

Package: dbpq (via r-universe)

October 5, 2026

Title Manage FASTA Reference Databases for Taxonomic Assignment

Version 0.1.0.9000

Description Download, format, summarize, and modify FASTA reference databases used for taxonomic assignment in metabarcoding pipelines. Supports major databases ('UNITE', 'SILVA', 'PR2', 'Greengenes2', 'RDP', 'MIDORI2', 'BOLD', 'Diat.barcode', 'MaarjAM', 'Eukaryome') and converts between taxonomy header formats ('dada2', 'SINTAX', 'Greengenes2'). Part of the 'pqverse' ecosystem.

License AGPL-3

URL <https://github.com/adrientaudiere/dbpq>,
<https://adrientaudiere.github.io/dbpq/>

BugReports <https://github.com/adrientaudiere/dbpq/issues>

Depends R (>= 4.1.0)

Imports Biostrings, cli, methods, dplyr, jsonlite, lifecycle, purrr,
rlang, stringr, tibble, tidyr, tools, utils

Suggests ComplexUpset, dada2, ggplot2, ggVennDiagram, knitr,
MiscMetabar, phyloseq, rmarkdown, R.utils, testthat (>= 3.0.0),
visNetwork

Remotes krassowski/complex-upset

VignetteBuilder knitr

RdMacros lifecycle

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libicu-dev zlib1g-dev

Repository <https://adrientaudiere.r-universe.dev>

Date/Publication 2026-07-07 20:01:37 UTC

RemoteUrl <https://github.com/adrientaudiere/dbpq>

RemoteRef HEAD

RemoteSha 3483ddfea241e5dc25fcebcdfe550a0abebd988d

Contents

add_sh_to_taxonomy	3
count_pattern_db	5
count_seq_db	6
count_unwanted_tax	6
cutadapt_rm_primers_db	8
detect_tax_format	10
diagnose_db	11
download_bold_db	12
download_diatbarcode_db	14
download_eukaryome_db	15
download_greengenes2_db	16
download_ksgp_db	18
download_ltpplus_db	20
download_marjaam_db	22
download_midori2_db	24
download_pr2_db	25
download_rdp_db	27
download_silva_db	28
download_unite_db	30
filter_db	32
find_vsearch	33
format_fasta_db	34
format2dada2	35
format2dada2_species	37
format2sintax	38
get_file_extension	39
is_vsearch_installed	40
list_ranks_db	40
profile_db	41
search_taxa_db	43
summarize_db	45
tax_prefixes	46

Index

47

add_sh_to_taxonomy	<i>Annotate query sequences with UNITE Species Hypothesis (SH) names</i>
--------------------	--

Description

[Experimental]

Runs `vsearch --usearch_global` to match query sequences against a UNITE reference database, then extracts the Species Hypothesis (SH) identifier from each best hit. The SH name is the first `|`-delimited field in UNITE sequence identifiers (e.g. SH123456.09FU).

This function ports the logic of the `nf-core/ampliseq add_sh_to_taxonomy.py` script into R, using `vsearch` directly instead of requiring external lookup tables.

Usage

```
add_sh_to_taxonomy(
  query_fasta,
  unite_db,
  vsearchpath = find_vsearch(),
  id = 0.97,
  maxaccepts = 1,
  maxrejects = 32,
  nproc = 1,
  top_hits_only = TRUE,
  keep_temporary_files = FALSE,
  verbose = FALSE
)
```

Arguments

<code>query_fasta</code>	(Character) Path to a FASTA file containing the query sequences (e.g. ASVs or OTUs).
<code>unite_db</code>	(Character) Path to a UNITE reference FASTA file. The file can be in any format (SINTAX, UNITE default, etc.) as long as the sequence identifiers contain SH names as the first <code> </code> -delimited field. Files downloaded via download_unite_db() meet this requirement.
<code>vsearchpath</code>	(Character) Path to the <code>vsearch</code> executable. Defaults to find_vsearch() .
<code>id</code>	(Numeric, default 0.97) Minimum sequence identity threshold (0–1) for <code>vsearch --id</code> parameter.
<code>maxaccepts</code>	(Integer, default 1) Maximum number of hits to accept per query. Set to 0 for unlimited hits (useful when multiple equally-good matches may have different SH names).
<code>maxrejects</code>	(Integer, default 32) Maximum number of rejected hits before stopping the search for a query.
<code>nproc</code>	(Integer, default 1) Number of threads for <code>vsearch</code> .

top_hits_only (Logical, default TRUE) If TRUE, only report the top hit per query (highest identity). If FALSE, report all accepted hits, which is useful for detecting ambiguous SH assignments.

keep_temporary_files (Logical, default FALSE) If TRUE, do not delete the temporary blast6 output file after parsing.

verbose (Logical, default FALSE) Print vsearch progress messages.

Value

A data.frame with columns:

- **query**: Query sequence identifier.
- **sh_name**: Species Hypothesis name extracted from the best database hit (e.g. SH123456.09FU), or NA if no hit was found.
- **target**: Full identifier of the matched database sequence.
- **pct_id**: Percent identity of the match.
- **aln_len**: Alignment length.
- **mismatches**: Number of mismatches.
- **e_value**: E-value of the match.
- **is_ambiguous**: Logical; TRUE if multiple hits with the same top identity disagree on the SH name.

Author(s)

Adrien Taudière

See Also

[download_unite_db\(\)](#), [is_vsearch_installed\(\)](#), [find_vsearch\(\)](#)

Examples

```
## Not run:
# Download a UNITE database first
unite_file <- download_unite_db(
  version = "10.0",
  taxonomic_format = "sintax",
  taxon_group = "fungi"
)

# Annotate ASVs with SH names
sh_res <- add_sh_to_taxonomy(
  query_fasta = "asvs.fasta",
  unite_db = unite_file,
  id = 0.97
)
head(sh_res)
```

```
# Check for ambiguous assignments
sh_res[sh_res$is_ambiguous, ]

## End(Not run)
```

count_pattern_db	<i>Count lines matching a pattern in a FASTA file</i>
------------------	---

Description

Count lines (sequences if fasta file) matching a pattern. Accepts gzip files. May not work on Windows.

Usage

```
count_pattern_db(file, pattern = ">")
```

Arguments

file	(Character, required) Path to a file (plain or gzip), often a FASTA file.
pattern	(Character, default ">") A pattern to search for.

Value

An integer, the number of matching lines.

Author(s)

Adrien Taudière

See Also

[filter_db\(\)](#), [count_seq_db\(\)](#)

Examples

```
db <- system.file("extdata", "example_unite.fasta", package = "dbpq")
count_pattern_db(db, "Amanita")
```

count_seq_db	<i>Count sequences in a FASTA file</i>
--------------	--

Description

Counts the number of sequences in a FASTA file by counting header lines (lines starting with >). Accepts gzip files.

Usage

```
count_seq_db(file)
```

Arguments

file (Character, required) Path to a FASTA file (plain or gzip).

Value

An integer, the number of sequences.

Author(s)

Adrien Taudière

See Also

[count_pattern_db\(\)](#)

Examples

```
db <- system.file("extdata", "example_unite.fasta", package = "dbpq")
count_seq_db(db)
```

count_unwanted_tax	<i>Count unwanted values in a taxonomy table</i>
--------------------	--

Description

Scans a taxonomy table for common problematic values such as NA-like strings, placeholder labels ("unclassified", "unknown", etc.), and empty QIIME-style rank prefixes. The input can be a [phyloseq](#) object or a FASTA reference database file.

Returns a tibble summarising, for each pattern found, how many matches occur in each taxonomic rank.

Usage

```
count_unwanted_tax(
  x,
  patterns = unwanted_tax_patterns_default(),
  tax_format = "auto"
)
```

Arguments

x	Either: • a character string giving the path to a FASTA file (plain or gzip), or • a phyloseq object with a taxonomy table.
patterns	(Character vector) Regular expressions to search for. When MiscMetabar is installed, defaults to MiscMetabar::unwanted_tax_patterns ; otherwise falls back to a built-in copy of the same patterns. See Details .
tax_format	(Character) Taxonomy format of the FASTA file. One of "unite", "sintax", "greengenes2", "pr2", or "auto". Only used when x is a file path. If "auto" (default), the format is detected with detect_tax_format() . Ignored when x is a phyloseq object.

Details

The default patterns are:

```
"^[Nn][Aa][Nn]?$" NaN, nan, NA, na
"^[Nn]/[Aa]$" N/A, n/a
"^[Nn]one$" None, none
"^$" empty string
"^\s+$" whitespace only
"[Uu]nclassified" unclassified, Unclassified, xxx_unclassified
"[Uu]nknown" unknown, Unknown, xxx_unknown
"[Uu]nidentified" unidentified, Unidentified
"[Uu]ncultured" uncultured, Uncultured
"[Ii]ncertae[_\\s]?[Ss]edis" incertae_sedis, Incertae sedis
"^[Mm]etagenome$" metagenome, Metagenome
"^[Ee]nvironmental" environmental, Environmental
"^[kpcofgs]__$" empty QIIME-style rank prefixes
```

Value

A [tibble](#) with columns:

pattern The regular expression that matched.
 description A human-readable label for the pattern.

rank The taxonomic rank (column name) where matches were found.

n_matches Number of values matching the pattern in that rank.

example_values Up to 3 unique matching values (comma-separated).

Rows with zero matches are omitted.

Author(s)

Adrien Taudière

See Also

[summarize_db\(\)](#), [list_ranks_db\(\)](#)

Examples

```
# From a FASTA file (no unwanted values in example data)
db <- system.file("extdata", "example_unite.fasta", package = "dbpq")
count_unwanted_tax(db)

if(requireNamespace("MiscMetabar", quietly = TRUE)) {
  ref_fasta <- Biostrings::readDNASTringSet(system.file("extdata",
    "mini_UNITE_fungi.fasta.gz",
    package = "MiscMetabar", mustWork = TRUE
  ))
  data("data_fungi_mini", package = "MiscMetabar")
  physeq_new <- MiscMetabar::assign_mmseqs2(
    MiscMetabar::data_fungi_mini,
    ref_fasta = ref_fasta,
    behavior = "add_to_phyloseq"
  )
  count_unwanted_tax(physeq_new)
}
```

cutadapt_rm_primers_db

Remove primers from a FASTA database using cutadapt

Description

Removes pairs of primers and flanking regions from a FASTA reference database using **cutadapt**.
 Uses linked adapters to trim between forward and reverse primers.

Usage

```
cutadapt_rm_primers_db(
  ref_fasta,
  output = NULL,
  primer_fw = NULL,
  primer_rev = NULL,
  discard_untrimmed = TRUE,
  nproc = 1,
  verbose = TRUE,
  cmd_is_run = TRUE,
  return_file_path = FALSE,
  start_with_fw = FALSE,
  output_json = FALSE,
  error_tolerance = 0.1,
  args_before_cutadapt = paste0("source ~/miniforge3/etc/profile.d/conda.sh ",
    "&& conda activate cutadaptenv && ")
)
```

Arguments

ref_fasta	(Character, required) Path to a FASTA file (plain or gzip).
output	(Character) Path to the output FASTA file. If NULL, defaults to {basename}_cutadapted.fasta.
primer_fw	(Character, required) The forward primer DNA sequence.
primer_rev	(Character, required) The reverse primer DNA sequence.
discard_untrimmed	(Logical, default TRUE) Discard sequences where primers were not found.
nproc	(Integer, default 1) Number of CPU cores for cutadapt.
verbose	(Logical, default TRUE) Print summary statistics.
cmd_is_run	(Logical, default TRUE) If FALSE, return the command string without executing it.
return_file_path	(Logical, default FALSE) If TRUE, return the output file path instead of the command.
start_with_fw	(Logical, default FALSE) If TRUE, the forward primer must be anchored at the start of the sequence.
output_json	(Logical, default FALSE) If TRUE, write a JSON summary of the cutadapt process.
error_tolerance	(Numeric, default 0.1) Maximum error rate for primer matching.
args_before_cutadapt	(Character) Shell commands to run before cutadapt (e.g., conda activation).

Details

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

Value

The cutadapt command string, or the output file path if `return_file_path = TRUE`.

Author(s)

Adrien Taudière

Examples

```
## Not run:
cutadapt_rm_primers_db(
  "database.fasta.gz",
  output = "db_cutadapted.fasta",
  primer_fw = "GCATCGATGAAGAACGCAGC",
  primer_rev = "TCCTCCGCTTATTGATATGC"
)

## End(Not run)
```

detect_tax_format	<i>Detect taxonomy format from FASTA headers</i>
-------------------	--

Description

Reads a few sequence headers from a FASTA file and guesses the taxonomy format based on characteristic patterns.

Usage

```
detect_tax_format(file, n_headers = 20L)
```

Arguments

file	(Character, required) Path to a FASTA file (plain or gzip).
n_headers	(Integer, default 20) Number of headers to inspect.

Value

A character string: one of "unite", "sintax", "greengenes2", "pr2", or "unknown".

See Also

[tax_prefixes\(\)](#), [list_ranks_db\(\)](#), [summarize_db\(\)](#)

Examples

```
db <- system.file("extdata", "example_unite.fasta", package = "dbpq")
detect_tax_format(db)
```

diagnose_db

*Diagnose one or several FASTA reference databases***Description**

Runs format, integrity, and quality checks on one or more FASTA reference databases and returns a structured diagnosis: per-file statistics, per-rank taxonomic coverage, a table of collected warnings, a cross-file comparison (which flags problems such as a **mixed taxonomy format** across files), and (optionally) diagnostic plots.

The three axes checked are:

Format Is the file valid FASTA? Which taxonomy format is detected ([detect_tax_format\(\)](#))? Are the taxonomy prefixes consistent across all headers?

Integrity Can the file be read to completion (a truncated gzip fails here)? Are there empty sequences or duplicated sequence IDs?

Quality Sequence-length distribution and unusually short sequences, percentage of sequences annotated at each rank, ambiguous (non-ACGT) bases, duplicated sequences, and unwanted taxonomic values ([count_unwanted_tax\(\)](#)).

Usage

```
diagnose_db(
  files,
  tax_format = "auto",
  plot = TRUE,
  min_length = 200L,
  check_duplicates = TRUE,
  verbose = TRUE
)
```

Arguments

files	(Character vector, required) One or more paths to FASTA files (plain or gzip).
tax_format	(Character, default "auto") Taxonomy format to assume for every file. One of "auto", "unite", "sintax", "greengenes2", "pr2", or "dada2". When "auto" (default) the format is detected per file from its headers.
plot	(Logical, default TRUE) Whether to build diagnostic plots. Requires the ggplot2 package; when it is not installed, \$plots is NULL and a message is emitted.
min_length	(Integer, default 200) Sequences shorter than this are counted as "short" and raise a quality warning.
check_duplicates	(Logical, default TRUE) Whether to look for duplicated sequences. This compares full sequences and can be slow on very large databases; set to FALSE to skip.
verbose	(Logical, default TRUE) Whether to show a cli progress bar (with an ETA) while files are processed, then print a summary and the collected warnings.

Value

An object of class "dbpq_diagnosis": a list with components

stats A [tibble](#), one row per file, with sequence counts, length statistics, and problem counts.

coverage A long tibble of per-rank annotation coverage (file, rank, n_annotated, pct_annotated).

warnings A tibble of collected issues (file, check, severity, message); severity is one of "info", "warning", "error". Cross-file issues have file = NA.

cross_file A list describing agreement across files (formats, format_agreement, n_files).

plots A list of **ggplot2** objects (length, coverage), or NULL.

Author(s)

Adrien Taudière

See Also

[summarize_db\(\)](#), [detect_tax_format\(\)](#), [count_unwanted_tax\(\)](#)

Examples

```
unite <- system.file("extdata", "example_unite.fasta", package = "dbpq")
diag <- diagnose_db(unite)
diag$stats

# Several files at once: a mismatched taxonomy format is flagged
sintax <- system.file("extdata", "example_sintax.fasta", package = "dbpq")
diag2 <- diagnose_db(c(unite, sintax))
diag2$warnings
```

download_bold_db

Download sequences from BOLD Systems

Description

Downloads reference sequences from BOLD Systems (Barcode of Life Data) for a given taxonomic group. Unlike other databases, BOLD does not provide a single pre-built reference FASTA — sequences are queried by taxon via the BOLD API.

Usage

```
download_bold_db(
  dest_dir = ".",
  taxon = NULL,
  marker = "COI-5P",
  tax_format = c("dada2", "sintax", "none"),
  verbose = TRUE
)
```

Arguments

dest_dir	(Character, default ".") Directory to save the downloaded file.
taxon	(Character, required) Taxonomic name to query (e.g., "Fungi", "Arthropoda", "Mammalia").
marker	(Character, default "COI-5P") The barcode marker. Common values: "COI-5P", "ITS", "matK", "rbcL".
tax_format	(Character, default "dada2") Taxonomy format written into the FASTA headers, so the file can feed <code>MiscMetabar::add_new_taxonomy_pq()</code> . One of: "dada2": unprefixed, positional ranks (>Phylum;Class;Order;Family;Genus;Species;). "sintax": >processid;tax=p:Phylum,c:Class,... "none": the raw BOLD sequence FASTA with processid taxon marker headers (no ranked taxonomy).
verbose	(Logical, default TRUE) Print progress messages.

Details

This function uses the BOLD public API hosted at `v3.boldsystems.org`, which remains available after the main BOLD site's migration to v5. With `tax_format = "none"` it queries the sequence endpoint (FASTA). With `"dada2"/"sintax"` it queries the combined endpoint (TSV with the full ranked taxonomy), keeps the requested marker, and writes a taxonomy-headed FASTA (gaps removed). BOLD's taxonomy starts at phylum, so the dada2 output has no kingdom level.

For very large taxa the download may be slow or hit server limits; use narrower queries, or the **BOLDconnectR** package for the full v5 (BCDM) data model.

Please cite: Ratnasingham S & Hebert PDN (2007) BOLD: The Barcode of Life Data System. *Molecular Ecology Notes* 7:355-364. doi:10.1111/j.14718286.2007.01678.x

Value

The path to the downloaded file (invisibly).

Author(s)

Adrien Taudière

Examples

```
## Not run:
# COI reference for a genus, dada2 taxonomy headers
download_bold_db(dest_dir = "databases", taxon = "Danaus")

# ITS sequences for an order, SINTAX headers
download_bold_db(
  dest_dir = "databases",
  taxon = "Agaricales",
  marker = "ITS",
  tax_format = "sintax"
)
```

```
## End(Not run)
```

```
download_diatbarcode_db
```

Download a Diat.barcode reference database

Description

Downloads the Diat.barcode reference database for diatom rbcL barcoding. This is a curated database for diatom identification using the rbcL marker gene. The recommended access method is through the **diatbarcode** R package, which provides additional tools for working with the database.

Usage

```
download_diatbarcode_db(
  dest_dir = ".",
  format = c("dada2", "dada2_species"),
  url = NULL,
  verbose = TRUE
)
```

Arguments

<code>dest_dir</code>	(Character, default ".") Directory to save the downloaded file.
<code>format</code>	(Character, default "dada2") One of "dada2" or "dada2_species". Uses dada2-formatted files from the INRAE Dataverse.
<code>url</code>	(Character) Direct download URL. If NULL (default), the function provides instructions for using the diatbarcode R package.
<code>verbose</code>	(Logical, default TRUE) Print progress messages.

Details

The Diat.barcode database is maintained by INRAE and hosted on Recherche Data Gouv. For more advanced access (metadata, full taxonomy tables, custom queries), consider using the **diatbarcode** R package directly:

```
diatbarcode::download_diatbarcode(
  path = "databases",
  flavor = "rbc1312_dada2_tax"
)
```

Please cite: Rimet F et al. (2019) Diat.barcode, an open-access curated barcode library for diatoms. Scientific Reports 9:15116. doi:[10.1038/s41598019515006](https://doi.org/10.1038/s41598019515006)

Value

The path to the downloaded file (invisibly), or a message with download instructions if no URL is provided.

Author(s)

Adrien Taudière

Examples

```
## Not run:
download_diatbarcode_db(dest_dir = "databases")

## End(Not run)
```

download_eukaryome_db *Download the Eukaryome reference database*

Description

Downloads the Eukaryome database for eukaryotic organisms. Supports multiple markers (SSU 18S, ITS, LSU 28S) and output formats (general FASTA, dada2, mothur).

Usage

```
download_eukaryome_db(dest_dir = ".", url = NULL, verbose = TRUE)
```

Arguments

dest_dir	(Character, default ".") Directory to save the downloaded file.
url	(Character) Direct download URL for the Eukaryome file. Override if the URL has changed. If NULL (default), the function directs you to the Eukaryome download page.
verbose	(Logical, default TRUE) Print progress messages.

Details

Eukaryome does not provide a stable programmatic download API. Files are available at <https://eukaryome.org/download/> in several formats:

- General FASTA: <https://eukaryome.org/generalfasta/>
- dada2 format: <https://eukaryome.org/dada2/>
- mothur format: <https://eukaryome.org/mothur/>
- SINTAX format: <https://eukaryome.org/syntax/>
- QIIME2 format: <https://eukaryome.org/qiime2/>

Visit one of these pages, copy the direct download link, and pass it via the url parameter.

Please cite: Vasar M et al. (2024) Eukaryome: the rRNA gene reference database for identification of all eukaryotes. Database. [doi:10.1093/database/baee043](https://doi.org/10.1093/database/baee043)

Value

The path to the downloaded file (invisibly), or a message with download instructions if no URL is provided.

Author(s)

Adrien Taudière

Examples

```
## Not run:
# Download with a specific URL from the Eukaryome website
download_eukaryome_db(
  dest_dir = "databases",
  url = "https://eukaryome.org/files/eukaryome_v1.9_SSU_dada2.fasta.gz"
)

## End(Not run)
```

download_greengenes2_db

Download a Greengenes2 reference database

Description

Downloads the Greengenes2 16S rRNA database. By default, downloads the dada2-formatted training sets from Zenodo (maintained by Benjamin Callahan). Alternatively, downloads backbone sequences from the Greengenes2 FTP server.

Note that Greengenes2 uses d__ (domain) instead of k__ (kingdom) as the first rank prefix. Use `tax_format = "greengenes2"` with `summarize_db()` and `list_ranks_db()` for correct parsing.

Usage

```
download_greengenes2_db(
  dest_dir = ".",
  version = "2024.09",
  format = c("dada2", "dada2_species", "fasta"),
  tax_format = c("dada2", "sintax", "keep"),
  verbose = TRUE
)
```

Arguments

<code>dest_dir</code>	(Character, default ".") Directory to save the downloaded file.
<code>version</code>	(Character, default "2024.09") Greengenes2 version in YYYY.MM format.
<code>format</code>	(Character, default "dada2") One of:

	• "dada2": dada2-formatted training set from Zenodo (recommended for <code>dada2::assignTaxonomy()</code>). • "dada2_species": species-level training set for <code>dada2::assignTaxonomy()</code> (includes species). • "fasta": plain FASTA sequences from the FTP server.
tax_format	(Character, default "dada2") How to write taxonomy in the headers of the "dada2"/"dada2_species" training set. The Greengenes2 trainset ships with d__/p__ rank prefixes, which <code>dada2::assignTaxonomy()</code> and <code>MiscMetabar::add_new_taxonomy_pq()</code> reject. One of: • "dada2": strip the prefixes to unprefix, positional <code>dada2(>Bacteria;Pseudomonadota;...)</code>. • "sintax": rewrite as <code>>ID;tax=d:Bacteria,p:... </code> • "keep": leave the original d__-prefixed headers untouched. Ignored for <code>format = "fasta"</code>.
verbose	(Logical, default TRUE) Print progress messages.

Details

The dada2-formatted files are maintained by Benjamin Callahan on Zenodo and are the same source as the SILVA dada2 training sets. See <https://benjjneb.github.io/dada2/training.html> for details.

The Greengenes2 trainset uses d__/p__ rank prefixes. By default (`tax_format = "dada2"`) the prefixes are stripped so the file is directly usable by `dada2::assignTaxonomy()` and `add_new_taxonomy_pq()`.

Please cite: McDonald D et al. (2024) Greengenes2 unifies microbial data in a single reference tree. Nature Biotechnology 42:715-718. [doi:10.1038/s41587023018451](https://doi.org/10.1038/s41587023018451)

Value

The path to the downloaded file (invisibly).

Author(s)

Adrien Taudière

See Also

[format2dada2\(\)](#), [tax_prefixes\(\)](#)

Examples

```
## Not run:
# Download dada2-formatted Greengenes2
download_greengenes2_db(dest_dir = "databases")

# Download plain FASTA from FTP
download_greengenes2_db(dest_dir = "databases", format = "fasta")

## End(Not run)
```

download_ksgp_db

*Download the KSGP or GTDB+ reference database***Description**

Downloads the KSGP (Karst, Silva, GTDB, and PR2) reference database for SSU rRNA taxonomic assignment, particularly optimized for Archaea communities. KSGP combines near full-length rRNA sequences from Karst et al. 2018, re-annotated SILVA prokaryote SSU sequences, cleaned GTDB 16S sequences, PR2 eukaryote 18S sequences, and MIDORI2 mitochondrial sequences. Taxonomy is based on GTDB, providing phylogenetically consistent classification.

Also provides access to the GTDB+ and GTDB_cleaned databases built as intermediate steps during KSGP construction.

Three annotation variants are available for database = "KSGP" and file_type = "tax":

- "sintax" (default): SINTAX-based taxonomic assignments (KSGP Sintax in the paper). Available for all versions.
- "lca": Conservative lowest common ancestor assignments (KSGP LCA). Only available for version "3.1".
- "ksgp_plus": Similarity-clustered putative taxa (KSGP+). Only available for version "3.1".

Usage

```
download_ksgp_db(
  dest_dir = ".",
  database = c("KSGP", "GTDB_plus", "GTDB_cleaned"),
  file_type = c("fasta", "tax", "archive"),
  annotation = c("lca", "sintax", "ksgp_plus"),
  tax_format = c("dada2", "sintax", "none"),
  version = "3.1",
  verbose = TRUE,
  timeout = Inf
)
```

Arguments

- | | |
|-----------|---|
| dest_dir | (Character, default ".") Directory to save the downloaded file. |
| database | (Character, default "KSGP") One of: <ul style="list-style-type: none"> • "KSGP": Full KSGP SSU database (Archaea + Bacteria + Eukaryota). • "GTDB_plus": Cleaned GTDB 16S sequences with PR2 and MIDORI2. • "GTDB_cleaned": Cleaned GTDB 16S sequences only (no eukaryote supplement). |
| file_type | (Character, default "fasta") One of: <ul style="list-style-type: none"> • "fasta": FASTA file with SSU sequences. • "tax": Taxonomy file (.tax) with taxonomic annotations. |

	• "archive": Complete .tar.gz archive (all KSGP files, all annotation variants). Only available for database = "KSGP".
annotation	(Character, default "lca") Taxonomic annotation method. One of "lca", "sintax", or "ksgp_plus". Used to pick the matching .tax file when file_type = "tax", and to pick the taxonomy merged into the FASTA headers when file_type = "fasta" and tax_format != "none". The lca annotation has the broadest sequence coverage and is the default for a fully-annotated KSGP FASTA. Only "sintax" is available for version "1.0".
tax_format	(Character, default "dada2") When file_type = "fasta", also download the companion .tax file and merge its taxonomy into the FASTA headers (matched by sequence ID), so the file feeds MiscMetabar::add_new_taxonomy_pq(). One of "dada2", "sintax", or "none" (keep accession-only headers). Sequences whose ID is absent from the .tax (e.g. the SILVA-derived portion) keep accession-only headers. Ignored for file_type = "tax" / "archive".
version	(Character, default "3.1") KSGP version. Known versions: "3.1" (2025, recommended) and "1.0".
verbose	(Logical, default TRUE) Print progress messages.
timeout	(Numeric, default Inf) Timeout in seconds for each HTTP request. The default disables R's 60-second timeout so the multi-hundred-MB to multi-GB downloads (KSGP FASTA, the v3.1 archive) can complete. Set to a positive number of seconds to restore a strict timeout.

Details

When file_type = "fasta", the function downloads the matching KSGP_v<version>.tar.gz archive (one HTTP request) and extracts the FASTA — and, when tax_format != "none", the chosen .tax file — to dest_dir, then removes the archive. The archive is roughly 3.5x smaller than the raw FASTA (e.g. ~686 MB vs ~2.4 GB for v3.1), so this is both faster and lighter on the server than two separate requests. The KSGP FASTA and taxonomy files are otherwise separate downloads.

With tax_format = "sintax" (or "dada2"), the taxonomy is merged into the FASTA headers (one sequence ID per row, matched against the .tax file) and the .tax file is removed, so the result is a single FASTA ready for VSEARCH/dada2 — the original prefix letters from the .tax are preserved in the SINTAX output (a KSGP line starting with k__Bacteria; becomes >ID;tax=k:Bacteria,..., not d:Bacteria,...). To use KSGP for taxonomic assignment:

- With VSEARCH SINTAX: download the FASTA (file_type = "fasta", tax_format = "sintax").
- With dada2: download the FASTA (file_type = "fasta", tax_format = "dada2").
- With LotuS2: the KSGP database is integrated directly.
- For a complete set of all files: use file_type = "archive".

KSGP substantially improves Archaea annotation over SILVA and Greengenes2: Class and Order assignments increase by 2.7x and 4.2x respectively.

Please cite: Grant A et al. (2025) KSGP 3.1: improved taxonomic annotation of Archaea communities using LotuS2, the genome taxonomy database and RNAseq data. ISME Communications 5(1): ycaf094. doi:10.1093/ismeco/ycaf094

Value

The path to the downloaded file (invisibly).

Author(s)

Adrien Taudière

See Also

`download_silva_db()`, `download_pr2_db()`, `format2sintax()`

Examples

```
## Not run:
# Download KSGP v3.1 FASTA
download_ksgp_db(dest_dir = "databases")

# Download KSGP v3.1 LCA taxonomy file
download_ksgp_db(
  dest_dir = "databases",
  file_type = "tax",
  annotation = "lca"
)

# Download KSGP+ taxonomy file
download_ksgp_db(
  dest_dir = "databases",
  file_type = "tax",
  annotation = "ksgp_plus"
)

# Download the complete KSGP archive (all annotation variants)
download_ksgp_db(dest_dir = "databases", file_type = "archive")

# Download GTDB+ (cleaned GTDB + PR2 + MIDORI2)
download_ksgp_db(dest_dir = "databases", database = "GTDB_plus")

## End(Not run)
```

download_ltplus_db

Download the LTPlus reference database

Description

Downloads the LTPlus 16S rRNA gene reference FASTA for Bacteria and Archaea. LTPlus extends the All-Species Living Tree Project (LTP) type-strain collection with the best-quality non-type sequences selected from the SILVA non-redundant and GTDB databases, plus the highest-quality 16S sequences deposited at NCBI between 2019 and 2025. Sequences are clustered non-redundantly at a 98.7% identity threshold, yielding a compact database that covers most of the known prokaryotic genealogical diversity.

Usage

```
download_ltplus_db(
  dest_dir = ".",
  url = "https://biocom.uib.es/opucheck-backend/api/releases/02_26_06",
  tax_format = c("dada2", "sintax", "none"),
  csv_url = "https://biocom.uib.es/opucheck-backend/api/releases/02_26_02",
  to_dna = TRUE,
  verbose = TRUE
)
```

Arguments

<code>dest_dir</code>	(Character, default ".") Directory to save the downloaded FASTA file.
<code>url</code>	(Character) Direct download URL for the LTPlus FASTA. Defaults to the February 2026 release served by the Marine Microbiology Group (IMEDEA, UIB-CSIC). Pass a different release URL to download another version (see Details).
<code>tax_format</code>	(Character, default "dada2") Taxonomy format to write into the FASTA headers, so the file can feed <code>MiscMetabar::add_new_taxonomy_pq()</code> . One of: "dada2": unprefixed, semicolon-delimited ranks (>Bacteria;Pseudomonadota;...). "sintax": >ID;tax=d:Bacteria,p:Pseudomonadota,... "none": keep the original accession-only headers. Taxonomy is read from the companion CSV (see <code>csv_url</code>).
<code>csv_url</code>	(Character) URL of the LTPlus metadata CSV that maps each sequence accession to its full taxonomy. Defaults to the CSV of the February 2026 release. Only used when <code>tax_format != "none"</code> .
<code>to_dna</code>	(Logical, default TRUE) Convert the downloaded RNA FASTA to DNA (transcribe U to T) and rewrite it as a standard FASTA. Set to FALSE to keep the original RNA file unchanged. Requires the Biostrings package.
<code>verbose</code>	(Logical, default TRUE) Print progress messages.

Details

The file is the LTPlus 16S FASTA exported from the underlying curated alignment with gap columns removed ("compressed"), so the sequences are **unaligned** and vary in length (~140 MB total). It is served directly (no registration) from the LTP release backend; the file name is taken from the server's Content-Disposition header when available (e.g. `ltplus_10_02_2026_compressed.fasta`).

The released sequences are in the **RNA alphabet** (U rather than T) and the sequence lines contain whitespace. With `to_dna = TRUE` (default) the function transcribes U to T and rewrites a clean, whitespace-free DNA FASTA in place, ready for DNA-based classifiers such as `dada2` or `VSEARCH`. With `to_dna = FALSE` the original RNA file is kept as-is.

The released FASTA headers carry only an accession (e.g. >LAJZ3046); the taxonomy lives in a companion CSV. With `tax_format = "dada2"` (default) or `"sintax"` the function downloads that CSV, maps each accession to its full LTPlus lineage, and rewrites the headers with taxonomy so the file is ready for `MiscMetabar::add_new_taxonomy_pq()`. Use `tax_format = "none"` to keep the accession-only headers.

The default url points to the current release file. To list available releases and files, see the Downloads section of <https://bioinfo.uib.es/ltp/> or query <https://biocom.uib.es/opucheck-backend/api/releases>; each file has an id appended to `.../api/releases/` to form its download URL. The ARB database, CSV and Newick tree files are also available there.

Please cite: Rosselló-Móra R et al. (2026) A pipeline for improved 16S rRNA gene-based phylogeny and diversity analyses of Bacteria and Archaea. Research Square. [doi:10.21203/rs.3.rs-9370187/v1](https://doi.org/10.21203/rs.3.rs-9370187/v1)

Value

The path to the downloaded file (invisibly).

Author(s)

Adrien Taudière

See Also

[download_silva_db\(\)](#), [download_greengenes2_db\(\)](#), [download_ksgp_db\(\)](#)

Examples

```
## Not run:
# Download the current LTPlus 16S FASTA (DNA, dada2 taxonomy headers)
download_ltplus_db(dest_dir = "databases")

# SINTAX-formatted headers instead
download_ltplus_db(dest_dir = "databases", tax_format = "sintax")

# Keep the original RNA, accession-only FASTA without conversion
download_ltplus_db(dest_dir = "databases", to_dna = FALSE, tax_format = "none")

## End(Not run)
```

download_marjaam_db	<i>Download the MaarjAM reference database</i>
---------------------	--

Description

Downloads the MaarjAM database of arbuscular mycorrhizal fungi (AMF) virtual taxa (VT) sequences, maintained at the University of Tartu. The QIIME-formatted release (a zip bundling a FASTA and a taxonomy table) is used so that the resulting FASTA carries taxonomy in its headers, ready for `MiscMetabar::add_new_taxonomy_pq()`.

Usage

```
download_marjaam_db(
  dest_dir = ".",
  dataset = c("SSU", "SSU_TYPE", "LSU", "full_ITS", "onlyITS"),
  tax_format = c("dada2", "sintax", "none"),
  url = NULL,
  verbose = TRUE
)
```

Arguments

dest_dir	(Character, default ".") Directory to save the downloaded file.
dataset	(Character, default "SSU") Which MaarjAM marker release to download. One of "SSU", "SSU_TYPE", "LSU", "full_ITS", "onlyITS".
tax_format	(Character, default "dada2") Taxonomy format written into the FASTA headers. One of "dada2", "sintax", or "none" (keep the QIIME accession_VTX... headers without taxonomy).
url	(Character) Direct download URL for the MaarjAM QIIME zip. If NULL (default), it is built from dataset. Override if the URL has changed.
verbose	(Logical, default TRUE) Print progress messages.

Details

The download links are listed at <https://maarjam.ut.ee/?action=bDownload>. The QIIME zip contains *.qiime.fasta and *.qiime.txt (a tab-separated id k__Fungi;p__...;s__VTX... table); taxonomy is merged into the FASTA headers by matching sequence IDs.

Please cite: Opik M et al. (2010) The online database MaarjAM reveals global and ecosystemic distribution patterns in arbuscular mycorrhizal fungi (Glomeromycota). New Phytologist 188:223-241. doi:[10.1111/j.14698137.2010.03334.x](https://doi.org/10.1111/j.14698137.2010.03334.x)

Value

The path to the downloaded FASTA file (invisibly).

Author(s)

Adrien Taudière

Examples

```
## Not run:
# SSU (18S) AMF database with dada2 taxonomy headers
download_marjaam_db(dest_dir = "databases")

# SINTAX-formatted headers
download_marjaam_db(dest_dir = "databases", tax_format = "sintax")

## End(Not run)
```

download_midori2_db *Download a MIDORI2 reference database*

Description

Downloads the MIDORI2 reference database for eukaryotic mitochondrial genes (COI, 12S, 16S, Cytb, etc.). MIDORI2 provides pre-formatted FASTA files for multiple classifiers (dada2, SINTAX, RDP, BLAST).

Usage

```
download_midori2_db(
  dest_dir = ".",
  gene = "CO1",
  format = c("dada2", "dada2_species", "SINTAX", "RDP", "BLAST"),
  seq_type = c("UNIQ", "LONGEST"),
  url = NULL,
  verbose = TRUE
)
```

Arguments

dest_dir	(Character, default ".") Directory to save the downloaded file.
gene	(Character, default "CO1") Mitochondrial gene marker. Common values: "CO1", "srRNA" (12S), "1rRNA" (16S), "Cytb".
format	(Character, default "dada2") One of "dada2", "dada2_species", "SINTAX", "RDP", or "BLAST".
seq_type	(Character, default "UNIQ") One of "UNIQ" (all unique haplotypes per species) or "LONGEST" (single longest sequence per species).
url	(Character) Direct download URL. If NULL (default), the function provides instructions and the download page URL.
verbose	(Logical, default TRUE) Print progress messages.

Details

MIDORI2 download URLs include a date-stamped directory path that changes with each GenBank release, making fully programmatic access fragile. Visit <https://www.reference-midori.info/download.php> to find the current download URL for your desired gene and format, then pass it via the url parameter.

Files are typically named following this pattern: MIDORI2_{TYPE}_NUC_SP_GB{VERSION}_{GENE}_{FORMAT}.fasta.gz

Please cite: Leray M et al. (2022) MIDORI2: A collection of quality controlled, preformatted, and regularly updated reference databases for taxonomic assignment of eukaryotic mitochondrial sequences. Environmental DNA 4:894-907. doi:10.1002/edn3.303

Value

The path to the downloaded file (invisibly), or a message with download instructions if no URL is provided.

Author(s)

Adrien Taudière

See Also

`format2sintax()`, `format2dada2()`

Examples

```
## Not run:
# Get instructions for downloading MIDORI2
download_midori2_db()

# Download with a specific URL
download_midori2_db(
  dest_dir = "databases",
  url = "https://reference-midori.info/download/Databases/..."
)

## End(Not run)
```

download_pr2_db

Download a PR2 reference database

Description

Downloads the PR2 protist ribosomal reference database from GitHub releases. PR2 provides 18S rRNA gene sequences for protists and other eukaryotes.

For more advanced access to PR2 data (e.g., full taxonomy tables, metadata, or custom queries), see the [pr2database](#) R package.

Usage

```
download_pr2_db(
  dest_dir = ".",
  version = NULL,
  format = c("dada2", "mothur", "UTAX", "sintax"),
  marker = c("SSU", "plastid"),
  verbose = TRUE
)
```

Arguments

dest_dir	(Character, default ".") Directory to save the downloaded file.
version	(Character) PR2 version number (e.g., "5.0.0"). If NULL (default), the latest release is fetched from GitHub.
format	(Character, default "dada2") One of "dada2", "mothur", "UTAX", or "sintax" (alias for "UTAX"). See Taxonomic ranks below: the "dada2" file keeps PR2's 9 ranks, whereas the "UTAX"/"sintax" file collapses them to 8 (Division and Subdivision are merged).
marker	(Character, default "SSU") One of "SSU" or "plastid".
verbose	(Logical, default TRUE) Print progress messages.

Details

PR2 releases are hosted on GitHub at <https://github.com/pr2database/pr2database/releases>.

Taxonomic ranks:

PR2 uses **9** taxonomic ranks. The "dada2" file keeps all nine as a positional, semicolon-delimited lineage (no rank prefixes); pass them to `dada2::assignTaxonomy()` (via `MiscMetabar::add_new_taxonomy_pq()` with method = "dada2") through taxLevels:

```
c("Domain", "Supergroup", "Division", "Subdivision", "Class", "Order", "Family", "Genus", "Species")
```

The "UTAX"/"sintax" file targets VSEARCH/USEARCH SINTAX and uses the **8** standard single-letter rank prefixes (k, d, p, c, o, f, g, s). To fit PR2's nine ranks onto them, PR2 **merges**

Division and Subdivision into the p: rank (joined by -). The 9 → 8 mapping is:

PR2 rank (dada2)	SINTAX prefix (UTAX)
Domain	k:
Supergroup	d:
Division + Subdivision	p: (e.g. Alveolata-Dinoflagellata)
Class	c:
Order	o:
Family	f:
Genus	g:
Species	s:

Mind the per-method argument when calling `MiscMetabar::add_new_taxonomy_pq()`:

- method = "dada2" (the "dada2" download) keeps all 9 ranks — pass the 9 names above as taxLevels (forwarded to `dada2::assignTaxonomy()`).
- method = "sintax" (the "sintax"/"UTAX" download) has 8 ranks — pass 8 names as taxa_ranks, e.g. c("Domain", "Supergroup", "Division-Subdivision", "Class", "Order", "Family", "Genus", "Species"). The dada2 taxLevels argument is **ignored** by the SINTAX path, so the default 7 ranks would be used and parsing the 8-rank output fails.

Please cite: Guillou L et al. (2013) The Protist Ribosomal Reference database (PR2). Nucleic Acids Research 41:D1108-D1113. doi:10.1093/nar/gks1160

Value

The path to the downloaded file (invisibly).

Author(s)

Adrien Taudière

See Also

`format2dada2()`, `format2sintax()`, `pr2database` R package

Examples

```
## Not run:
# Download latest PR2 in dada2 format
download_pr2_db(dest_dir = "databases")

# Download specific version in UTAX format
download_pr2_db(dest_dir = "databases", version = "5.0.0", format = "UTAX")

## End(Not run)
```

download_rdp_db

Download an RDP reference database

Description

Downloads the Ribosomal Database Project (RDP) 16S rRNA database. By default, downloads the dada2-formatted training sets from Zenodo (maintained by Benjamin Callahan).

Usage

```
download_rdp_db(
  dest_dir = ".",
  trainset = "19",
  format = c("dada2", "dada2_species"),
  verbose = TRUE
)
```

Arguments

<code>dest_dir</code>	(Character, default ".") Directory to save the downloaded file.
<code>trainset</code>	(Character, default "19") RDP trainset version number.
<code>format</code>	(Character, default "dada2") One of: "dada2": training set for <code>dada2::assignTaxonomy()</code>. "dada2_species": species assignment file for <code>dada2::addSpecies()</code>.
<code>verbose</code>	(Logical, default TRUE) Print progress messages.

Details

The dada2-formatted files are maintained by Benjamin Callahan on Zenodo. See <https://benjjneb.github.io/dada2/training.html> for details.

Please cite: Cole JR et al. (2014) Ribosomal Database Project: data and tools for high throughput rRNA analysis. Nucleic Acids Research 42:D633-D642. doi:[10.1093/nar/gkt1244](https://doi.org/10.1093/nar/gkt1244)

Value

The path to the downloaded file (invisibly).

Author(s)

Adrien Taudière

See Also

[format2dada2\(\)](#), [download_silva_db\(\)](#)

Examples

```
## Not run:
# Download RDP trainset 19 for assignTaxonomy()
download_rdp_db(dest_dir = "databases")

# Download species assignment file
download_rdp_db(dest_dir = "databases", format = "dada2_species")

## End(Not run)
```

download_silva_db

Download a SILVA reference database NR99 version

Description

Downloads the SILVA ribosomal RNA database (16S/18S/23S/28S). By default, downloads the dada2-formatted training sets from the official arb-silva.de DADA2 release (available for both SSU and LSU). Can also produce a SINTAX-formatted database (converted locally from the dada2 trainset) or download the raw SILVA NR99 export.

For the PARC version (all sequences, not clustered like NR99), see `dada2::makeSpeciesFasta_Silva()` on the manually downloaded FASTA file.

Usage

```
download_silva_db(
  dest_dir = ".",
  version = "138.2",
  target = c("SSU", "LSU"),
  format = c("dada2", "dada2_species", "sintax", "raw"),
  verbose = TRUE
)
```

Arguments

dest_dir	(Character, default ".") Directory to save the downloaded file.
version	(Character, default "138.2") SILVA version number. Only the current SILVA release is hosted at the arb-silva DADA2 path used by the dada2/dada2_species/sintax formats.
target	(Character, default "SSU") One of "SSU" or "LSU".
format	(Character, default "dada2") One of: • "dada2": dada2-formatted toSpecies training set (NR99, recommended for dada2::assignTaxonomy()). • "dada2_species": species assignment file for dada2::addSpecies(). • "sintax": VSEARCH/USEARCH SINTAX database, converted locally from the dada2 toSpecies trainset (7 ranks d,p,c,o,f,g,s). Sequence labels are synthetic (SILVA<version>_<target>_NNNNNN) because the dada2 trainset carries no accession. Written as a separate *_sintax.fasta.gz file. • "raw": raw SILVA NR99 FASTA with taxonomy from arb-silva.de.
verbose	(Logical, default TRUE) Print progress messages.

Details

The dada2-formatted files are provided by arb-silva.de and are the recommended format for `dada2::assignTaxonomy()` and `dada2::addSpecies()`. See <https://benjjneb.github.io/dada2/training.html> for details.

SILVA data is free for academic use. Commercial use requires a license. See <https://www.arb-silva.de/silva-license-information/>.

Please cite: Quast C et al. (2013) The SILVA ribosomal RNA gene database project. Nucleic Acids Research 41:D590-D596. doi:10.1093/nar/gks1219

Value

The path to the resulting file (invisibly). For "sintax" this is the converted *_sintax.fasta.gz; the intermediate trainset is also kept in dest_dir.

Author(s)

Adrien Taudière

See Also

[format2dada2\(\)](#), [format2sintax\(\)](#)

Examples

```
## Not run:
# Download dada2-formatted SILVA for assignTaxonomy()
download_silva_db(dest_dir = "databases")

# Download species assignment file
download_silva_db(dest_dir = "databases", format = "dada2_species")

# Download a SINTAX database for SSU (converted from the dada2 trainset)
download_silva_db(dest_dir = "databases", format = "sintax")

# SINTAX database for LSU
download_silva_db(dest_dir = "databases", target = "LSU", format = "sintax")

# Download raw SILVA NR99 FASTA
download_silva_db(dest_dir = "databases", format = "raw")

## End(Not run)
```

download_unite_db	<i>Download a UNITE reference database</i>
-------------------	--

Description

Downloads the UNITE fungal ITS database for taxonomic assignment from the UNITE DOI repository (PlutoF/Zenodo). Provides both "dynamic" (including singletons) and "static" versions, for fungi or all eukaryotes.

UNITE general FASTA releases use the k__/p__ taxonomy format and are compatible with dada2, VSEARCH, and other classifiers after reformatting with [format2dada2\(\)](#) or [format2sintax\(\)](#).

Usage

```
download_unite_db(
  dest_dir = ".",
  type = c("dynamic", "static"),
  taxon_group = c("fungi", "eukaryotes"),
  version = "10.0",
  taxonomic_format = c("default", "sintax"),
  doi = NULL,
  verbose = TRUE
)
```

Arguments

dest_dir	(Character, default ".") Directory to save the downloaded file.
type	(Character, default "dynamic") One of "dynamic" or "static". Dynamic files include singletons. Only used when taxonomic_format = "default". Note: as of UNITE v10.0, separate static/dynamic archives are not available for taxon_group = "fungi"; both options download the same archive.
taxon_group	(Character, default "fungi") One of "fungi" or "eukaryotes".
version	(Character, default "10.0") UNITE version. Use "10.0" for the 2024 release.
taxonomic_format	(Character, default "default") One of "default" or "sintax". When "default", downloads the general FASTA release (.tgz archive with k__/p__ taxonomy). When "sintax", downloads a single FASTA file already formatted for VSEARCH SINTAX classification (with tax=d: ... headers).
doi	(Character) If provided, overrides version-based URL construction with a direct DOI download. Useful for older or alternative releases.
verbose	(Logical, default TRUE) Print progress messages.

Details

When taxonomic_format = "default": since UNITE v10.0, the .tgz archive contains all clustering variants (dynamic, 97%, 99%) bundled together. After downloading, extract the archive and select the appropriate FASTA file.

When taxonomic_format = "sintax": the downloaded .gz file is a single FASTA with SINTAX-ready taxonomy headers. This file can be used directly with vsearch --sintax.

The S3 download URLs use opaque UUIDs that change between releases. If the automatic URL fails, visit <https://unite.ut.ee/repository.php> to find the current DOI and pass it via the doi parameter, or download manually.

Please cite UNITE: Abarenkov K et al. (2024) UNITE general FASTA release for Fungi. UNITE Community. [doi:10.15156/BIO/2959336](https://doi.org/10.15156/BIO/2959336)

Value

The path to the downloaded file (invisibly).

Author(s)

Adrien Taudière

See Also

[format2dada2\(\)](#), [format2sintax\(\)](#)

Examples

```
## Not run:
# Download UNITE v10.0 default (general FASTA archive) for fungi
download_unite_db(dest_dir = "databases")

# Download UNITE v10.0 SINTAX-formatted for fungi
download_unite_db(dest_dir = "databases", taxonomic_format = "sintax")

# Download for all eukaryotes in SINTAX format
download_unite_db(
  dest_dir = "databases",
  taxon_group = "eukaryotes",
  taxonomic_format = "sintax"
)

## End(Not run)
```

filter_db

Filter a FASTA database by taxonomic pattern

Description

Filters sequences from a FASTA database whose header lines match a given pattern. Accepts gzip files. May not work on Windows.

Usage

```
filter_db(
  ref_fasta,
  pattern,
  output = NULL,
  force_two_lines_per_seq = TRUE,
  keep_temporary_files = FALSE
)
```

Arguments

ref_fasta	(Character, required) Path to a FASTA file (plain or gzip).
pattern	(Character, required) A pattern to search for in sequence headers.
output	(Character, required) Path to the output FASTA file (must not be gzipped).
force_two_lines_per_seq	(Logical, default TRUE) Force the FASTA file to have exactly two lines per sequence (one header, one nucleotide line). If FALSE, the input must already be in this format.
keep_temporary_files	(Logical, default FALSE) If TRUE and force_two_lines_per_seq is TRUE, keep intermediate temporary files.

Value

The path to the output file (invisibly).

Author(s)

Adrien Taudière

See Also

[count_pattern_db\(\)](#)

Examples

```
db <- system.file("extdata", "example_unite.fasta", package = "dbpq")
out <- tempfile(fileext = ".fasta")
filter_db(db, "Amanita", output = out)
count_seq_db(out)
```

find_vsearch

Find the vsearch executable

Description

Locates the vsearch executable on the system PATH, or verifies a user-supplied path. This is a lightweight standalone version of the helper found in MiscMetabar, so that dbpq does not need to depend on MiscMetabar for vsearch operations.

Usage

```
find_vsearch(path = NULL)
```

Arguments

path (Character, default NULL) Explicit path to the vsearch executable. If NULL, the function searches the system PATH via [Sys.which\(\)](#).

Value

A character string with the path to vsearch, or NA if not found.

See Also

[is_vsearch_installed\(\)](#), [add_sh_to_taxonomy\(\)](#)

Examples

```
find_vsearch()
```

format_fasta_db	<i>Convert a FASTA database to a specified taxonomy format</i>
-----------------	--

Description

Detects (or uses) the input taxonomy format and rewrites sequence headers to the requested output format. This is the primary conversion function; `format2sintax()`, `format2dada2()`, and `format2dada2_species()` are convenience wrappers around it.

Supported **input** formats (prefix-based, with detectable rank labels): "sintax", "unite", "greengenes2".

Supported **output** formats:

- "sintax" — VSEARCH/USEARCH SINTAX (>ID;tax=k:Kingdom,p:Phylum,...)
- "unite" — UNITE default (>ID;k__Kingdom;p__Phylum;...)
- "greengenes2" — Greengenes2 (>ID d__Domain;p__Phylum;...)
- "dada2" — Unprefixed semicolon-delimited (>Kingdom;Phylum;...;)
- "dada2_species" — For dada2::addSpecies() (>ID Genus Species)

Positional formats ("pr2", "dada2") can be detected by `detect_tax_format()` but cannot be used as input for conversion because they lack rank labels.

Usage

```
format_fasta_db(
  fasta_db = NULL,
  taxnames = NULL,
  output_format = c("sintax", "unite", "greengenes2", "dada2", "dada2_species"),
  input_format = "auto",
  output_path = NULL,
  id_prefix = "seq"
)
```

Arguments

fasta_db	(Character) Path to a FASTA file (plain or gzipped). Mutually exclusive with taxnames.
taxnames	(Character vector) Taxonomy header strings (without leading >). Mutually exclusive with fasta_db.
output_format	(Character) Target format. One of "sintax", "unite", "greengenes2", "dada2", "dada2_species".
input_format	(Character, default "auto") Input format. One of "auto" (auto-detect via <code>detect_tax_format()</code>), "sintax", "unite", "greengenes2", "dada2". The positional "dada2" input (taxonomy-only headers, no sequence ID) is assigned ranks by position (d,p,c,o,f,g,s); see id_prefix for the generated labels.
output_path	(Character) If provided and fasta_db is used, write the reformatted FASTA to this path and return the DNASTringSet invisibly.

`id_prefix` (Character, default "seq") Prefix used to build synthetic sequential sequence IDs (e.g. "seq000001") for input formats that carry no per-sequence identifier ("dada2"). Ignored when the input already provides IDs.

Value

If `taxnames` is used, a character vector of reformatted headers. If `fasta_db` is used, a `DNAStringSet` with reformatted names (invisibly when `output_path` is given).

Author(s)

Adrien Taudière

See Also

[detect_tax_format\(\)](#), [format2sintax\(\)](#), [format2dada2\(\)](#), [format2dada2_species\(\)](#)

Examples

```
# UNITE → SINTAX
format_fasta_db(
  taxnames = "AB123;k__Fungi;p__Ascomycota;c__Sordariomycetes",
  output_format = "sintax"
)

# SINTAX → UNITE
format_fasta_db(
  taxnames = "AB123;tax=k:Fungi,p:Ascomycota,c:Sordariomycetes",
  output_format = "unite"
)

# Greengenes2 → dada2
format_fasta_db(
  taxnames = "abc123 d__Bacteria;p__Pseudomonadota;g__Escherichia",
  output_format = "dada2"
)
```

format2dada2

Format taxonomy headers for `dada2::assignTaxonomy`

Description

Converts taxonomy headers to the format expected by `dada2::assignTaxonomy()`: unprefixed semicolon-delimited taxonomy (>Kingdom;Phylum;Class;Order;Family;Genus;). Wrapper around [format_fasta_db\(\)](#).

Usage

```
format2dada2(
  fasta_db = NULL,
  taxnames = NULL,
  input_format = "auto",
  output_path = NULL,
  pattern_to_remove = NULL
)
```

Arguments

<code>fasta_db</code>	(Character) Path to a FASTA file. Mutually exclusive with <code>taxnames</code> .
<code>taxnames</code>	(Character vector) Taxonomy header strings (without leading >). Mutually exclusive with <code>fasta_db</code> .
<code>input_format</code>	(Character, default "auto") Input taxonomy format. One of "auto", "sintax", "unite", "greengenes2".
<code>output_path</code>	(Character) If provided and <code>fasta_db</code> is used, write the reformatted FASTA to this path. The <code>DNAStringSet</code> is returned invisibly.
<code>pattern_to_remove</code>	(Character) Optional regex pattern to remove from the reformatted names (applied after conversion).

Value

If `taxnames` is used, a character vector. If `fasta_db` is used, a `DNAStringSet` with reformatted names. When `output_path` is provided, returned invisibly.

Author(s)

Adrien Taudière

See Also

[format_fasta_db\(\)](#), [format2sintax\(\)](#), [format2dada2_species\(\)](#)

Examples

```
# SINTAX format → dada2
format2dada2(
  taxnames = "AB123;tax=k:Fungi,p:Ascomycota,c:Sordariomycetes"
)

# UNITE format → dada2
format2dada2(
  taxnames = "AB123;k__Fungi;p__Ascomycota;c__Sordariomycetes",
  input_format = "unite"
)
```

format2dada2_species *Format taxonomy headers for dada2::addSpecies*

Description

Converts taxonomy headers to the format expected by `dada2::addSpecies()`: ID Genus Species. Wrapper around `format_fasta_db()`.

Usage

```
format2dada2_species(  
  fasta_db = NULL,  
  taxnames = NULL,  
  input_format = "auto",  
  output_path = NULL  
)
```

Arguments

<code>fasta_db</code>	(Character) Path to a FASTA file. Mutually exclusive with <code>taxnames</code> .
<code>taxnames</code>	(Character vector) Taxonomy header strings (without leading >). Mutually exclusive with <code>fasta_db</code> .
<code>input_format</code>	(Character, default "auto") Input taxonomy format. One of "auto", "sintax", "unite", "greengenes2".
<code>output_path</code>	(Character) If provided and <code>fasta_db</code> is used, write the reformatted FASTA to this path.

Value

If `taxnames` is used, a character vector. If `fasta_db` is used, a `DNAStringSet` with reformatted names.

Author(s)

Adrien Taudière

See Also

`format_fasta_db()`, `format2sintax()`, `format2dada2()`

Examples

```
# UNITE format → dada2_species  
format2dada2_species(  
  taxnames = "AB123;k__Fungi;g__Aspergillus;s__fumigatus"  
)
```

```
# SINTAX format → dada2_species
format2dada2_species(
  taxnames = "AB123;tax=k:Fungi,g:Aspergillus,s:fumigatus",
  input_format = "sintax"
)
```

format2sintax

*Format taxonomy headers to SINTAX format***Description**

Converts taxonomy headers to the VSEARCH SINTAX format (>ID;tax=k:Kingdom,p:Phylum,...). Wrapper around [format_fasta_db\(\)](#).

Usage

```
format2sintax(
  fasta_db = NULL,
  taxnames = NULL,
  input_format = "auto",
  output_path = NULL,
  id_prefix = "seq"
)
```

Arguments

fasta_db	(Character) Path to a FASTA file. Mutually exclusive with taxnames.
taxnames	(Character vector) Taxonomy header strings (without leading >). Mutually exclusive with fasta_db.
input_format	(Character, default "auto") Input taxonomy format. One of "auto", "unite", "greengenes2", "dada2".
output_path	(Character) If provided and fasta_db is used, write the reformatted FASTA to this path.
id_prefix	(Character, default "seq") Prefix for synthetic sequence IDs generated when the input has none (e.g. "dada2").

Value

If taxnames is used, a character vector of reformatted names. If fasta_db is used, a DNASTringSet with reformatted names.

Author(s)

Adrien Taudière

See Also

[format_fasta_db\(\)](#), [format2dada2\(\)](#), [format2dada2_species\(\)](#)

Examples

```
# UNITE format → SINTAX
format2sintax(taxnames = "AB123;k__Fungi;p__Ascomycota;c__Sordariomycetes")

# Greengenes2 format → SINTAX
format2sintax(
  taxnames = "abc123 d__Bacteria;p__Pseudomonadota",
  input_format = "greengenes2"
)

# dada2 trainset (taxonomy-only, positional) → SINTAX with synthetic IDs
format2sintax(
  taxnames = "Bacteria;Pseudomonadota;Gammaproteobacteria;Vibrio;",
  input_format = "dada2",
  id_prefix = "SILVA_"
)
```

get_file_extension	<i>Get file extension(s)</i>
--------------------	------------------------------

Description

Returns all extensions from a file name. Double extensions such as `.fasta.gz` are treated as a first-class case and returned as a two-element vector (e.g. `c("fasta", "gz")`).

Usage

```
get_file_extension(file_path)
```

Arguments

file_path (Character, required) Path to a file.

Value

A character vector of file extensions (one element per extension).

Examples

```
get_file_extension("my_database.fasta")
get_file_extension("my_database.fasta.gz")
```

is_vsearch_installed	<i>Check whether vsearch is installed</i>
----------------------	---

Description

Tests whether the vsearch executable is available on the system.

Usage

```
is_vsearch_installed(path = find_vsearch())
```

Arguments

path	(Character, default NULL) Explicit path to the vsearch executable. If NULL, the function searches the system PATH.
------	--

Value

TRUE if vsearch is available, FALSE otherwise.

See Also

[find_vsearch\(\)](#), [add_sh_to_taxonomy\(\)](#)

Examples

```
is_vsearch_installed()
```

list_ranks_db	<i>List and count taxonomic ranks from a FASTA database</i>
---------------	---

Description

Extracts and counts occurrences of a given taxonomic rank from FASTA sequence headers. Supports both prefix-based formats (unite, syntax, greengenes2) and positional formats (dada2, pr2).

Usage

```
list_ranks_db(
  file,
  rank_prefix = "k__",
  tax_format = NULL,
  rank_position = NULL
)
```

Arguments

file	(Character, required) Path to a FASTA file (plain or gzip).
rank_prefix	(Character, default "k__") The prefix identifying the taxonomic rank to extract (e.g., "k__" for kingdom, "p__" for phylum). Ignored if tax_format is provided.
tax_format	(Character) If provided, one of "unite", "sintax", "greengenes2", or "pr2". Overrides rank_prefix with the first rank from tax_prefixes() . If NULL (default), rank_prefix is used as-is.
rank_position	(Integer) For positional (prefix-less) taxonomy headers, the 1-based position of the rank to extract from the semicolon-delimited string. Can be used with tax_format = "pr2" or standalone (without tax_format). Ignored for prefix-based formats.

Value

A named integer vector of counts, sorted in decreasing order. Names are the taxonomic rank values.

Author(s)

Adrien Taudière

See Also

[tax_prefixes\(\)](#), [detect_tax_format\(\)](#), [summarize_db\(\)](#)

Examples

```
db <- system.file("extdata", "example_unite.fasta", package = "dbpq")
list_ranks_db(db, rank_prefix = "p__")
list_ranks_db(db, tax_format = "unite")
```

profile_db

Profile the taxonomic content of one or several FASTA databases

Description

Builds a taxonomic profile of one or more FASTA reference databases. It first runs [diagnose_db\(\)](#) (format/integrity/quality checks) and then adds:

- a **richness** table and bar plot giving the number of distinct taxa (levels) annotated at each taxonomic rank, and
- when several databases are supplied, a **cross-database comparison** of the taxa present at each rank, drawn as a Venn diagram ([ggVennDiagram](#), up to venn_max databases) or an UpSet plot ([ComplexUpset](#)), with one plot per rank.

By default the comparison overlaps the *sets of distinct taxon names* at each rank (presence/absence). With weight_by_seqs = TRUE the UpSet bars instead show the total number of sequences backing the taxa in each intersection (summed across the databases that contain each taxon).

Usage

```
profile_db(
  files,
  tax_format = "auto",
  weight_by_seqs = FALSE,
  plot = TRUE,
  venn_max = 7L,
  verbose = TRUE,
  ...
)
```

Arguments

<code>files</code>	(Character vector, required) One or more paths to FASTA files (plain or gzip).
<code>tax_format</code>	(Character, default "auto") Taxonomy format, passed to <code>diagnose_db()</code> and used for taxon extraction. One of "auto", "unite", "sintax", "greengenes2", "pr2", or "dada2".
<code>weight_by_seqs</code>	(Logical, default FALSE) When TRUE, the cross-database UpSet plot weights each intersection by the number of sequences (summed over the databases containing each taxon) rather than by the number of taxa. Forces an UpSet plot (Venn diagrams cannot be weighted) and therefore needs ComplexUpset ; with <code>ggplot2</code> $\geq 4.0.0$ this requires the dev version ($\geq 1.3.6$, install with <code>remotes::install_github("krassowski/ggvenn")</code>). When it is unavailable, an unweighted Venn is drawn instead and the weighted counts are still returned in <code>comparison\$signatures</code> .
<code>plot</code>	(Logical, default TRUE) Whether to build plots. Requires ggplot2 ; the comparison plots additionally require ggVennDiagram and/or ComplexUpset .
<code>venn_max</code>	(Integer, default 7) Maximum number of databases for which a Venn diagram is drawn; beyond this an UpSet plot is used. (Also forced to UpSet when <code>weight_by_seqs = TRUE</code> .)
<code>verbose</code>	(Logical, default TRUE) Whether to show a cli progress bar while reading files and print a summary report.
<code>...</code>	Further arguments passed to <code>diagnose_db()</code> (e.g. <code>min_length</code> , <code>check_duplicates</code>).

Value

An object of class "dbpq_profile": a list with components

`diagnosis` The `diagnose_db()` result (a `dbpq_diagnosis`).

`taxa` A long `tibble` (file, rank, taxon, n_seqs) of every taxon found at every rank.

`richness` A `tibble` (file, rank, n_levels, n_seqs_annotated) of per-rank taxonomic richness.

`comparison` NULL for a single file; otherwise a list with `per_db` and `signatures` tibbles, per-rank membership data frames, and per-rank plots.

`plots` A list with `richness` (a `ggplot`) and `comparison` (a named list of per-rank Venn/UpSet plots, or NULL).

Author(s)

Adrien Taudière

See Also[diagnose_db\(\)](#), [summarize_db\(\)](#), [list_ranks_db\(\)](#)**Examples**

```
unite <- system.file("extdata", "example_unite.fasta", package = "dbpq")
prof <- profile_db(unite, verbose = FALSE)
prof$richness

# Compare two databases (needs ggVennDiagram / ComplexUpset for the plots)

sintax <- system.file("extdata", "example_sintax.fasta", package = "dbpq")
prof2 <- profile_db(c(unite, sintax), verbose = FALSE)
prof2$comparison$signatures
```

search_taxa_db

*Search sequences by taxonomic name in a FASTA database***Description**

Searches a FASTA database (plain or gzip) for sequences whose header matches a set of taxonomic name parts. This is useful to retrieve all sequences assigned to a given taxon across databases that use different header formats (e.g. UNITE `g__Apiotrichum;s__akiyoshidainum` vs SINTAX `k:Fungi,...,g:Apiotrichum,s:akiyoshidainum`).

Matching is case-insensitive by default and based on substrings, so the rank prefixes and separators used by a given format do not matter. Provide multiple name parts in `taxa` to narrow the search: by default a header matches only when **all** parts are present (`match = "all"`); use `match = "any"` to match when at least one part is present.

On Linux/macOS, filtering is done with `awk` in a single streaming pass over the file, so memory usage stays low even for multi-GB reference databases — only the matched records are loaded into R. On Windows (or whenever `awk` is unavailable), the function automatically falls back to an in-memory `Biostrings::readDNASTringSet()` filter, which is slower and more memory-hungry for large databases but produces the same result.

Usage

```
search_taxa_db(
  file,
  taxa,
  match = c("all", "any"),
  case_sensitive = FALSE,
  output_path = NULL
)
```

Arguments

file	(Character, required) Path to a FASTA file (plain or gzip).
taxa	(Character, required) One or more taxonomic name parts to search for in sequence headers, e.g. c("Apiotrichum", "akiyoshidainum").
match	(Character, default "all") Whether a header must contain all ("all") or at least one ("any") of the taxa parts.
case_sensitive	(Logical, default FALSE) If TRUE, matching is case-sensitive.
output_path	(Character, default NULL) Optional path to write the matched sequences to a FASTA file. A .gz extension produces a gzipped file. When set, the Biostrings::DNASTringSet is returned invisibly.

Value

A [Biostrings::DNASTringSet](#) of the matching sequences, with original headers preserved (including all taxonomy information). An empty DNASTringSet (length 0) is returned when nothing matches.

Author(s)

Adrien Taudière

See Also

[filter_db\(\)](#) for a shell-based header filter that writes to a file; [format_fasta_db\(\)](#) to reformat the retrieved sequences.

Examples

```
db <- system.file("extdata", "example_unite.fasta", package = "dbpq")

# Retrieve all Amanita sequences (single name part)
search_taxa_db(db, "Amanita")

# Narrow to a species with two name parts (AND logic)
search_taxa_db(db, c("Amanita", "muscaria"))

# Match any of several genera (OR logic)
search_taxa_db(db, c("Amanita", "Fusarium"), match = "any")

# Write the result to a gzipped FASTA file and read it back
out <- tempfile(fileext = ".fasta.gz")
search_taxa_db(db, "Boletus", output_path = out)
Biostrings::readDNASTringSet(out)
```

summarize_db

*Summarize a FASTA reference database***Description**

Provides an overview of a FASTA reference database: number of sequences, sequence length distribution, and taxonomic coverage at each rank. Supports both prefix-based formats (unite, syntax, greengenes2) and positional formats (dada2, pr2).

Usage

```
summarize_db(
  file,
  rank_prefixes = c("k__", "p__", "c__", "o__", "f__", "g__", "s__"),
  tax_format = NULL
)
```

Arguments

file	(Character, required) Path to a FASTA file (plain or gzip).
rank_prefixes	(Character vector) Taxonomic rank prefixes to summarize. Defaults to kingdom through species. Ignored if tax_format is provided.
tax_format	(Character) If provided, one of "unite", "syntax", "greengenes2", or "pr2". Overrides rank_prefixes with the full set from tax_prefixes() . If "auto", the format is detected from the file headers using detect_tax_format() . If NULL (default), rank_prefixes is used as-is.

Value

A list with components:

- n_sequences: total number of sequences
- length_summary: summary statistics of sequence lengths
- ranks: a named integer vector of annotated counts per rank

Author(s)

Adrien Taudière

See Also

[tax_prefixes\(\)](#), [detect_tax_format\(\)](#), [list_ranks_db\(\)](#)

Examples

```
db <- system.file("extdata", "example_unite.fasta", package = "dbpq")
summarize_db(db)
summarize_db(db, tax_format = "unite")
```

tax_prefixes

Get rank information for a taxonomy format

Description

Returns the taxonomic rank information for common reference database formats. For prefix-based formats (unite, sintax, greengenes2), returns a named character vector of prefixes. For positional formats (pr2), returns a named integer vector of rank positions.

Use the result with [list_ranks_db\(\)](#) and [summarize_db\(\)](#) via their tax_format parameter.

Note: `dada2::assignTaxonomy()` is a classifier, not a taxonomy format. It accepts any semicolon-separated taxonomy with any number of levels, regardless of whether prefixes are present or not. Use the taxLevels argument in `dada2::assignTaxonomy()` to specify the rank names.

Usage

```
tax_prefixes(tax_format = c("unite", "sintax", "greengenes2", "pr2"))
```

Arguments

- tax_format (Character) One of:
- "unite": k__/p__/_... format used by UNITE general FASTA releases.
 - "sintax": d:/k:/p:/... format used by VSEARCH SINTAX and USEARCH UTX databases (UNITE SINTAX, PR2 UTX). Note that UNITE SINTAX files use k: (kingdom) as their first rank and do not include d: (domain). When calling [summarize_db\(\)](#) on a UNITE SINTAX file, the d: row will show 0 sequences — this is expected.
 - "greengenes2": d__/p__/_... format used by Greengenes2 (starts with domain d__ instead of kingdom k__).
 - "pr2": positional format with 9 levels specific to protist taxonomy: Domain, Supergroup, Division, Subdivision, Class, Order, Family, Genus, Species.

Value

For prefix-based formats: a named character vector of rank prefixes. For positional formats: a named integer vector of rank positions.

See Also

[list_ranks_db\(\)](#), [summarize_db\(\)](#), [detect_tax_format\(\)](#)

Examples

```
tax_prefixes("unite")
tax_prefixes("sintax")
tax_prefixes("greengenes2")
tax_prefixes("pr2")
```

Index

add_sh_to_taxonomy, [3](#)
add_sh_to_taxonomy(), [33](#), [40](#)

Biostrings::DNAStringSet, [44](#)
Biostrings::readDNAStringSet(), [43](#)

count_pattern_db, [5](#)
count_pattern_db(), [6](#), [33](#)
count_seq_db, [6](#)
count_seq_db(), [5](#)
count_unwanted_tax, [6](#)
count_unwanted_tax(), [11](#), [12](#)
cutadapt_rm_primers_db, [8](#)

dada2::assignTaxonomy(), [26](#)
detect_tax_format, [10](#)
detect_tax_format(), [7](#), [11](#), [12](#), [34](#), [35](#), [41](#),
[45](#), [46](#)
diagnose_db, [11](#)
diagnose_db(), [41–43](#)
download_bold_db, [12](#)
download_diatbarcode_db, [14](#)
download_eukaryome_db, [15](#)
download_greengenes2_db, [16](#)
download_greengenes2_db(), [22](#)
download_ksgp_db, [18](#)
download_ksgp_db(), [22](#)
download_ltpplus_db, [20](#)
download_marjaam_db, [22](#)
download_midori2_db, [24](#)
download_pr2_db, [25](#)
download_pr2_db(), [20](#)
download_rdp_db, [27](#)
download_silva_db, [28](#)
download_silva_db(), [20](#), [22](#), [28](#)
download_unite_db, [30](#)
download_unite_db(), [3](#), [4](#)

filter_db, [32](#)
filter_db(), [5](#), [44](#)

find_vsearch, [33](#)
find_vsearch(), [3](#), [4](#), [40](#)
format2dada2, [35](#)
format2dada2(), [17](#), [25](#), [27](#), [28](#), [30](#), [31](#), [34](#),
[35](#), [37](#), [38](#)
format2dada2_species, [37](#)
format2dada2_species(), [34–36](#), [38](#)
format2sintax, [38](#)
format2sintax(), [20](#), [25](#), [27](#), [30](#), [31](#), [34–37](#)
format_fasta_db, [34](#)
format_fasta_db(), [35–38](#), [44](#)

get_file_extension, [39](#)

is_vsearch_installed, [40](#)
is_vsearch_installed(), [4](#), [33](#)

list_ranks_db, [40](#)
list_ranks_db(), [8](#), [10](#), [16](#), [43](#), [45](#), [46](#)

MiscMetabar::add_new_taxonomy_pq(), [26](#)
MiscMetabar::unwanted_tax_patterns, [7](#)

phyloseq, [6](#), [7](#)
profile_db, [41](#)

search_taxa_db, [43](#)
summarize_db, [45](#)
summarize_db(), [8](#), [10](#), [12](#), [16](#), [41](#), [43](#), [46](#)
Sys.which(), [33](#)

tax_prefixes, [46](#)
tax_prefixes(), [10](#), [17](#), [41](#), [45](#)
tibble, [7](#), [12](#), [42](#)