Package 'algaeClassify'

May 14, 2025

Title Tools to Query the 'Algaebase' Online Database, Standardize Phytoplankton Taxonomic Data, and Perform Functional Group Classifications

Version 2.0.4

Date 2025-05-12

Author Vijay Patil [aut, cre],
Torsten Seltmann [aut],
Nico Salmaso [aut],
Orlane Anneville [aut],
Marc Lajeunesse [aut],
Dietmar Straile [aut]

Maintainer Vijay Patil <vij.patil@gmail.com>

Description Functions that facilitate the use of accepted taxonomic nomenclature, collection of functional trait data, and assignment of functional group classifications to phytoplankton species. Possible classifications include Morpho-functional group (MFG; Salmaso et al. 2015 <doi:10.1111/fwb.12520>) and CSR (Reynolds 1988; Functional morphology and the adaptive strategies of phytoplankton. In C.D. Sandgren (ed). Growth and reproductive strategies of freshwater phytoplankton, 388-433. Cambridge University Press, New York). Versions 2.0.0 and later includes new functions for querying the 'algaebase' online taxonomic database (www.algaebase.org), however these functions require a valid API key that must be acquired from the 'algaebase' administrators.

Note that none of the 'algaeClassify' authors are affiliated with 'algaebase' in any way. Taxonomic names can also be checked against a variety of taxonomic databases using the 'Global Names Resolver' service via its API (<https://resolver.globalnames.org/api>). In addition, currently accepted and outdated synonyms, and higher taxonomy, can be extracted for lists of species from the 'ITIS' database using wrapper functions for the ritis package.

The 'algaeClassify' package is a product of the GEISHA (Global Evaluation of the Impacts of Storms on freshwater Habitat and Structure of phytoplankton Assemblages), funded by CESAB (Centre for Synthesis and Analysis of Biodiversity) and the U.S. Geological Survey John Wesley Powell Center for Synthesis and Analysis, with data and other support provided by members of GLEON (Global Lake Ecology Observation Network).

DISCLAIMER: This software has been approved for release by the

U.S. Geological Survey (USGS). Although the software has been subjected to rigorous review, the USGS reserves the right to update the software as needed pursuant to further analysis and review. No warranty, expressed or implied, is made by the USGS or the U.S. Government as to the
functionality of the software and related material nor shall the fact of release constitute any such warranty. Furthermore, the software is released on condition that neither the USGS nor the U.S. Government shall be held liable for any damages resulting from its authorized or unauthorized use.

Depends R (>= 4.4.0)

URL https://doi.org/10.5066/F7S46Q3F

Imports lubridate, stats, ritis, curl, jsonlite, methods, RCurl

License CC0

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Date/Publication 2025-05-14 08:50:08 UTC

Contents

`accum` *Split a dataframe column with binomial name into genus and species columns. Plots change in species richness over time, generates species accumulation curve, and compares SAC against simulated idealized curve assuming all unique taxa have equal probability of being sampled at any point in the time series. (author Dietmar Straile)*

Description

Split a dataframe column with binomial name into genus and species columns. Plots change in species richness over time, generates species accumulation curve, and compares SAC against simulated idealized curve assuming all unique taxa have equal probability of being sampled at any point in the time series. (author Dietmar Straile)

Usage

```
accum(
  b_data,
  phyto_name = "phyto_name",
  column = NA,
  n = 100,
  save.pdf = FALSE,
  lakename = "",
  datename = "date_dd_mm_yy",
  dateformat = "%d-%m-%y"
)
```

Arguments

`b_data`	Name of data.frame object
`phyto_name`	Character string: field containing phytoplankton id (species, genus, etc.)
`column`	column name or number for field containing abundance (biomass,biovol, etc.). Can be NA if the dataset only contains a species list for each sampling date.
`n`	number of simulations for randomized ideal species accumulation curve

save.pdf	TRUE/FALSE- should plots be displayed or saved to a pdf?
lakename	optional character string for adding lake name to pdf output
datename	character string name of b_data field containing date
dateformat	character string: posix format for datename column

Value

a two panel plot with trends in richness on top, and cumulative richness vs. simulated accumulation curve on bottom

Examples

```
data(lakegeneva)
#example dataset with 50 rows
head(lakegeneva)

accum(b_data=lakegeneva,column='biovol_um3_ml',n=10,save.pdf=FALSE)
```

---

algaebase_genus_search

Search algaebase for information about a genus of phytoplankton

---

Description

Search algaebase for information about a genus of phytoplankton

Usage

```
algaebase_genus_search(
  genus = NULL,
  apikey = NULL,
  handle = NULL,
  higher = TRUE,
  print.full.json = FALSE,
  newest.only = TRUE,
  long = FALSE,
  exact.matches.only = TRUE,
  return.higher.only = FALSE,
  api_file = NULL
)
```

Arguments

genus	genus name as character string
apikey	valid key for algaebase API as character string
handle	curl handle with API key. Will be created if not present.

higher	boolean should higher taxonomy be included in output?
print.full.json	boolean returns raw json output if TRUE. Default is FALSE (return R data frame)
newest.only	boolean should results be limited to the most recent matching entry in algaebase?
long	boolean return long output including full species name and authorship, and entry date from algaebase.
exact.matches.only	boolean should results be limited to exact matches?
return.higher.only	boolean should output only included higher taxonomy?
api_file	path to text file containing a valid API key

Value

data frame that may include: accepted.name (currently accepted synonym if different from input name), input.name (name supplied by user), input.match (1 if exact match, else 0), currently.accepted (1=TRUE/0=FALSE), genus.only (1=genus search/0=genus+species search),higher taxonomy (kingdom,phylum,class,order,family), genus, species (always NA for genus search), infraspecies name (always NA for genus search), long.name (includes author and date if given), taxonomic.status (currently accepted, synonym, or unverified), taxon.rank (taxonomic rank of accepted name (genus, species, infraspecies), mod.date (date when entry was last modified in algaebase).

Examples

```
## Not run: algaebase_genus_search("Anabaena") #not run.
```

---

algaebase_output_parse

Helper function for parsing output from algaebase

Description

Helper function for parsing output from algaebase

Usage

```
algaebase_output_parse(x, field.name)
```

Arguments

x	list object containing output from an algaebase query
field.name	character string

Value

selected output variable as character vector

`algaebase_search_df` *Search algaebase for information about a list of phytoplankton names*

Description

Search algaebase for information about a list of phytoplankton names

Usage

```
algaebase_search_df(
  df,
  apikey = NULL,
  handle = NULL,
  genus.only = FALSE,
  genus.name = "genus",
  species.name = "species",
  higher = TRUE,
  print.full.json = FALSE,
  long = FALSE,
  exact.matches.only = TRUE,
  api_file = NULL,
  sleep.time = 1
)
```

Arguments

`df`	data frame containing columns with genus and species names
`apikey`	valid key for algaebase API as character string
`handle`	curl handle with API key. Will be created if not present.
`genus.only`	boolean: should searches be based solely on the genus name?
`genus.name`	name of data.frame column that contains genus names
`species.name`	name of data.frame column that contains species names
`higher`	boolean should higher taxonomy be included in output?
`print.full.json`	boolean returns raw json output if TRUE. Default is FALSE (return R data frame)
`long`	boolean return long output including full species name and authorship, and entry date from algaebase.
`exact.matches.only`	boolean should results be limited to exact matches?
`api_file`	path to text file containing a valid API key
`sleep.time`	delay between algaebase queries (in seconds). Should be at least 1 second if querying more than 10 names at once.

Value

data frame that may include: accepted.name (currently accepted synonym if different from input name), input.name (name supplied by user), input.match (1 if exact match, else 0), currently.accepted (1=TRUE/0=FALSE), genus.only (1=genus search/0=genus+species search),higher taxonomy (kingdom,phylum,class,order,family), genus, species (always NA for genus search), infraspecies name (always NA for genus search), long.name (includes author and date if given), taxonomic.status (currently accepted, synonym, or unverified), taxon.rank (taxonomic rank of accepted name (genus, species, infraspecies), mod.date (date when entry was last modified in algaebase).

Examples

```
## Not run:
data(lakegeneva)
#example dataset with 50 rows

new.lakegeneva <- genus_species_extract(lakegeneva,'phyto_name')
lakegeneva.algaebase<-algaebase_search_df(new.lakegeneva[1:10,],higher=TRUE,long=TRUE)
head(lakegeneva.algaebase)
## End(Not run)
```

`algaebase_species_search`

Retrieve taxonomic information from the algaebase online database (www.algaebase.org) based on a user-specified genus and species name . This function requires a valid API key for algaebase.

Description

Retrieve taxonomic information from the algaebase online database (www.algaebase.org) based on a user-specified genus and species name . This function requires a valid API key for algaebase.

Usage

```
algaebase_species_search(
  genus,
  species,
  apikey = NULL,
  handle = NULL,
  higher = TRUE,
  print.full.json = FALSE,
  newest.only = TRUE,
  long = FALSE,
  exact.matches.only = TRUE,
  api_file = NULL
)
```

Arguments

genus	genus name as character string
species	species name as character string
apikey	valid key for algaebase API as character string
handle	curl handle with API key. Will be created if not present.
higher	boolean should higher taxonomy be included in output?
print.full.json	boolean returns raw json output if TRUE. Default is FALSE (return R data frame)
newest.only	boolean should results be limited to the most recent matching entry in algaebase?
long	boolean return long output including full species name and authorship, and entry date from algaebase.
exact.matches.only	boolean should results be limited to exact matches?
api_file	path to text file containing a valid API key

Value

data frame that may include: accepted.name (currently accepted synonym if different from input name), input.name (name supplied by user), input.match (1 if exact match, else 0), currently.accepted (1=TRUE/0=FALSE), genus.only (1=genus search/0=genus+species search),higher taxonomy (kingdom,phylum,class,order,family), genus, species (always NA for genus search), infraspecies name (always NA for genus search), long.name (includes author and date if given), taxonomic.status (currently accepted, synonym, or unverified), taxon.rank (taxonomic rank of accepted name (genus, species, infraspecies), mod.date (date when entry was last modified in algaebase).

Examples

```
## Not run: algaebase_species_search("Anabaena flos-aquae") #not run
```

---

bestmatch *fuzzy partial matching between a scientific name and a list of possible matches*

Description

fuzzy partial matching between a scientific name and a list of possible matches

Usage

```
bestmatch(enteredName, possibleNames, maxErr = 3, trunc = TRUE)
```

Arguments

enteredName	Character string with name to check
possibleNames	Character vector of possible matches
maxErr	maximum number of different bits allowed for a partial match
trunc	TRUE/FALSE. if true and no match, retry with last three letters truncated

Value

a character string with the best match, or 'multiplePartialMatches'

Examples

```
possibleMatches=c('Viburnum edule','Viburnum acerifolia')
bestmatch(enteredName='Viburnum edulus',possibleNames=possibleMatches)
```

---

csrTraits — *Database of functional traits for MFG classification, derived from Rimet et al. 2019*

Description

Database of functional traits for MFG classification, derived from Rimet et al. 2019

Usage

```
data(mfgTraits)
```

Format

A data frame with columns:

phyto_name binomial scientific name

genus genus name

species species name

SAV surface area:volume ratio

MLD maximum linear dimension (micrometers)

MSV product of SAV and MLD; unitless

volume.um3 cell or colony biovolume

surface.area.um2 biological unit (cell or colony) surface area accounting for mucilage

Colonial 1/0 indicates colonial growth form

Number.of.cells.per.colony literature-based average colony abundance

Geometrical.shape.of.the.colony Shape descriptions. See Rimet et al. 2019 for abbreviations

traitCSR CSR classification using traits_to_CSR function and criteria from Reynolds 2006

`date_mat` *Transform a phytoplankton timeseries into a matrix of abundances for ordination*

Description

Transform a phytoplankton timeseries into a matrix of abundances for ordination

Usage

```
date_mat(
  phyto.df,
  abundance.var = "biovol_um3_ml",
  summary.type = "abundance",
  taxa.name = "phyto_name",
  date.name = "date_dd_mm_yy",
  format = "%d-%m-%y",
  time.agg = c("day", "month", "year", "monthyear"),
  fun = mean_naomit
)
```

Arguments

`phyto.df`	Name of data.frame object
`abundance.var`	Character string: field containing abundance data. Can be NA if the dataset only contains a species list for each sampling date.
`summary.type`	'abundance' for a matrix of aggregated abundance,'presence.absence' for 1 (present) and 0 (absent).
`taxa.name`	Character string: field containing taxonomic identifiers.
`date.name`	Character string: field containing date.
`format`	Character string: POSIX format string for formatting date column.
`time.agg`	Character string: time interval for aggregating abundance. default is day.
`fun`	function for aggregation. default is mean, excluding NA's.

Value

A matrix of phytoplankton abundance, with taxa in rows and time in columns. If time.agg = 'monthyear', returns a 3dimensional matrix (taxa,month,year). If abundance.var = NA, matrix cells will be 1 for present, 0 for absent

Examples

```
data(lakegeneva)
#example dataset with 50 rows

geneva.mat1<-date_mat(lakegeneva,time.agg='month',summary.type='presence.absence')
```

```
geneva.mat2<-date_mat(lakegeneva,time.agg='month',summary.type='abundance')

geneva.mat1
geneva.mat2
```

---

genus_search_itis *Wrapper function for several functions in ritis:: Searches ITIS database for matches to a genus name*

Description

Wrapper function for several functions in ritis:: Searches ITIS database for matches to a genus name

Usage

```
genus_search_itis(genus, higher = FALSE)
```

Arguments

genus	Character string. genus name to search for in ITIS
higher	Boolean. If TRUE, add higher taxonomic classifications to output

Value

input data.frame with matches, current accepted names, synonyms, and higher taxonomy

Examples

```
genus='Anabaena'
genus_search_itis(genus,higher=FALSE)
```

---

genus_species_extract *Split a dataframe column with binomial name into genus and species columns.*

Description

Split a dataframe column with binomial name into genus and species columns.

Usage

```
genus_species_extract(phyto.df, phyto.name)
```

Arguments

phyto.df	Name of data.frame object
phyto.name	Character string: field in phyto.df containing species name.

Value

A data.frame with new character fields 'genus' and 'species'

Examples

```
data(lakegeneva)
#example dataset with 50 rows

head(lakegeneva) #need to split the phyto_name column
new.lakegeneva=genus_species_extract(lakegeneva,'phyto_name')

head(new.lakegeneva)
```

---

get_apikey *Get value of algaebase API key from Environment variable Return an error if variable not set.*

Description

Get value of algaebase API key from Environment variable Return an error if variable not set.

Usage

```
get_apikey()
```

Value

api key as character string (invisibly)

---

get_apikey_fromfile *Get value of algaebase API key from a file*

Description

Get value of algaebase API key from a file

Usage

```
get_apikey_fromfile(keyfile)
```

Arguments

keyfile	path to text file

Value

api key as character string (invisibly)

Examples

```
## Not run: apikey<-get_apikey_fromfile("keyfile.txt")
```

---

gnr_df *Wrapper function to apply gnr_simple across a data.frame or list of species names*

Description

Provides convienent output with a row per name. To streamline merging with original data.

Usage

```
gnr_df(
  df,
  name.column,
  sourceid = NULL,
  best_match = TRUE,
  fuzzy_uninomial = TRUE,
  canonical = TRUE,
  with_context = TRUE,
  higher = FALSE
)
```

Arguments

df	data.frame containing names to check
name.column	integer or character string with column name containing species names
sourceid	integer vector with data source ids. see https://resolver.globalnames.org/sources/
best_match	boolean. Should the best match be returned based on score?
fuzzy_uninomial	boolean. Use fuzzy matching for uninomial names?
canonical	If TRUE, names do not include authorship or date
with_context	If TRUE, Match scores are weighted for taxonomic consistency
higher	boolean: Return higher taxonomic classifications?

Value

new data.frame original names (input_name), 1/0 flag for an exact match,the best match (match_name, and other output from gnr_simple(). Will contain a row of NAs if no matches were found for a name.

Examples

```
data(lakegeneva)
#example dataset with 50 rows

lakegeneva<- genus_species_extract(lakegeneva,'phyto_name')
lakegeneva$genus_species <- trimws(paste(lakegeneva$genus,
lakegeneva$species))

#checking for matches from all GNRS sources, first 5 rows:
lakegeneva.namematches <- gnr_df(lakegeneva,"genus_species")
lakegeneva.namematches
```

---

gnr_simple *checks species names against a variety of online databases supports fuzzy partial matching, using the Global Names Resolver (https://resolver.globalnames.org/)*

Description

Provides convienent output with a single result, using a variety of criteria for the best match

Usage

```
gnr_simple(
  name,
  sourceid = NULL,
  best_match = TRUE,
  fuzzy_uninomial = TRUE,
  canonical = TRUE,
  with_context = TRUE,
  higher = FALSE
)
```

Arguments

name	character string binomial scientific name to resolve
sourceid	integer vector with data source ids. see https://resolver.globalnames.org/sources/
best_match	boolean. Should the best match be returned based on score?
fuzzy_uninomial	boolean. Use fuzzy matching for uninomial names?
canonical	boolean. return canonical name?
with_context	boolean. Return context (auther of species name?)
higher	boolean: Return higher taxonomic classifications?

Value

new data.frame with name matches, column indicating match type and scores from Global Names Resolver (https://resolver.globalnames.org/). Will contain a row of NAs if no matches found

Examples

```
#Visit https://resolver.globalnames.org/data_sources to see all possible
#data sources for name checking.
name<-"Aphanazomenon flos-aquae"
#sourceid=3 for ITIS database,195 for Algaebase
gnr_simple(name,sourceid=3) #search for ITIS matches
gnr_simple(name,sourceid=NULL) #search for matches from any source
```

---

itis_search_df *Wrapper function for applying genus_search_itis and species_search_itis to a whole data.frame containing scientific names*

Description

Wrapper function for applying genus_search_itis and species_search_itis to a whole data.frame containing scientific names

Usage

```
itis_search_df(df, namecol = NA, higher = FALSE, genus.only = FALSE)
```

Arguments

df	data.frame containing names to check
namecol	integer or character string with column name containing species or genus names
higher	Boolean. If TRUE, add higher taxonomic classifications to output
genus.only	boolean If TRUE, search for matches with just the genus name using genus_search_itis

Value

data.frame with submitted names (orig.name), matched names (matched.name), 1/0 flag indicating that original name is currently accepted (orig.name.accepted), 1/0 flag indicating if search was genus_only (for distinguishing genus_search_itis and species_search_itis results), synonyms if any, and higher taxonomy (if higher=TRUE)

Examples

```
data(lakegeneva)
#example dataset

new.lakegeneva <- genus_species_extract(lakegeneva[1,],'phyto_name')
new.lakegeneva$genus_species <- trimws(paste(new.lakegeneva$genus,
new.lakegeneva$species))
#checking for genus-only name matches in ITIS, and extracting higher taxonomy
#flagging names with imperfect or no matches

lakegeneva.genus.itischeck <-
         itis_search_df(new.lakegeneva,"genus_species")
         lakegeneva.genus.itischeck
```

---

lakegeneva *example dataset from lake Geneva, Switzerland*

Description

example dataset from lake Geneva, Switzerland

Usage

```
data(lakegeneva)
```

Format

A data frame with columns:

lake lake name

phyto_name phytoplankton species name

month month of sampling

year year of sampling

date_dd_mm_yy date of sampling

biovol_um3_ml biovolume

mean_naomit *Compute mean value while ignoring NA's*

Description

Compute mean value while ignoring NA's

Usage

```
mean_naomit(x)
```

Arguments

x	A numeric vector that may contain NA's

Value

the mean value

Examples

```
data(lakegeneva)
#example dataset with 50 rows

mean_naomit(lakegeneva$biovol_um3_ml)
```

mfgTraits *Functional Trait Database derived from Rimet et al.*

Description

Functional Trait Database derived from Rimet et al.

Usage

```
data(mfgTraits)
```

Format

A data frame with columns:

phyto_name binomial scientific name

genus genus name

species species name

Mobility.apparatus 1/0 indicates presence/absence of flagella or motility

Size character values 'large' or 'small'; based on 35 micrometer max linear dimension

Colonial 1/0 indicates typical colonial growth form or not

Filament 1/0 indicates filamentous growth form or not

Centric 1/0 indicates diatoms with centric growth form

Gelatinous 1/0 indicates presence/absence of mucilage

Aerotopes 1/0 indicates presence/absence of aerotopes

Class Taxonomic class

Order Taxonomic order

MFG.fromtraits MFG classification using traits_to_mfg function

---

`mfg_csr_convert` *Returns a CSR classification based on Morphofunctional group (MFG). Correspondence based on Salmaso et al. 2015 and Reynolds et al. 1988*

Description

Returns a CSR classification based on Morphofunctional group (MFG). Correspondence based on Salmaso et al. 2015 and Reynolds et al. 1988

Usage

```
mfg_csr_convert(mfg)
```

Arguments

`mfg`	Character string with MFG name, following Salmaso et al. 2015

Value

A character string with values 'C','S','R','CR','SC','SR', or NA

Examples

```
mfg_csr_convert("11a-NakeChlor")
```

`mfg_csr_convert_df` *Returns a CSR classification based on Morphofunctional group (MFG). Correspondence based on Salmaso et al. 2015 and Reynolds et al. 1988*

Description

Returns a CSR classification based on Morphofunctional group (MFG). Correspondence based on Salmaso et al. 2015 and Reynolds et al. 1988

Usage

```
mfg_csr_convert_df(phyto.df, mfg)
```

Arguments

`phyto.df`	dataframe containing a character field containing MFG classifications
`mfg`	Character string with MFG name, following Salmaso et al. 2015

Value

A dataframe with an additional field named CSR, containing CSR classifications or NA

Examples

```
data(lakegeneva)
lakegeneva<-genus_species_extract(lakegeneva,'phyto_name')
lakegeneva<-species_to_mfg_df(lakegeneva)
lakegeneva<-mfg_csr_convert_df(lakegeneva,mfg='MFG')
head(lakegeneva)
```

`mfg_csr_library` *MFG-CSR correspondence based on CSR-trait relationships in Reynolds et al. 1988 and MFG-trait relationships in Salmaso et al. 2015*

Description

MFG-CSR correspondence based on CSR-trait relationships in Reynolds et al. 1988 and MFG-trait relationships in Salmaso et al. 2015

Usage

```
data(mfg_csr_library)
```

Format

A data frame with columns:

MFG full MFG name from Salmaso et al. 2015

CSR CSR classification including intermediate classes

`phyto_ts_aggregate` *Aggregate phytoplankton timeseries based on abundance. Up to 3 grouping variables can be given: e.g. genus, species, stationid, depth range. If no abundance var is given, will aggregate to presence/absence of grouping vars.*

Description

Aggregate phytoplankton timeseries based on abundance. Up to 3 grouping variables can be given: e.g. genus, species, stationid, depth range. If no abundance var is given, will aggregate to presence/absence of grouping vars.

Usage

```
phyto_ts_aggregate(
  phyto.data,
  DateVar = "date_dd_mm_yy",
  SummaryType = c("abundance", "presence.absence"),
  AbundanceVar = "biovol_um3_ml",
  GroupingVar1 = "phyto_name",
  GroupingVar2 = NA,
  GroupingVar3 = NA,
  remove.rare = FALSE,
  fun = sum,
  format = "%d-%m-%y"
)
```

Arguments

`phyto.data`	data.frame
`DateVar`	character string: field name for date variable. character or POSIX data.
`SummaryType`	'abundance' for a matrix of aggregated abundance,'presence.absence' for 1 (present) and 0 (absent).
`AbundanceVar`	character string with field name containing abundance data Can be NA if data is only a species list and aggregated presence/absence is desired.
`GroupingVar1`	character string: field name for first grouping variable. defaults to spp.
`GroupingVar2`	character string: name of additional grouping var field
`GroupingVar3`	character string: name of additional grouping var field

remove.rare	TRUE/FALSE. If TRUE, removes all instances of GroupingVar1 that occur < 5 of time periods.
fun	function used to aggregate abundance based on grouping variables
format	character string: format for DateVar POSIXct conversion

Value

a data.frame with grouping vars, date_dd_mm_yy, and abundance or presence/absence

Examples

```
data(lakegeneva)
lakegeneva<-genus_species_extract(lakegeneva,'phyto_name')
lg.genera=phyto_ts_aggregate(lakegeneva,SummaryType='presence.absence',
                             GroupingVar1='genus')
head(lg.genera)
```

---

sampeff *Visually assess change in sampling effort over time (author: Dietmar Straile)*

Description

Visually assess change in sampling effort over time (author: Dietmar Straile)

Usage

```
sampeff(
  b_data,
  column,
  save.pdf = FALSE,
  lakename = "",
  datecolumn = "date_dd_mm_yy",
  dateformat = "%d-%m-%y"
)
```

Arguments

b_data	Name of data.frame object
column	column name or number for field containing abundance (biomass,biovol, etc.) can be NA for presence absence
save.pdf	TRUE/FALSE Should the output plot be saved to a file? defaults to FALSE
lakename	Character string for labeling output plot
datecolumn	Character String or number specifying dataframe field with date information
dateformat	Character string specifying POSIX data format

Value

a time-series plot of minimum relative abundance over time. This should change systematically with counting effort.

Examples

```
data(lakegeneva)
#example dataset with 50 rows

sampeff(lakegeneva,column=6) #column 6 contains biovolume
```

`set_algaebase_apikey_header` *Add algaebase API key to curl handle*

Description

Add algaebase API key to curl handle

Usage

```
set_algaebase_apikey_header(apikey = NULL)
```

Arguments

`apikey`	character string with valid key

Value

curl handle object

`species_mfg_library` *Trait-based MFG classifications for common Eurasion/North American phytoplankton species. See accompanying manuscript for sources*

Description

Trait-based MFG classifications for common Eurasion/North American phytoplankton species. See accompanying manuscript for sources

Usage

```
data(species_mfg_library)
```

Format

A data frame with columns:

genus genus name

species species name

MFG corresponding MFG classification based on Salmaso et al. 2015

source literature or online source for MFG classification

References

Algaebase https://www.algaebase.org

Phycokey https://www.cfb.unh.edu/phycokey/phycokey.htm

Western Diatoms of North America https://diatoms.org

CyanoDB 2 http://www.cyanodb.cz/

Nordic Microalgae https://nordicmicroalgae.org

Phytopedia https://phytoplankton.eoas.ubc.ca/

Kapustin, D., Sterlyagova, I. and Patova, E., 2019. Morphology of Chrysastrella paradoxa stomatocysts from the Subpolar Urals (Russia) with comments on related morphotypes. Phytotaxa, 402(6), pp.295-300.

---

species_search_itis *Wrapper function for several functions in ritis:: Searches ITIS database for matches to a binomial scientific name outputs matches, current accepted names, synonyms, and higher taxonomy*

Description

Wrapper function for several functions in ritis:: Searches ITIS database for matches to a binomial scientific name outputs matches, current accepted names, synonyms, and higher taxonomy

Usage

```
species_search_itis(genspp, higher = FALSE)
```

Arguments

genspp	Character string. Binomial scientific name with space between genus and species.
higher	Boolean. If TRUE, add higher taxonomic classifications to output

Value

data.frame with submitted name (orig.name), matched name (matched.name), 1/0 flag indicating that original name is currently accepted (orig.name.accepted), 1/0 flag indicating if search was genus_only (for distinguishing genus_search_itis and species_search_itis results), synonyms if any, and higher taxonomy (if higher=TRUE)

Examples

```
species="Aphanizomenon flosaquae"
species_search_itis(species,higher=FALSE)
```

`species_to_mfg` *Conversion of a single genus and species name to a single MFG. Uses species.mfg.library*

Description

Conversion of a single genus and species name to a single MFG. Uses species.mfg.library

Usage

```
species_to_mfg(genus, species = "", flag = 1, mfgDbase = NA)
```

Arguments

`genus`	Character string: genus name
`species`	Character string: species name
`flag`	Resolve ambiguous mfg: 1 = return(NA),2= manual selection
`mfgDbase`	data.frame of species MFG classifications. Defaults to the supplied species.mfg.library data object

Value

a data frame with MFG classification and diagnostic information. ambiguous.mfg=1 if multiple possible mfg matches genus.classification=1 if no exact match was found with genus + species name partial.match=1 if mfg was based on fuzzy matching of taxonomic name.

Examples

```
species_to_mfg('Scenedesmus','bijuga')
#returns "11a-NakeChlor"
```

species_to_mfg_df — *Wrapper function to apply species_phyto_convert() across a data.frame*

Description

Wrapper function to apply species_phyto_convert() across a data.frame

Usage

```
species_to_mfg_df(phyto.df, flag = 1, mfgDbase = NA)
```

Arguments

phyto.df	Name of data.frame. Must have character fields named 'genus' and 'species'
flag	Resolve ambiguous MFG: 1 = return(NA), 2 = manual selection
mfgDbase	specify library of species to MFG associations.

Value

input data.frame with a new character column of MFG classifications and diagnostic information

Examples

```
data(lakegeneva)
#example dataset with 50 rows

new.lakegeneva <- genus_species_extract(lakegeneva,'phyto_name')
new.lakegeneva <- species_to_mfg_df(new.lakegeneva)
head(new.lakegeneva)
```

traitranges — *surface/volume ratio and max linear dimension criteria for CSR From Reynolds 1988 and Reynolds 2006*

Description

surface/volume ratio and max linear dimension criteria for CSR From Reynolds 1988 and Reynolds 2006

Usage

```
data(traitranges)
```

Format

A data frame with columns:

Measurement measurement type

C.min minimum value for C

S.min minimum value for S

R.min minimum value for R

C.max maximum value for C

S.max maximum value for S

R.max maximum value for R

units units of measurement

source source for criteria

`traits_to_csr` *Assign phytoplankton species to CSR functional groups, based on surface to volume ratio and maximum linear dimension ranges proposed by Reynolds et al. 1988;2006*

Description

Assign phytoplankton species to CSR functional groups, based on surface to volume ratio and maximum linear dimension ranges proposed by Reynolds et al. 1988;2006

Usage

```
traits_to_csr(
  sav,
  msv,
  msv.source = "Reynolds 2006",
  traitrange = algaeClassify::traitranges
)
```

Arguments

`sav`	numeric estimate of cell or colony surface area /volume ratio
`msv`	numeric product of surface area/volume ratio and maximum linear dimension
`msv.source`	character string with reference source for distinguishing criteria
`traitrange`	data frame with trait criteria for c,s,r groups. The included table can be replaced with user-defined criteria if desired. Measurements are: Surface area/volume ratio (sav), maximum linear dimension (mld) and mld*sav (msv).

Value

a character string with one of 5 return values: C,CR,S,R, or SR. CR and SR groups reflect overlap between criteria for the 3 main groups.

See Also

<https://powellcenter.usgs.gov/geisha> for project information

Examples

```
traits_to_csr(sav=0.2,msv=10,msv.source='Reynolds 2006',traitrange=traitranges)
```

`traits_to_csr_df` *Add CSR functional group classifications to a dataframe of phytoplankton species, based on surface to volume ratio and maximum linear dimension ranges proposed by Reynolds et al. 1988;2006*

Description

Add CSR functional group classifications to a dataframe of phytoplankton species, based on surface to volume ratio and maximum linear dimension ranges proposed by Reynolds et al. 1988;2006

Usage

```
traits_to_csr_df(
  df,
  sav,
  msv,
  msv.source = "Reynolds 2006",
  traitrange = algaeClassify::traitranges
)
```

Arguments

`df`	name of dataframe
`sav`	character string with name of column that contains surface to volume ratio values
`msv`	character string with name of column that contains maximum linear dimension * surface to volume ratio values
`msv.source`	character string with reference source for distinguishing criteria
`traitrange`	data frame with trait criteria for c,s,r groups. The included table can be replaced with user-defined criteria if desired. Measurements are: Surface area/volume ratio (sav), maximum linear dimension (mld) and mld*sav (msv).

Value

a character string with one of 5 return values: C,CR,S,SR, or R

Examples

```
csr.df<-data.frame(msv=10,sav=1)

csr.df$CSR<-traits_to_csr_df(csr.df,'msv','sav')

print(csr.df)
```

`traits_to_mfg` *Assign MFG based on binary functional traits and taxonomy (Class and Order)*

Description

Assign MFG based on binary functional traits and taxonomy (Class and Order)

Usage

```
traits_to_mfg(
  flagella = NA,
  size = NA,
  colonial = NA,
  filament = NA,
  centric = NA,
  gelatinous = NA,
  aerotopes = NA,
  class = NA,
  order = NA
)
```

Arguments

`flagella`	1 if flagella are present, 0 if they are absent.
`size`	Character string: 'large' or 'small'. Classification criteria is left to the user.
`colonial`	1 if typically colonial growth form, 0 if typically unicellular.
`filament`	1 if dominant growth form is filamentous, 0 if not.
`centric`	1 if diatom with centric growth form, 0 if not. NA for non-diatoms.
`gelatinous`	1 mucilagenous sheath is typically present, 0 if not.
`aerotopes`	1 if aerotopes allowing buoyancy regulation are typically present, 0 if not.
`class`	Character string: The taxonomic class of the species
`order`	Character string: The taxonomic order of the species

Value

A character string of the species' morphofunctional group

Examples

```
traits_to_mfg(flagella = 1,size = "large",colonial = 1,filament = 0,centric = NA,gelatinous = 0,
              aerotopes = 0,class = "Euglenophyceae",order = "Euglenales")
```

`traits_to_mfg_df` *Assign morphofunctional groups to a dataframe of functional traits and higher taxonomy*

Description

Assign morphofunctional groups to a dataframe of functional traits and higher taxonomy

Usage

```
traits_to_mfg_df(
  dframe,
 arg.names = c("flagella", "size", "colonial", "filament", "centric", "gelatinous",
    "aerotopes", "class", "order")
)
```

Arguments

`dframe`	An R dataframe containing functional trait information and higher taxonomy
`arg.names`	Character string of column names corresponding to arguments for traits_to_mfg()

Value

A character vector containing morpho-functional group (MFG) designations

Examples

```
#create a two-row example dataframe of functional traits
func.dframe=data.frame(flagella=1,size=c("large","small"),colonial=0,filament=0,centric=NA,
                     gelatinous=0,aerotopes=0,class="Euglenophyceae",order="Euglenales",
                       stringsAsFactors=FALSE)

#check the dataframe
print(func.dframe)

#run the function to produce a two-element character vector
func.dframe$MFG<-traits_to_mfg_df(func.dframe,c("flagella","size","colonial",
                                  "filament","centric","gelatinous",
                                  "aerotopes","class","order"))

print(func.dframe)
```

Index