-end sequence reads*

Description

vs_fastq_mergepairs merges paired-end sequence reads with overlapping regions into one sequence using VSEARCH.

Usage

```
vs_fastq_mergepairs(
  fastq_input,
  reverse = NULL,
  output_format = "fasta",
  fastaout = NULL,
  fastqout = NULL,
  minovlen = 10,
  minlen = 0,
  fasta_width = 0,
  sample = NULL,
  log_file = NULL,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fastq_input	(Required). A FASTQ file path, a FASTQ tibble (forward reads), or a paired-end tibble of class "pe_df". See *Details*.
reverse	(Optional). A FASTQ file path or a FASTQ tibble (reverse reads). Optional if fastq_input is a "pe_df" object. See *Details*.
output_format	(Optional). Desired output format of file or tibble: "fasta" (default) or "fastq".
fastaout	(Optional). Name of the FASTA output file with the merged reads. If NULL (default), no output is written to file. See *Details*.
fastqout	(Optional). Name of the FASTQ output file with the merged reads. If NULL (default) no output is written to file. See *Details*.
minovlen	(Optional). Minimum overlap between the merged reads. Must be at least 5. Defaults to 10.
minlen	(Optional). Minimum number of bases a sequence must have to be retained. Defaults to 0. See *Details*.
fasta_width	(Optional). Number of characters per line in the output FASTA file. Only applies if the output file is in FASTA format. Defaults to 0, which eliminates wrapping.
sample	(Optional). Add the given sample identifier string to sequence headers. For instance, if the given string is "ABC", the text ";sample=ABC" will be added to the header. If NULL (default), no identifier is added.
log_file	(Optional). Name of the log file to capture messages from VSEARCH. If NULL (default), no log file is created.
threads	(Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.
vsearch_options	(Optional). Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Read pairs from the input FASTQ files (fastq_input and reverse) are merged into a single sequence by overlapping regions. The resulting sequences consist of the merged forward and reverse reads with the specified minimum overlap.

fastq_input and reverse can either be file paths to FASTQ files or FASTQ objects. FASTQ objects are tibbles that contain the columns Header, Sequence, and Quality, see readFastq. Forward and reverse reads must appear in the same order and have the same total number of reads in both files.

If fastq_input is an object of class "pe_df", the reverse reads are automatically extracted from its "reverse" attribute unless explicitly provided via the reverse argument. This allows streamlined input handling for paired-end tibbles created by fastx_synchronize or vs_fastx_trim_filt.

If fastaout or fastqout is specified, the merged reads are written to the respective file in either FASTA or FASTQ format.

If both fastaout or fastqout are NULL, the results are returned as a FASTA or FASTQ object, and no file is written.

output_format has to match the desired output files/objects.

Any input sequence with fewer bases than the value set in minlen will be discarded. Default minlen is 0, meaning no sequences are removed. However, using the default value may allow empty sequences to remain in the results.

If log_file is NULL and fastqout or fastaout is specified, merging statistics from VSEARCH will not be captured.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If fastaout or fastqout is specified , the merged sequences are written to the specified output file, and no tibble is returned.

If fastaout or fastqout is NULL, a tibble containing the merged reads in the format specified by output_format is returned.

The "statistics" attribute of the returned tibble (when fastaout or fastqout is NULL) is a tibble with the following columns:

- Tot_num_pairs: Total number of read pairs before merging.
- Merged: Number of read pairs that merged.
- Mean_Read_Length_before_merging: Mean read length before merging (R1 and R2).
- Mean_Read_Length_after_merging: Mean read length after merging.
- StdDev_Read_Length: Standard deviation of read length after merging.
- R1: Name of the file/object with forward (R1) reads used in the merging.
- R2: Name of the file/object with reverse (R2) reads used in the merging.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Define arguments
fastq_input <- file.path(file.path(path.package("Rsearch"), "extdata"),
                         "small_R1.fq")
reverse <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small_R2.fq")
output_format <- "fastq"

# Merge sequences and return a FASTQ tibble
merge_seqs <- vs_fastq_mergepairs(fastq_input = fastq_input,
                                  reverse = reverse,
```

```
                                output_format = output_format)

# Extract merging statistics
statistics <- attr(merge_seqs, "statistics")

# Merge sequences and write sequences to a FASTQ file
vs_fastq_mergepairs(fastq_input = fastq_input,
                    reverse = reverse,
                    output_format = output_format,
                    fastqout = "merged_sequences.fq")

## End(Not run)
```

`vs_fastx_subsample` *Subsample sequences*

Description

`vs_fastx_subsample` subsamples sequences in FASTA/FASTQ file or object by randomly extracting sequences based on number or percentage using `VSEARCH`.

Usage

```
vs_fastx_subsample(
  fastx_input,
  output_format = "fastq",
  fastx_output = NULL,
  sample_pct = NULL,
  sample_size = NULL,
  sizein = TRUE,
  sizeout = TRUE,
  relabel = NULL,
  relabel_sha1 = FALSE,
  randseed = NULL,
  fasta_width = 0,
  sample = NULL,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

`fastx_input`	(Required). A FASTA/FASTQ file path or FASTA/FASTQ object. See *Details*.
`output_format`	(Optional). Desired output format of file or tibble: `"fasta"` or `"fastq"` (default). If `fastx_input` is a FASTA file path or a FASTA object, `output_format` cannot be `"fastq"`.

fastx_output	(Optional). Name of the output file for subsampled reads from fastx_input. File can be in either FASTA or FASTQ format, depending on output_format. If NULL (default), no sequences are written to file. See *Details*.
sample_pct	(Optional). Percentage of the input sequences to be subsampled. Numeric value ranging from 0.0 to 100.0. Defaults to NULL.
sample_size	(Optional). The given number of sequences to extract. Must be a positive integer if specified. Defaults to NULL.
sizein	(Optional). If TRUE (default), abundance annotations present in sequence headers are taken into account.
sizeout	(Optional). If TRUE (default), abundance annotations are added to FASTA headers.
relabel	(Optional). Relabel sequences using the given prefix and a ticker to construct new headers. Defaults to NULL.
relabel_sha1	(Optional). If TRUE (default), relabel sequences using the SHA1 message digest algorithm. Defaults to FALSE.
randseed	(Optional). Random seed. Must be a positive integer. A given seed always produces the same output, which is useful for replicability. Defaults to NULL.
fasta_width	(Optional). Number of characters per line in the output FASTA file. Defaults to 0, which eliminates wrapping.
sample	(Optional). Add the given sample identifier string to sequence headers. For instance, if the given string is "ABC", the text ";sample=ABC" will be added to the header. If NULL (default), no identifier is added.
threads	(Optional). Number of computational threads to be used by VSEARCH.Defaults to 1.
vsearch_options	Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Sequences in the input file/object (fastx_input) are subsampled by randomly extracting a specified number or percentage of sequences. Extraction is performed as random sampling with a uniform distribution among the input sequences and without replacement.

fastx_input can either be a FASTA/FASTQ file or a FASTA/FASTQ object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta. FASTQ objects are tibbles that contain the columns Header, Sequence, and Quality, see readFastq.

Specify either sample_size or sample_pct to determine the number or percentage of sequences to subsample. Only one of these parameters can be specified at a time. If neither is specified, an error is thrown.

If fastx_output is specified, the sampled sequences are output to this file in format given by output_format. If fastx_output is NULL, the sample sequences are returned as a FASTA or FASTQ object, depending on output_format.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If fastx_output is specified, the subsampled sequences are written to the specified output file, and no tibble is returned.

If fastx_output NULL, a tibble containing the subsampled reads in the format specified by output_format is returned.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Define arguments
fastx_input <- file.path(file.path(path.package("Rsearch"), "extdata"),
                         "small_R1.fq")
fastx_output <- NULL
output_format <- "fastq"
sample_size <- 10

# Subsample sequences and return a FASTQ tibble
subsample_R1 <- vs_fastx_subsample(fastx_input = fastx_input,
                                   fastx_output = fastx_output,
                                   output_format = output_format,
                                   sample_size = sample_size)

# Subsample sequences and write subsampled sequences to a file
vs_fastx_subsample(fastx_input = fastx_input,
                   fastx_output = "subsample.fq",
                   output_format = output_format,
                   sample_size = sample_size)

## End(Not run)
```

---

vs_fastx_trim_filt *Trim and/or filter sequences in FASTA/FASTQ format*

Description

vs_fastx_trim_filt trims and/or filters FASTA/FASTQ sequences using VSEARCH. This function processes both forward and reverse reads (if provided) and allows for various filtering criteria based on sequence quality, length, abundance, and more.

Usage

```
vs_fastx_trim_filt(
  fastx_input,
  reverse = NULL,
  output_format = "fastq",
  fastaout = NULL,
  fastqout = NULL,
  fastaout_rev = NULL,
  fastqout_rev = NULL,
  trunclen = NULL,
  truncqual = 1,
  truncee = NULL,
  truncee_rate = NULL,
  stripright = 0,
  stripleft = 0,
  maxee_rate = 0.01,
  minlen = 0,
  maxlen = NULL,
  maxns = 0,
  minsize = NULL,
  maxsize = NULL,
  minqual = 0,
  relabel = NULL,
  relabel_sha1 = FALSE,
  fasta_width = 0,
  sample = NULL,
  stats = TRUE,
  log_file = NULL,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

`fastx_input`	(Required). A FASTA/FASTQ file path or FASTA/FASTQ object containing (forward) reads. See *Details*.
`reverse`	(Optional). A FASTA/FASTQ file path or object containing reverse reads. If `fastx_input` is a `"pe_df"` object and `reverse` is not provided, the reverse reads will be extracted from its `"reverse"` attribute.
`output_format`	(Optional). Desired output format of file or tibble: `"fasta"` or `"fastq"` (default). If `fastx_input` is a FASTA file path or a FASTA object, `output_format` cannot be `"fastq"`.
`fastaout`	(Optional). Name of the FASTA output file for the sequences given in `fastx_input`. If `NULL` (default), no FASTA sequences are written to file. See *Details*.
`fastqout`	(Optional). Name of the FASTQ output file for the sequences given in `fastx_input`. If `NULL` (default), no FASTQ sequences are written to file. See *Details*.

`fastaout_rev` (Optional). Name of the FASTA output file for the reverse sequences. If `NULL` (default), no FASTA sequences are written to file. See *Details*.

`fastqout_rev` (Optional). Name of the FASTQ output file for the reverse sequences. If `NULL` (default), no FASTQ sequences are written to file. See *Details*.

`trunclen` (Optional). Truncate sequences to the specified length. Shorter sequences are discarded. If `NULL` (default), the trimming is not applied.

`truncqual` (Optional). Truncate sequences starting from the first base with a quality score of the specified value or lower. Defaults to `1`.

`truncee` (Optional). Truncate sequences so that their total expected error does not exceed the specified value. If `NULL` (default), the trimming is not applied.

`truncee_rate` (Optional). Truncate sequences so that their average expected error per base is not higher than the specified value. The truncation will happen at first occurrence. The average expected error per base is calculated as the total expected number of errors divided by the length of the sequence after truncation. If `NULL` (default), the trimming is not applied.

`stripright` (Optional). Number of bases stripped from the right end of the reads. Defaults to `0`.

`stripleft` (Optional). Number of bases stripped from the left end of the reads. Defaults to `0`.

`maxee_rate` (Optional). Threshold for average expected error. Numeric value ranging form `0.0` to `1.0`. Defaults to `0.01`. See *Details*.

`minlen` (Optional). Minimum number of bases a sequence must have to be retained. Defaults to `0`. See *Details*.

`maxlen` (Optional). Maximum number of bases a sequences can have to be retained. If `NULL` (default), the filter is not applied.

`maxns` (Optional). Maximum number of N's for a given sequence. Sequences with more N's than the specified number are discarded. Defaults to `0`.

`minsize` (Optional). Minimum abundance for a given sequence. Sequences with lower abundance are discarded. If `NULL` (default), the filter is not applied.

`maxsize` (Optional). Maximum abundance for a given sequence. Sequences with higher abundance are discarded. If `NULL` (default), the filter is not applied.

`minqual` (Optional). Minimum base quality for a read to be retained. A read is discarded if it contains bases with a quality score below the given value. Defaults to `0`, meaning no reads are discarded.

`relabel` (Optional). Relabel sequences using the given prefix and a ticker to construct new headers. Defaults to `NULL`.

`relabel_sha1` (Optional). If `TRUE` (default), relabel sequences using the SHA1 message digest algorithm. Defaults to `FALSE`.

`fasta_width` (Optional). Number of characters per line in the output FASTA file. Defaults to `0`, which eliminates wrapping.

`sample` (Optional). Add the given sample identifier string to sequence headers. For instance, if the given string is "ABC", the text ";sample=ABC" will be added to the header. If `NULL` (default), no identifier is added.

stats (Optional). If TRUE (default), a tibble with statistics about the filtering is added as an attribute of the returned tibble. If FALSE, no statistics are added.

log_file (Optional). Name of the log file to capture messages from VSEARCH. If NULL (default), no log file is created.

threads (Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.

vsearch_options (Optional). Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.

tmpdir (Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Reads from the input files/objects (fastx_input and reverse) are trimmed and/or filtered based on the specified criteria using VSEARCH.

fastx_input and reverse can either be file paths to FASTA/FASTQ files or FASTA/FASTQ objects. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta. FASTQ objects are tibbles that contain the columns Header, Sequence, and Quality, see readFastq.

If fastx_input is an object of class "pe_df", the reverse reads are automatically extracted from its "reverse" attribute unless explicitly provided via the reverse argument.

If reverse is provided, it is processed alongside fastx_input using the same trimming/filtering criteria.

Note that if you want to trim/filter the forward and reverse reads differently, you must pass them separately to this function, get two result files/objects, and then use fastx_synchronize to synchronize the read pairs again.

If fastaout and fastaout_rev or fastqout and fastqout_rev are specified, trimmed and/or filtered sequences are written to these files in the specified format.

If output files are NULL, results are returned as a tibbles. When returning tibbles, the reverse sequences (if provided) are attached as an attribute named "reverse".

When reverse reads are returned as an attribute, the primary tibble is also assigned the S3 class "pe_df" to indicate that it represents paired-end data. This class tag can be used by downstream tools to recognize paired-end tibbles.

Note that certain options are not compatible with both file formats. For instance, options that trim or filter sequences based on quality scores are unavailable when the input is of type "fasta". Visit the VSEARCH documentation for more details.

Sequences with an average expected error greater than the specified maxee_rate are discarded. For a given sequence, the average expected error is the sum of error probabilities for all the positions in the sequence, divided by the length of the sequence.

Any input sequence with fewer bases than the value set in minlen will be discarded. By default, minlen is set to 0, which means that no sequences are removed. However, using the default value may allow empty sequences to remain in the results.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If output files are specified, the results are written directly to the specified output files, and no tibble is returned.

If output files (fastaout/fastqout and fastaout_rev/fastqout_rev) are NULL, a tibble containing the trimmed and/or filtered reads from fastx_input in the format specified by output_format is returned.

If reverse is provided, a tibble containing the trimmed and/or filtered reverse sequences is attached as an attribute, named "reverse" to the returned table.

When the reverse reads are present, the returned tibble is assigned the class "pe_df", identifying it as paired-end data.

The "statistics" attribute of the returned tibble (when output files are NULL) is a tibble with the following columns:

- Kept_Sequences: Number of retained sequences.
- Discarded_Sequences: Number of discarded sequences.
- fastx_source: Name of the file/object with forward (R1) reads.
- reverse_source: (If reverse is specified) Name of the file/object with reverse (R2) reads.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Define arguments
fastx_input <- file.path(file.path(path.package("Rsearch"), "extdata"),
                         "small_R1.fq")
reverse <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small_R1.fq")
output_format <- "fastq"
maxee_rate <- 0.01
minlen <- 0

# Trim/filter sequences and return a FASTQ tibble
filt_seqs <- vs_fastx_trim_filt(fastx_input = fastx_input,
                                reverse = reverse,
                                output_format = output_format,
                                maxee_rate = maxee_rate,
                                minlen = minlen)

# Extract tibbles
R1_filt <- filt_seqs
R2_filt <- attr(filt_seqs, "reverse")

# Extract filtering statistics
statistics <- attr(filt_seqs, "statistics")
```

```
# Trim/filter sequences and write results to FASTQ files
vs_fastx_trim_filt(fastx_input = fastx_input,
                   reverse = reverse,
                   fastqout = "filt_R1.fq",
                   fastqout_rev = "filt_R2.fq",
                   output_format = output_format,
                   maxee_rate = maxee_rate,
                   minlen = minlen)

## End(Not run)
```

`vs_fastx_uniques` *Dereplicate sequences*

Description

`vs_fastx_uniques` performs dereplication of sequences in a FASTA/FASTQ file or object by merging identical sequences using `VSEARCH`.

Usage

```
vs_fastx_uniques(
  fastx_input,
  output_format = "fasta",
  fastx_output = NULL,
  minuniquesize = 1,
  strand = "plus",
  sizein = TRUE,
  sizeout = TRUE,
  relabel = NULL,
  relabel_sha1 = FALSE,
  fastq_qout_max = FALSE,
  fasta_width = 0,
  sample = NULL,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

`fastx_input`	(Required). A FASTA/FASTQ file path or FASTA/FASTQ object. See *Details*.
`output_format`	(Optional). Desired output format of file or tibble: `"fasta"` (default) or `"fastq"`. If `fastx_input` is a FASTA file path or a FASTA object, `output_format` cannot be `"fastq"`.

fastx_output	(Optional). Name of the output file for dereplicated reads from fastx_input. File can be in either FASTA or FASTQ format, depending on output_format. If NULL (default), no sequences are written to file. See *Details*.
minuniquesize	(Optional). Minimum abundance value post-dereplication for a sequence not to be discarded. Defaults to 1.
strand	(Optional). Specifies which strand to consider when comparing sequences. Can be either "plus" (default) or "both".
sizein	(Optional). If TRUE (default), abundance annotations present in sequence headers are taken into account.
sizeout	(Optional). If TRUE (default), abundance annotations are added to FASTA headers.
relabel	(Optional). Relabel sequences using the given prefix and a ticker to construct new headers. Defaults to NULL.
relabel_sha1	(Optional). If TRUE (default), relabel sequences using the SHA1 message digest algorithm. Defaults to FALSE.
fastq_qout_max	(Optional). If TRUE, the quality score will be the highest (best) quality score observed in each position. Defaults to FALSE.
fasta_width	(Optional). Number of characters per line in the output FASTA file. Defaults to 0, which eliminates wrapping.
sample	(Optional). Add the given sample identifier string to sequence headers. For nstance, if the given string is "ABC", the text ";sample=ABC" will be added to the header. If NULL (default), no identifier is added.
vsearch_options	(Optional). A character string of additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Sequences in the input file/object (fastx_input) are dereplicated by merging identical sequences. Identical sequences are defined as sequences with the same length and the same string of nucleotides (case insensitive, T and U are considered the same).

fastx_input can either be a FASTA/FASTQ file or a FASTA/FASTQ object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta. FASTQ objects are tibbles that contain the columns Header, Sequence, and Quality, see readFastq.

By default, the quality scores in FASTQ output files will correspond to the average error probability of the nucleotides in the each position. If fastq_qout_max = TRUE, the quality score will be the highest (best) quality score observed in each position.

If fastx_output is specified, the dereplicated sequences are output to this file in format given by output_format. If fastx_output is NULL, the dereplicated sequences are returned as a FASTA or FASTQ object, depending on output_format.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If fastx_output is specified, the dereplicated sequences are written to the specified output file, and no tibble is returned.

If fastx_output NULL, a tibble containing the dereplicated reads in the format specified by output_format is returned.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Define arguments
fastx_input <- file.path(file.path(path.package("Rsearch"), "extdata"),
                         "small_R1.fq")
fastx_output <- NULL
output_format <- "fastq"

# Dereplicate sequences and return a FASTQ tibble
derep_R1 <- vs_fastx_uniques(fastx_input = fastx_input,
                             fastx_output = fastx_output,
                             output_format = output_format)

# Dereplicate sequences and write derelicated sequences to a file
vs_fastx_uniques(fastx_input = fastx_input,
                 fastx_output = "dereplicated_sequences.fq",
                 output_format = output_format)

## End(Not run)
```

---

vs_merging_lengths *Length statistics after merging*

Description

vs_merging_lengths computes length statistics for forward reads, reverse reads, merged reads, and their overlaps before and after merging.

Usage

```
vs_merging_lengths(
  fastq_input,
  reverse = NULL,
  minovlen = 10,
  minlen = 0,
```

```
  threads = 1,
  plot_title = TRUE,
  tmpdir = NULL
)
```

Arguments

`fastq_input`	(Required). A FASTQ file path, a FASTQ tibble (forward reads), or a paired-end tibble of class `"pe_df"`. See *Details*.
`reverse`	(Optional). A FASTQ file path or FASTQ tibble containing reverse reads. Optional if `fastq_input` is a `"pe_df"` object.
`minovlen`	(Optional). Minimum overlap between the merged reads. Must be at least 5. Defaults to `10`.
`minlen`	(Optional). Minimum number of bases a sequence must have to be retained. Defaults to `0`. See *Details*.
`threads`	(Optional). Number of computational threads to be used by `VSEARCH`. Defaults to `1`.
`plot_title`	(Optional). If `TRUE` (default), a summary title will be displayed in the plot. Set to `FALSE` for no title.
`tmpdir`	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to `NULL`, which resolves to the session-specific temporary directory (`tempdir()`).

Details

The function uses `vs_fastq_mergepairs` where the arguments to this function are described in detail.

If `fastq_input` is an object of class `"pe_df"`, the reverse reads are automatically extracted from its `"reverse"` attribute unless explicitly provided via the `reverse` argument. This allows streamlined input handling for paired-end tibbles created by `fastx_synchronize` or `vs_fastx_trim_filt`.

These length statistics are most typically used in order to tune the filter and trimming of reads such that the merged reads are of high quality.

Value

A tibble with the following columns:

- `length_1`: The length of the forward reads.
- `length_2`: The length of the reverse reads.
- `length_merged`: The length of the merged reads.
- `length_overlap`: The length of the overlap between the forward and reverse reads.

In case of missing values for the latter two columns, it means that the corresponding reads were not merged.

The tibble includes additional attributes:

`plot` A `ggplot2` object visualizing the returned data frame.

`statistics` Additional statistics returned from `vs_fastq_mergepairs`.

References

https://github.com/torognes/vsearch

See Also

vs_fastq_mergepairs

Examples

```
## Not run:
# Define arguments
R1.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small_R1.fq")
R2.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small_R2.fq")

# Run function
merging.tbl <- vs_merging_lengths(fastq_input = R1.file,
                                  reverse = R2.file)

# Display plot
merging_stats_plot <- attr(merging.tbl, "plot")
print(merging_stats_plot)


## End(Not run)
```

`vs_optimize_truncee_rate` *Optimize read truncation with truncee_rate*

Description

`vs_optimize_truncee_rate` optimizes the truncation parameter `truncee_rate` to achieve the best possible merging results. The function iterates through a specified range of `truncee_rate` values to identify the optimal value that maximizes the proportion of high-quality merged read pairs.

Usage

```
vs_optimize_truncee_rate(
  fastq_input,
  reverse = NULL,
  minovlen = 10,
  truncee_rate_range = c(seq(0.002, 0.04, by = 0.002)),
  minlen = 1,
  min_size = 2,
  maxee_rate = 0.01,
  threads = 1,
```

```
  plot_title = TRUE,
  tmpdir = NULL
)
```

Arguments

`fastq_input`	(Required). A FASTQ file path, FASTQ tibble (forward reads), or a paired-end tibble of class `"pe_df"`. See *Details*.
`reverse`	(Optional). A FASTQ file path or FASTQ tibble (reverse reads). Optional if `fastq_input` is a `"pe_df"` object.
`minovlen`	(Optional). Minimum overlap between the merged reads. Must be at least 5. Defaults to `10`.
`truncee_rate_range`	(Optional). A numeric vector of `truncee_rate` values to test. Sequences are truncated so that their average expected error per base is lower than the specified value. Defaults to (`0.002`, `0.004`,`0.006`, `0.008`, `0.010`, `0.012`, `0.014`, `0.016`, `0.018`, `0.020`, `0.022`, `0.024`, `0.026`,`0.028`, `0.030`, `0.032`, `0.034`, `0.036`, `0.038`, `0.040`).
`minlen`	(Optional). Minimum number of bases a sequence must have to be retained. Defaults to `0`. See *Details*.
`min_size`	(Optional). Minimum copy number (size) for a merged read to be included in the results. Defaults to `2`.
`maxee_rate`	(Optional). Threshold for average expected error. Must range from `0.0` to `1.0`. Defaults to `0.01`. See *Details*.
`threads`	(Optional). Number of computational threads to be used by `VSEARCH`. Defaults to `1`.
`plot_title`	(Optional). If `TRUE` (default), a summary title will be displayed in the plot. Set to `FALSE` for no title.
`tmpdir`	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to `NULL`, which resolves to the session-specific temporary directory (`tempdir()`).

Details

The function uses `vs_fastq_mergepairs`, `vs_fastx_trim_filt`, and `vs_fastx_uniques` where the arguments to this functions are described in detail.

If `fastq_input` has class `"pe_df"`, the reverse reads will be automatically extracted from the `"reverse"` attribute unless explicitly provided in the `reverse` argument.

The best possible truncation option (`truncee_rate`) for merging is measured by the number of merged read-pairs with a copy number above the number specified by `min_size` after dereplication.

Changing `min_size` will affect the results. A low `min_size` will include merged sequences with a lower copy number after dereplication, and a higher `min_size` will filter out more reads and only count high-frequency merged sequences.

Value

A data frame with the following columns:

- `truncee_rate_value`: Tested `truncee_rate` value.
- `merged_read_pairs`: Count of merged read-pairs with a copy number above `min_size` after dereplication.
- `R1_length`: Average length of R1-reads after trimming.
- `R2_length`: Average length of R2-reads after trimming.

The returned data frame has an attribute named `"plot"` containing a ggplot2 object based on the returned data frame. The plot visualizes `truncee_rate` values against `merged_read_pairs`, `R1_length`, and `R2_length`, with the optimal `truncee_rate` value marked by a red dashed line.

Additionally, the returned data frame has an attribute named `"optimal_truncee_rate"` containing the optimal `truncee_rate` value.

References

https://github.com/torognes/vsearch

See Also

vs_fastq_mergepairs, vs_fastx_trim_filt, vs_fastx_uniques

Examples

```
## Not run:
# Define arguments
R1.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small_R1.fq")
R2.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small_R2.fq")

# Run optimizing function
optimize.tbl <- vs_optimize_truncee_rate(fastq_input = R1.file,
                                         reverse = R2.file)

# Display plot
print(attr(optimize.tbl, "plot"))


## End(Not run)
```

vs_optimize_truncqual *Optimize read truncation with truncqual*

Description

vs_optimize_truncqual optimizes the truncation parameter truncqual to achieve the best possible merging results. The function iterates through a specified range of truncqual values to identify the optimal value that maximizes the proportion of high-quality merged read pairs.

Usage

```
vs_optimize_truncqual(
  fastq_input,
  reverse = NULL,
  minovlen = 10,
  truncqual_range = 1:20,
  minlen = 1,
  min_size = 2,
  maxee_rate = 0.01,
  threads = 1,
  plot_title = TRUE,
  tmpdir = NULL
)
```

Arguments

fastq_input	(Required). A FASTQ file path, FASTQ tibble (forward reads), or a paired-end tibble of class "pe_df". See *Details*.
reverse	(Optional). A FASTQ file path or FASTQ tibble (reverse reads). Optional if fastq_input is a "pe_df" object.
minovlen	(Optional). Minimum overlap between the merged reads. Must be at least 5. Defaults to 10.
truncqual_range	(Optional). A numeric vector of truncqual values to test. Sequences are truncated starting from the first base with the specified base quality score or lower. Defaults to 1:20.
minlen	(Optional). Minimum number of bases a sequence must have to be retained. Defaults to 0. See *Details*.
min_size	(Optional). Minimum copy number (size) for a merged read to be included in the results. Defaults to 2.
maxee_rate	(Optional). Threshold for average expected error. Must range from 0.0 to 1.0. Defaults to 0.01. See *Details*.
threads	(Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.

plot_title (Optional). If TRUE (default), a summary title will be displayed in the plot. Set to FALSE for no title.

tmpdir (Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

The function uses vs_fastq_mergepairs, vs_fastx_trim_filt, and vs_fastx_uniques where the arguments to this functions are described in detail.

If fastq_input has class "pe_df", the reverse reads will be automatically extracted from the "reverse" attribute unless explicitly provided in the reverse argument.

The best possible truncation option (truncqual) for merging is measured by the number of merged read-pairs with a copy number above the number specified by min_size after dereplication.

Changing min_size will affect the results. A low min_size will include merged sequences with a lower copy number after dereplication, and a higher min_size will filter out more reads and only count high-frequency merged sequences.

Value

A data frame with the following columns:

- truncqual_value: Tested truncqual value.
- merged_read_pairs: Count of merged read-pairs with a copy number above min_size after dereplication.
- R1_length: Average length of R1-reads after trimming.
- R2_length: Average length of R2-reads after trimming.

The returned data frame has an attribute named "plot" containing a ggplot2 object based on the returned data frame. The plot visualizes truncqual values against merged_read_pairs, R1_length, and R2_length, with the optimal truncqual value marked by a red dashed line.

Additionally, the returned data frame has an attribute named "optimal_truncqual" containing the optimal truncqual value.

References

https://github.com/torognes/vsearch

See Also

vs_fastq_mergepairs, vs_fastx_trim_filt, vs_fastx_uniques

Examples

```
## Not run:
# Define arguments
R1.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small_R1.fq")
```

```
R2.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small_R1.fq")

# Run optimizing function
optimize.tbl <- vs_optimize_truncqual(fastq_input = R1.file,
                                      reverse = R2.file)

# Display plot
print(attr(optimize.tbl, "plot"))


## End(Not run)
```

---

vs_search_exact *Search for exact full-length matches*

Description

vs_search_exact searches for exact full-length matches of query sequences in a database of target sequences using VSEARCH.

Usage

```
vs_search_exact(
  fastx_input,
  database,
  userout = NULL,
  otutabout = NULL,
  userfields = "query+target+id+alnlen+mism+opens+qlo+qhi+tlo+thi+evalue+bits",
  strand = "plus",
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fastx_input	(Required). A FASTA/FASTQ file path or FASTA/FASTQ tibble object containing the query sequences. See *Details*.
database	(Required). A FASTA/FASTQ file path or FASTA/FASTQ tibble object containing the target sequences.
userout	(Optional). A character string specifying the name of the output file for the alignment results. If NULL (default), no output is written to a file and the results are returned as a tibble with the columns specified in userfields. See *Details*.
otutabout	(Optional). A character string specifying the name of the output file in an OTU table format. If NULL (default), no output is written to a file. If TRUE, the output is returned as a tibble. See *Details*.

userfields (Optional). Fields to include in the output file. Defaults to "query+target+id+alnlen+mism+opens+qlo
See *Details*.

strand (Optional). Specifies which strand to consider when comparing sequences. Can be either "plus" (default) or "both".

threads (Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.

vsearch_options (Optional). A character string of additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.

tmpdir (Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Identifies exact full-length matches between query and target sequences using VSEARCH. Only 100 specificity and making this command much faster than vs_usearch_global.

fastx_input and database can either be file paths to a FASTA/FASTQ files or FASTA/FASTQ objects. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta. FASTQ objects are tibbles that contain the columns Header, Sequence, and Quality, see readFastq.

userfields specifies the fields to include in the output file. Fields must be given as a character string separated by "+". The default value of userfields equals "query+target+id+alnlen+mism+opens+qlo+qhi+tlo+t which gives a blast-like tab-separated format of twelve fields. See the 'Userfields' section in the VSEARCH manual for more information.

otutabout gives the option to output the results in an OTU table format with tab-separated columns. When writing to a file, the first line starts with the string "#OTU ID", followed by a tab-separated list of all sample identifiers (formatted as "sample=X"). Each subsequent line, corresponding to an OTU, begins with the OTU identifier and is followed by tab-separated abundances for that OTU in each sample. If otutabout is a character string, the output is written to the specified file. If otutabout is TRUE, the function returns the OTU table as a tibble, where the first column is named otu_id instead of "#OTU ID".

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If userout is NULL a tibble containing the alignment results with the fields specified by userfields is returned. If userout is specified the alignment results are written to the specified file, and no tibble is returned.

If otutabout is TRUE, an OTU table is returned as a tibble. If otutabout is a character string, the output is written to the file, and no tibble is returned.

If neither userout nor otutabout is specified, a tibble containing the alignment results is returned.

References

https://github.com/torognes/vsearch

See Also

vs_usearch_global

Examples

```
## Not run:
# You would typically use something else as database
query_file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                        "small.fasta")
db <- query_file

# Search for exact full-length matches with default parameters, with file as output
vs_search_exact(fastx_input = query_file,
                database = db,
                userout = "delete_me.txt")

# Read results, and give column names
result.tbl <- read.table("delete_me.txt",
                         sep = "\t",
                         header = FALSE,
                         col.names = c("query", "target", "id", "alnlen",
                                       "mism", "opens", "qlo", "qhi",
                                       "tlo", "thi", "evalue", "bits"))

## End(Not run)
```

---

vs_sintax *Taxonomic classification using the Sintax algorithm*

Description

vs_sintax classifies sequences using the Sintax algorithm implemented in VSEARCH.

Usage

```
vs_sintax(
  fasta_input,
  database,
  outfile = NULL,
  cutoff = 0,
  strand = "plus",
  sintax_random = TRUE,
  randseed = NULL,
  logfile = NULL,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fasta_input	(Required). A FASTA file path or a FASTA object with reads to classify, see *Details*.
database	(Required). A FASTA file path or a FASTA object containing the reference database in FASTA format. The sequences need to be annotated with taxonomy, see *Details*.
outfile	(Optional). Name of the output file. If NULL (default), results are returned as a data.frame.
cutoff	(Optional). Minimum level of bootstrap support (0.0-1.0) for the classifications. Defaults to 0.0.
strand	(Optional). Specifies which strand to consider when comparing sequences. Can be either "plus" (default) or "both".
sintax_random	(Optional). If TRUE (default), the Sintax algorithm breaks ties between sequences with equally many kmer matches by a random draw.
randseed	(Optional). Seed for the random number generator used in the Sintax algorithm. Defaults to NULL.
logfile	(Optional). Name of the log file to capture messages from VSEARCH. If NULL (default), no log file is created.
threads	(Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.
vsearch_options	(Optional). A character string of additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

The sequences in the input file are classified according to the Sintax algorithm, using VSEARCH, see https://www.biorxiv.org/content/10.1101/074161v1.

fasta_input can either be a file path to a FASTA file or a FASTA object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta.

database can either be a file path to a FASTA file or a FASTA object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta. The Header texts of this file must follow the sintax-pattern, see make_sintax_db.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

If outfile is NULL a data.frame is returned. If it contains a file name (text) the data.frame is written to that file with tab-separated columns.

The data.frame contains the classification results for each input sequence. Both the Header and Sequence columns of fasta_input are copied into this table, and in addition are also the columns

for each rank. The ranks depend on the database file used, but are typically domain, phylum, class, order,family, genus and species. For each classification is also a bootstrap support score. These are in separate columns with corresponding names, i.e. domain_score, phylum_score, etc.

References

https://github.com/torognes/vsearch https://www.biorxiv.org/content/10.1101/074161v1

Examples

```
## Not run:
# Example files
db.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "sintax_db.fasta")
fasta.file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                     "small.fasta")

tax.tbl <- vs_sintax(fasta_input = fasta.file, database = db.file)
View(tax.tbl)

## End(Not run)
```

---

vs_uchime_denovo *Detect chimeras without external references (i.e. de novo)*

Description

vs_uchime_denovo detects chimeras present in the FASTA sequences in using VSEARCH's uchime_denovo algorithm. Automatically sorts sequences by decreasing abundance to enhance chimera detection accuracy.

Usage

```
vs_uchime_denovo(
  fasta_input,
  nonchimeras = NULL,
  chimeras = NULL,
  sizein = TRUE,
  sizeout = TRUE,
  relabel = NULL,
  relabel_sha1 = FALSE,
  fasta_width = 0,
  sample = NULL,
  log_file = NULL,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fasta_input	(Required). A FASTA file path or a FASTA object with reads. If a tibble is provided, any columns in addition to Header and Sequence will be preserved in the output. See *Details*.
nonchimeras	(Optional). Name of the FASTA output file for the non-chimeric sequences. If NULL (default), no output is written to file.
chimeras	(Optional). Name of the FASTA output file for the chimeric sequences. If NULL (default), no output is written to file.
sizein	(Optional). If TRUE (default), abundance annotations present in sequence headers are taken into account.
sizeout	(Optional). If TRUE (default), abundance annotations are added to FASTA headers.
relabel	(Optional). Relabel sequences using the given prefix and a ticker to construct new headers. Defaults to NULL.
relabel_sha1	(Optional). If TRUE (default), relabel sequences using the SHA1 message digest algorithm. Defaults to FALSE.
fasta_width	(Optional). Number of characters per line in the output FASTA file. Defaults to 0, which eliminates wrapping.
sample	(Optional). Add the given sample identifier string to sequence headers. For instance, if the given string is "ABC", the text ";sample=ABC" will be added to the header. If NULL (default), no identifier is added.
log_file	(Optional). Name of the log file to capture messages from VSEARCH. If NULL (default), no log file is created.
vsearch_options	(Optional). Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Chimeras in the input FASTA sequences are detected using VSEARCH´s uchime_denovo. In de novo mode, input FASTA file/object must present abundance annotations (i.e. a pattern [;]size=integer[;] in the header). Input order matters for chimera detection, so it is recommended to sort sequences by decreasing abundance.

fasta_input can either be a FASTA file or a FASTA object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta.

When providing a tibble as fasta_input, you can include additional columns with metadata (e.g., OTU IDs, sample origins). The function will preserve these columns by joining them back to the results based on the DNA sequence. This allows you to keep your metadata associated with your sequences throughout the chimera detection process.

If nonchimeras and chimeras are specified, resulting non-chimeric and chimeric sequences are written to these files in FASTA format.

If nonchimeras and chimeras are NULL, results are returned as a FASTA-objects.

nonchimeras and chimeras must either both be specified or both be NULL.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If nonchimeras and chimeras are specified, the resulting sequences after chimera detection written directly to the specified files in FASTA format, and no tibbles are returned.

If nonchimeras and chimeras are NULL, a FASTA object containing non-chimeric sequences is returned. This output tibble will include any additional columns that were present in the fasta_input tibble. An attribute named "chimeras" will contain a tibble of the chimeric sequences, also with the additional columns preserved.

Additionally, the returned tibble (when applicable) has an attribute "statistics" containing a tibble with chimera detection statistics.

The statistics tibble has the following columns:

- num_nucleotides: Total number of nucleotides used as input for chimera detection.
- num_sequences: Total number of sequences used as input for chimera detection.
- min_length_input_seq: Length of the shortest sequence used as input for chimera detection.
- max_length_input_seq: Length of the longest sequence used as input for chimera detection.
- avg_length_input_seq: Average length of the sequences used as input for chimera detection.
- num_non_chimeras: Number of non-chimeric sequences.
- num_chimeras: Number of chimeric sequences.
- input: Name of the input file/object for the chimera detection.

References

https://github.com/torognes/vsearch

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Define arguments
fasta_input <- file.path(file.path(path.package("Rsearch"), "extdata"),
                         "small_R1.fq")
nonchimeras <- "nonchimeras.fa"
chimeras <- "chimeras.fa"

# Detect chimeras with default parameters and return FASTA files
vs_uchime_denovo(fasta_input = fasta_input,
                 nonchimeras = nonchimeras,
                 chimeras = chimeras)
```

```
# Detect chimeras with default parameters and return a FASTA tibble
nonchimeras.tbl <- vs_uchime_denovo(fasta_input = fasta_input,
                                    nonchimeras = NULL,
                                    chimeras = NULL)

# Get chimeras tibble
chimeras.tbl <- attr(nonchimeras.tbl, "chimeras")

# Get statistics tibble
statistics.tbl <- attr(nonchimeras.tbl, "statistics")

## End(Not run)
```

---

vs_uchime_ref *Detect chimeras by comparing sequences to a reference database*

Description

vs_uchime_ref detects chimeras present in the FASTA sequences in using VSEARCH's uchime_ref algorithm.

Usage

```
vs_uchime_ref(
  fasta_input,
  database,
  nonchimeras = NULL,
  chimeras = NULL,
  sizein = TRUE,
  sizeout = TRUE,
  relabel = NULL,
  relabel_sha1 = FALSE,
  fasta_width = 0,
  sample = NULL,
  log_file = NULL,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fasta_input	(Required). A FASTA file path or a FASTA object with reads. See *Details*.
database	(Required). A FASTA file path or FASTA tibble object containing the reference sequences. These sequences are assumed to be chimera-free.

nonchimeras	(Optional). Name of the FASTA output file for the non-chimeric sequences. If NULL (default), no output is written to file.
chimeras	(Optional). Name of the FASTA output file for the chimeric sequences. If NULL (default), no output is written to file.
sizein	(Optional). If TRUE (default), abundance annotations present in sequence headers are taken into account.
sizeout	(Optional). If TRUE (default), abundance annotations are added to FASTA headers.
relabel	(Optional). Relabel sequences using the given prefix and a ticker to construct new headers. Defaults to NULL.
relabel_sha1	(Optional). If TRUE (default), relabel sequences using the SHA1 message digest algorithm. Defaults to FALSE.
fasta_width	(Optional). Number of characters per line in the output FASTA file. Defaults to 0, which eliminates wrapping.
sample	(Optional). Add the given sample identifier string to sequence headers. For instance, if the given string is "ABC", the text ";sample=ABC" will be added to the header. If NULL (default), no identifier is added.
log_file	(Optional). Name of the log file to capture messages from VSEARCH. If NULL (default), no log file is created.
threads	(Optional). Number of computational threads to be used by VSEARCH. Defaults to 1.
vsearch_options	(Optional). Additional arguments to pass to VSEARCH. Defaults to NULL. See *Details*.
tmpdir	(Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to NULL, which resolves to the session-specific temporary directory (tempdir()).

Details

Chimeras in the input FASTA sequences are detected using VSEARCH´s uchime_ref.

fasta_input can either be a FASTA file or a FASTA object. FASTA objects are tibbles that contain the columns Header and Sequence, see readFasta.

database must be a FASTA file or a FASTA object with high-quality non-chimeric sequences.

vsearch_options allows users to pass additional command-line arguments to VSEARCH that are not directly supported by this function. Refer to the VSEARCH manual for more details.

Value

A tibble or NULL.

If nonchimeras and chimeras are specified, the resulting sequences after chimera detection written directly to the specified files in FASTA format, and no tibbles are returned.

If nonchimeras and chimeras are NULL, A FASTA object containing non-chimeric sequences with an attribute "chimeras" containing a tibble of chimeric sequences is returned. If no chimeras are found, the "chimeras" attribute is an empty data frame.

Additionally, the returned tibble (when applicable) has an attribute "statistics" containing a tibble with chimera detection statistics.

The statistics tibble has the following columns:

- num_nucleotides: Total number of nucleotides used as input for chimera detection.
- num_sequences: Total number of sequences used as input for chimera detection.
- min_length_input_seq: Length of the shortest sequence used as input for chimera detection.
- max_length_input_seq: Length of the longest sequence used as input for chimera detection.
- avg_length_input_seq: Average length of the sequences used as input for chimera detection.
- num_non_chimeras: Number of non-chimeric sequences.
- num_chimeras: Number of chimeric sequences.
- input: Name of the input file/object for the chimera detection.

References

https://github.com/torognes/vsearch

https://github.com/torognes/vsearch

Examples

```
## Not run:
# Define arguments
query_file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                        "small.fasta")
db <- file.path(file.path(path.package("Rsearch"), "extdata"),
                "sintax_db.fasta")

# Detect chimeras with default parameters and return FASTA files
vs_uchime_ref(fasta_input = query_file,
              database = db,
              nonchimeras = "nonchimeras.fa",
              chimeras = "chimeras.fa")

# Detect chimeras with default parameters and return a FASTA tibble
nonchimeras.tbl <- vs_uchime_ref(fasta_input = query_file,
                                 database = db,
                                 nonchimeras = NULL,
                                 chimeras = NULL)

# Get chimeras tibble
chimeras.tbl <- attr(nonchimeras.tbl, "chimeras")

# Get statistics tibble
statistics.tbl <- attr(nonchimeras.tbl, "statistics")

## End(Not run)
```

vs_usearch_global *Global pairwise alignment*

Description

vs_usearch_global performs global pairwise alignment of query sequences against target sequences using VSEARCH.

Usage

```
vs_usearch_global(
  fastx_input,
  database,
  userout = NULL,
  otutabout = NULL,
  userfields = "query+target+id+alnlen+mism+opens+qlo+qhi+tlo+thi+evalue+bits",
  gapopen = "20I/2E",
  gapext = "2I/1E",
  id = 0.7,
  strand = "plus",
  maxaccepts = 1,
  maxrejects = 32,
  threads = 1,
  vsearch_options = NULL,
  tmpdir = NULL
)
```

Arguments

fastx_input	(Required). A FASTA/FASTQ file path or FASTA/FASTQ object. See *Details*.
database	(Required). A FASTA/FASTQ file path or FASTA/FASTQ tibble object containing the target sequences.
userout	(Optional). A character string specifying the name of the output file for the alignment results. If NULL (default), no output is written to a file and the results are returned as a tibble with the columns specified in userfields. See *Details*.
otutabout	(Optional). A character string specifying the name of the output file in an OTU table format. If NULL (default), no output is written to a file. If TRUE, the output is returned as a tibble. See *Details*.
userfields	(Optional). Fields to include in the output file. Defaults to "query+target+id+alnlen+mism+opens+qlo See *Details*.
gapopen	(Optional). Penalties for gap opening. Defaults to "20I/2E". See *Details*.
gapext	(Optional). Penalties for gap extension. Defaults to "2I/1E". See *Details*.
id	(Optional). Pairwise identity threshold. Defines the minimum identity required for matches. Defaults to 0.7.

`strand` (Optional). Specifies which strand to consider when comparing sequences. Can be either `"plus"` (default) or `"both"`.

`maxaccepts` (Optional). Maximum number of matching target sequences to accept before stopping the search for a given query. Defaults to `1`.

`maxrejects` (Optional). Maximum number of non-matching target sequences to consider before stopping the search for a given query. Defaults to 32. If `maxaccepts` and `maxrejects` are both set to 0, the complete database is searched.

`threads` (Optional). Number of computational threads to be used by `VSEARCH`. Defaults to `1`.

`vsearch_options` (Optional). Additional arguments to pass to `VSEARCH`. Defaults to `NULL`. See *Details*.

`tmpdir` (Optional). Path to the directory where temporary files should be written when tables are used as input or output. Defaults to `NULL`, which resolves to the session-specific temporary directory (`tempdir()`).

Details

Performs global pairwise alignment between query and target sequences using `VSEARCH`, and reports matches based on the specified pairwise identity threshold (`id`). Only alignments that meet or exceed the identity threshold are included in the output.

`fastx_input` and `database` can either be file paths to a FASTA/FASTQ files or FASTA/FASTQ objects. FASTA objects are tibbles that contain the columns `Header` and `Sequence`, see `readFasta`. FASTQ objects are tibbles that contain the columns `Header`, `Sequence`, and `Quality`, see `readFastq`.

`userfields` specifies the fields to include in the output file. Fields must be given as a character string separated by `"+"`. The default value of `userfields` equals `"query+target+id+alnlen+mism+opens+qlo+qhi+tlo+t` which gives a blast-like tab-separated format of twelve fields. See the 'Userfields' section in the `VSEARCH` manual for more information.

`otutabout` gives the option to output the results in an OTU table format with tab-separated columns. When writing to a file, the first line starts with the string "#OTU ID", followed by a tab-separated list of all sample identifiers (formatted as "sample=X"). Each subsequent line, corresponding to an OTU, begins with the OTU identifier and is followed by tab-separated abundances for that OTU in each sample. If `otutabout` is a character string, the output is written to the specified file. If `otutabout` is `TRUE`, the function returns the OTU table as a tibble, where the first column is named `otu_id` instead of "#OTU ID".

Pairwise identity (`id`) is calculated as the number of matching columns divided by the alignment length minus terminal gaps.

`vsearch_options` allows users to pass additional command-line arguments to `VSEARCH` that are not directly supported by this function. Refer to the `VSEARCH` manual for more details.

Visit the `VSEARCH` documentation for information about defining `gapopen` and `gapext`.

Value

A tibble or `NULL`.

If `userout` is specified the alignment results are written to the specified file, and no tibble is returned. If `userout` is `NULL` a tibble containing the alignment results with the fields specified by `userfields` is returned.

If `otutabout` is `TRUE`, an OTU table is returned as a tibble. If `otutabout` is a character string, the output is written to the file, and no tibble is returned.

References

https://github.com/torognes/vsearch

Examples

```
## Not run:
# You would typically use something else as database
query_file <- file.path(file.path(path.package("Rsearch"), "extdata"),
                        "small.fasta")
db <- query_file

# Run global pairwise alignment with default parameters and write results to file
vs_usearch_global(fastx_input = query_file,
                  database = db,
                  userout = "delete_me.txt")

# Read results, and give column names
result.tbl <- read.table("delete_me.txt",
                         sep = "\t",
                         header = FALSE,
                         col.names = c("query", "target", "id", "alnlen",
                                       "mism", "opens", "qlo", "qhi",
                                       "tlo", "thi", "evalue", "bits"))

## End(Not run)
```

Index