Package ‘vegtable’

January 13, 2020

Version 0.1.5
Encoding UTF-8
Title Handling Vegetation Data Sets
Depends R(>= 3.0.0), taxlist
Imports foreign, methods, plotKML, qdapRegex, sp, stringi, vegdata
Suggests devtools, vegan
LazyData true
Description Import and handling data from vegetation-plot databases, especially
data stored in 'Turboveg' (<https://www.synbiosys.alterra.nl/turboveg>). Also
import/export routines for exchange of data with 'Juice'
(<http://www.sci.muni.cz/botany/juice>) are implemented.
License GPL (>= 2)
URL https://github.com/kamapu/vegtable
BugReports https://github.com/kamapu/vegtable/issues
Collate 'NULLing.R'
'coverconvert-class.R"vegtable-class.R"shaker-class.R'
'vegtable_stat.R"df2vegtable.R"used_synonyms.R"subset.R'
'coverconvert-methods.R"names.R'
'tv2coverconvert.R"tv2vegtable.R'
'crosstable.R"aggregate.R"write_juice.R"read_juice.R"vegetable2kml.R'
'layers2samples.R"shaker-methods.R"make_cocktail.R'
'summary.R"match_names.R"taxa2samples.R"count_taxa.R'
'trait_proportion.R"trait_stats.R"StartMessage.R'

NeedsCompilation no
Author Miguel Alvarez [aut, cre] (<https://orcid.org/0000-0003-1500-1834>)
Maintainer Miguel Alvarez <kamapu78@gmail.com>
Repository CRAN
Date/Publication 2020-01-13 19:30:02 UTC
R topics documented:

add_releves .............................................. 2
aggregate .................................................. 4
as.list ..................................................... 5
aspect_conv-data ........................................ 6
braun_blanquet-data ................................. 6
clean ......................................................... 7
count_taxa ................................................ 8
coverconvert-class ...................................... 9
crosstable,cross2db ........................................ 10
df2vetable .................................................. 11
dune_egg-data ............................................ 13
Extract ....................................................... 13
header ......................................................... 14
Kenya_egg-data ........................................... 15
layers2samples ............................................ 16
make_cocktail,set_pseudo,set_group,set_formula .......... 17
match_names .............................................. 19
merge_taxa,taxa2samples ................................ 20
names,dimnames .......................................... 21
read_juice .................................................. 22
shaker-class ............................................. 24
subset ....................................................... 25
summary .................................................... 26
trait_stats, trait_proportion .......................... 27
transform .................................................. 29
tv2coverconvert .......................................... 30
tv2vegtable .............................................. 31
used_synonyms ........................................... 32
vetable-class ............................................ 34
vetable2kml ............................................... 35
vetable_stat .............................................. 36
vet Relation,relation2header .......................... 36
Wetlands-data ............................................ 38
write_juice ................................................ 38

Index ...................................................... 40

add_releves Merge relevés from data frames into vetable objects.

Description
Addition of plot observations into existing data sets may implicate merging data frames with vetable objects.
Usage

```r
## S4 method for signature 'vegtable, data.frame'
add_releves(vegtable, releves, header, abundance, split_string, usage_ids=FALSE, layers=FALSE, layers_var, format="crosstable", ...)
```

Arguments

- `vegtable`: An object of class `vegtable`.
- `releves`: A data frame including plot observations to be added into 'vegtable'.
- `header`: A data frame (optional) including header information for plots.
- `abundance`: A character value (or vector of length 2) indicating the names of abundance variable in 'vegtable'.
- `split_string`: Character value used to split mixed abundance codes.
- `usage_ids`: Logical value indicating whether species are as taxon usage ids (integers) or names in 'releves'.
- `layers`: Logical value indicating whether layers are included in 'releves' or not.
- `layers_var`: Name of the layer variable in 'vegtable'.
- `format`: Character value indicating input format of 'releves' (whether 'crosstable' or 'databaselist').
- `...`: Further arguments passed to function `cross2db` (i.e. 'na_strings').

Details

Since this function will only update slots 'samples' and 'header', consistency with slots 'layers', 'relations' and 'species' have to be checked and accordingly updated in advance.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

See Also

- `cross2db`

Examples

```r
## No example at the moment
```
Description

This function aggregates information contained in `vegtable` objects to a summarizing data frame.

Usage

```r
## S4 method for signature 'formula'
aggregate(x, data, FUN, use_nas=TRUE, ...)
```

Arguments

- `x` A formula indicating the variables used for the summary.
- `data` Either a data frame or an object of class `vegtable`.
- `FUN` Function used to aggregate values.
- `use_nas` Logical value indicating whether NA's should be included in categorical variables or not.
- `...` Further arguments passed to the function `aggregate`.

Details

This function works in a similar way as `crosstable`.

Value

An object of class `data.frame`.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

See Also

`aggregate`.

Examples

```r
## Example follows
```
as.list

Coerce an S4 object to a list.

Description
Coercion used to explore content in S4 objects.

Usage

## S4 method for signature 'vegetable'
as.list(x, ...)

## S4 method for signature 'coverconvert'
as.list(x, ...)

Arguments

x an object of class coverconvert or vegetable.

... further arguments passed from or to other methods.

Details
S4 objects will be coerced to lists, where each slot in the input object becomes a member of the output list. This way allows to explore content and solve problems when validity checks fail.

Value
An object of class list.

Author(s)
Miguel Alvarez (<kamapu78@gmail.com>).

Examples

## Importing 'Easplist' from taxlist
library(taxlist)
data(Easplist)

## Head of slot 'taxonNames'
class(Easplist)
head(Easplist@taxonNames)

## The same after coercing to list
Easplist <- as.list(Easplist)
class(Easplist)
head(Easplist$taxonNames)
Conversion of Aspect Classes to Azimuth

Description
Conversion table required to transform values of aspect to azimuth in degrees.

Usage
data(aspect_conv)

Format
A numeric vector of values in degrees for the symbols used as names.

Author(s)
Miguel Alvarez, <kamapu78@gmail.com>.

Examples
library(vegtable)
data(aspect_conv)

aspect_conv[c("N", "S", "ENE", "SSW")]

Conversion of Braun-Blanquet codes to cover percentage.

Description
Cover values conversion as coverconvert object.

Usage
data(braun_blanquet)

Format
An object of class coverconvert.

Details
Object of class coverconvert contains conversion tables usually from a categorical variable (a cover scale) to a numerical one (equivalent percentage cover value). Cover values are stored as range for each level in the scale (minimum and maximum cover value).
**clean**

See Also

*coverconvert, transform.*

Examples

```r
library(vegtable)
data(braun_blanquet)

## Quick displays
names(braun_blanquet)
summary(braun_blanquet)
summary(braun_blanquet$b_bbds)
```

---

**clean**  
*Clean orphaned records in vegtable object.*

Description

Delete entries in slots header and species orphaned by manipulation of slots.

Usage

```r
## S4 method for signature 'vegtable'
clean(object, times=2, ...)
```

Arguments

- `object`  
  A `vegtable` object.
- `times`  
  Numeric value indicating how many times should be the cleaning be repeated.
- `...`  
  Further arguments passed from or to other methods.

Details

Orphaned records generated by modifications in some slots may cause a loss on the validity of `vegtable` objects. This function should be applied to optimise the allocated size of a `vegtable` object, as well. Since running cleaning only once does not assure the deletion of all orphaned entries, it is recommended to run it at least twice. This repetition of cleaning is controlled by the argument 'times'.

Value

A clean `vegtable` object.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).
Examples

library(vegtable)
data(Kenya_veg)

## Direct manipulation of slot header generates an invalid object
Kenya_veg@header <- Kenya_veg@header[1:50,]
summary(Kenya_veg)

## Now apply cleaning
Kenya_veg <- clean(Kenya_veg)
summary(Kenya_veg)

---

count_taxa

Count taxa within a taxlist object

Description

Counting number of taxa within taxlist objects or character vectors containing taxon names.

Usage

## S4 method for signature 'vegtable'
count_taxa(object, level, include_lower=FALSE, ...)

## S4 method for signature 'formula'
count_taxa(object, data, include_lower=FALSE, suffix="_count", in_header=FALSE, ...)

Arguments

- **object**: An object of class `vegtable` or a formula.
- **data**: An object of class `vegtable`.
- **level**: Character value indicating the taxonomic rank of counted taxa.
- **include_lower**: Logical value, whether lower taxonomic ranks should be included at the requested level.
- **suffix**: Character value used as suffix on the calculated variable.
- **in_header**: Logical value, whether the result should be included in the slot header of the input `vegtable` object or not. A warning message is provided if the calculation is not done for every plot observation.
- **...**: further arguments passed among methods.
Details

This function provides a quick calculation of taxa in `vegtable` objects, considering only records in slot samples. Such records can be also merged from lower ranks.

For the formula method, units without any requested taxa will not appear in the output data frame. If no taxa at all is occurring at the requested level in any unit, an error message will be retrieved.

Value

An data frame with the number of taxa from requested level at requested units for the formula method, or just an integer value.

Author(s)

Miguel Alvarez, <kamapu78@gmail.com>.

Examples

```r
library(vegtable)

## Different alternatives
count_taxa(Kenya_veg)
head(count_taxa(~ ReleveID, Kenya_veg))
head(count_taxa(species ~ ReleveID, Kenya_veg))
head(count_taxa(species ~ ReleveID, Kenya_veg, TRUE))
head(count_taxa(family ~ ReleveID, Kenya_veg, TRUE))
```

coverconvert-class  Cover conversion tables.

Description

Cover conversion tables for `vegtable` objects.

Details

This class implements conversions from different cover scales in percentage cover. For transformations to percentage cover, the function `transform` should be than used.

Slots

- **value**: List containing the levels of each scale.
- **conversion**: List with the respective start and end cut levels for the scale levels.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).
See Also
tv2coverconvert, braun_blanquet.

Examples

library(vegtable)
showClass("coverconvert")

## Add a custom scale
Scale <- new("coverconvert")
Scale$my_scale <- list(
  value=factor(c("low","medium","high"), levels=c("low","medium","high")),
  conversion=c(0,50,75,100))
summary(Scale)

crosstable, cross2db   Generating cross tables from database lists.

description

This function is generating cross tables, which are the most common format used by statistical packages analysing vegetation data (e.g. vegan).

Usage

## S4 method for signature 'formula, data.frame'
crosstable(formula, data, FUN, na_to_zero=FALSE, use_nas=TRUE, as_matrix=FALSE, ...)

## S4 method for signature 'formula, vegtable'
crosstable(formula, data, FUN, na_to_zero=FALSE, use_nas=TRUE, ...)

cross2db(object, layers=FALSE, na_strings)

Arguments

formula A formula indicating the variables used in the cross table.
data Either a data frame or an object of class vegtable.
FUN Function used to aggregate values.
na_to_zero A logical value indicating whether zeros should be inserted into empty cells or not.
use_nas Logical value indicating whether NAs should be considered as levels for categorical variables or not.
as_matrix A logical value, whether output should be done as matrix or data frame.
...
Further arguments passed to the function aggregate.
**df2vegtable**

A data frame including a cross table.

**layers**

Logical value, whether the cross table includes a layer column or not.

**na_strings**

Character vector indicating no records in the cross table.

**Details**

Most applications and displays of vegetation data use preferentially the cross table format. For convenience, the formula has the form `abundance ~ plot + species + ...`. Additional variables used for rows (... ) can be for instance the layers.

For objects of class `vegtable`, the formula can also include variables from the species list (for example `AcceptedName`, `AuthorName`) or even taxon traits.

**Value**

An object of class `data.frame`.

**Author(s)**

Miguel Alvarez (<kamapu78@gmail.com>).

**Examples**

```r
library(vegtable)

## load Kenya_veg and subset to reference 2331 (Bronner 1990)
data(Kenya_veg)
Kenya_veg <- subset(Kenya_veg, REFERENCE == 2331, slot="header")

## transform cover to percentage
Kenya_veg <- transform(Kenya_veg, to="cover_perc", rule="middle")

## cross table of the first 5 plots
Cross <- crosstable(cover_perc ~ ReleveID + AcceptedName + AuthorName, 
                   Kenya_veg[1:5,], mean, na_to_zero=TRUE)
head(Cross)
```

---

**df2vegtable**

*Convert a data frame into a vegtable object.*

**Description**

Conversion of a data frame containing a crosstable of abundance or cover of species in single plots.

**Usage**

`df2vegtable(x, species, layer, ...)`
Arguments

x  A data frame formatted for a taxlist object.
species  Numeric or integer indicating the position of the column with species names.
layer  Numeric or integer indicating the position of the column with layers.
...  Further arguments passed from or to other methods.

Details

This function coerces a data frame containing a vegetation cross table into a `vegtable` object. The input data frame 'x' may include information on the layers or not.

Value

A `vegtable` object.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

Examples

```r
## Creating data set 'dune_veg'
library(vegtable)
library(vegan)

## Load data from vegan
data(dune)
data(dune.env)

## Conversion to vegtable
dune_veg <- data.frame(species=colnames(dune), t(dune), stringsAsFactors=FALSE,
                        check.names=FALSE)
dune_veg <- df2vegtable(dune_veg, species=1)

summary(dune_veg)

## Adding environmental variables
dune.env$ReleveID <- as.integer(rownames(dune.env))
header(dune_veg) <- dune.env

summary(dune_veg)
```
**dune_veg-data**

---

**Dutch Dune Meadows.**

**Description**

Data set from the package vegan, converted to a vegtable object.

**Usage**

```r
data(dune_veg)
```

**Format**

An object of class vegtable.

**Source**

Original data were imported from dune.

**References**


**Examples**

```r
library(vegtable)
data(dune_veg)
summary(dune_veg)
```

---

**Extract**

*Select or replace elements in objects.*

**Description**

Methods for quick access to slot 'header' of vegtable objects or for access to single cover scales in coverconvert objects. Also replacement methods are implemented.

**Usage**

```r
## S4 method for signature 'vegtable'
x$name

## S4 method for signature 'vegtable'
x[i, j, ..., drop=FALSE]

## S4 method for signature 'coverconvert'
x$name
```
`header` Retrieve or replace slot `header` in 'vegtable' objects.

**Arguments**

- `x` Object of class `vegtable`.
- `...` Further arguments passed to or from other methods.
- `name` A name to access.
- `i,j` Indices for access.
- `drop` A logical value passed to `Extract`.

**Examples**

```r
library(vegtable)
data(Kenya_veg)

## Range of latitude values in database
range(Kenya_veg$LATITUDE)

## Summary of countries
summary(Kenya_veg$COUNTRY)
summary(droplevels(Kenya_veg$COUNTRY))

## First 5 samples
summary(Kenya_veg[1:5,])
```

**Description**

Retrieve or replace the content of slot `header` in `vegtable` objects.

**Usage**

```r
## S4 method for signature 'vegtable'
header(x, ...)

header(x) <- value
```

**Arguments**

- `x` Object of class `vegtable`.
- `value` Data frame to be set as slot header.
- `...` Further arguments passed to or from other methods.

**Author(s)**

Miguel Alvarez (<kamapu78@gmail.com>).
**Author(s)**

Miguel Alvarez (<kamapu78@gmail.com>).

**Examples**

```r
library(vegtable)
data(Kenya_veg)
head(header(Kenya_veg))
```

---

**Kenya_veg-data**

**Vegetation-Plots from Kenya.**

**Description**

A subset of SWEA-Dataveg including five references providing plots collected in Kenya.

**Usage**

```r
data(Kenya_veg)
```

**Format**

An object of class `vegtable`.

**Author(s)**

Miguel Alvarez (<kamapu78@gmail.com>), Michael Curran (<currmi01@gmail.com>).

**Source**

[http://www.givd.info/ID/AF-00-006](http://www.givd.info/ID/AF-00-006).

**References**

Examples

```
library(vegtable)
data(Kenya_veg)
summary(Kenya_veg)
```

```
layers2samples

Add information from slot 'layers' into slot 'samples'.
```

Description

Slot layers may include additional information that should be moved to samples in order to use it by `subset`, `aggregate` or `crosstable` methods.

Usage

```
## S4 method for signature 'vegtable,character,character'
layers2samples(object, layer, variable, ...)

## S4 method for signature 'vegtable,character,missing'
layers2samples(object, layer, variable, ...)
```

Arguments

- `object` An object of class `vegtable`.
- `layer` Character value indicating a target layer.
- `variable` Character vector with the names of variables to be inserted in slot 'samples'.
- `...` Further arguments to be passed among methods.

Details

If names of variables are not provided, all variables from the respective layer table will be inserted in slot 'samples'.

Value

An object of class `vegtable` with variables added to samples.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

Examples

```
## No example available for this function.
```
Produce a Cocktail classification.

Description

Classification of vegtable objects according to Cocktail algorithms.

Usage

```r
## S4 method for signature 'shaker,vegtable'
make_cocktail(shaker, vegtable, which, cover, syntax="Syntax", FUN=sum, ...)
```

```r
## S4 method for signature 'shaker,taxlist,character'
set_pseudo(shaker, companion, pseudo, pseudo_id, authority=FALSE,
enc_cont="latin1", enc_gr="utf8", ...)
```

```r
## S4 method for signature 'shaker,vegtable,character'
set_pseudo(shaker, companion, pseudo, ...)
```

```r
## S4 method for signature 'shaker,taxlist,character'
set_group(shaker, companion, group, group_id, authority=FALSE, enc_cont="latin1",
enc_gr="utf8", ...)
```

```r
## S4 method for signature 'shaker,vegtable,character'
set_group(shaker, companion, group, ...)
```

```r
## S4 method for signature 'shaker,taxlist,character'
set_formula(shaker, companion, formula, formula_id, authority=FALSE,
enc_cont="latin1", enc_gr="utf8", ...)
```

```r
## S4 method for signature 'shaker,vegtable,character'
set_formula(shaker, companion, formula, ...)
```

Arguments

- **shaker**: An object of class `shaker` containing the respective cocktail definitions.
- **vegtable**: An object of class `vegtable` containing the vegetation observations to be classified.
- **which**: Integer or character indicating the definition to be applied for classification.
- **cover**: Name of the cover variable in `vegtable`.
- **syntax**: Character value indicating the name of the retrieved variable including the final classification of plots.
- **FUN**: Function used for merging multiple occurrence of species in a single plot.
- **companion**: Either a `taxlist` or a `vegtable` object.
pseudo.group  Character vector with names of taxa included in a pseudo-species or a species group.
formula        Character vector including a formula as definition of a vegetation unit.
pseudo_id,group_id,formula_id  Character value as name of the pseudo-species, species group or defined vegetation unit.
authority      Logical value indicating whether author names should be included in the taxon name or not.
enc_cont,enc_gr Encodings used for special characters.
...            Further arguments passes from or to other methods.

Details

Cocktail algorithms are logical functions selecting plots according to either occurrence of species groups and cover values of single species. A group will be declared as occurring in a plot when at least a half of its members is present in the plot.

This function inserts single columns with logical values indicating whether a plot is classified in the vegetation unit or not. An additional column (name provided in argument ‘syntax’) compile all vegetation units, indicating with a ‘+’ symbol those plots classified in more than one vegetation unit. When only a part of the formulas will be used, it should be specified by the argument `which`.

These functions are implemented for constructing or complementing `shaker` objects. Note that construction of those objects will always require a `companion` object, which is either an object of class `taxlist` or `vegtable`.

Value

A data frame corresponding to the slot ‘header’ of input object ‘vegtable’, including the results of Cocktail classification for the respective plots.
A `shaker` object.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>.

References


See Also

`shaker`, `vegtable`, `Wetlands`.
**Examples**

```r
library(vegtable)

## Example from Alvarez (2017)
data(Wetlands)

Wetlands_veg@header <- make_cocktail(Wetlands, Wetlands_veg, cover="percen")
summary(as.factor(Wetlands_veg@header$Syntax))

## Same but only for two vegetation units
data(Wetlands)
Wetlands_veg@header <- make_cocktail(Wetlands, Wetlands_veg, which=c("HY1","HY2"), cover="percen")
summary(as.factor(Wetlands_veg@header$Syntax))

## Construct the 'shaker' object anew
Wetlands <- new("shaker")

## Set a pseudo-species
Wetlands <- set_pseudo(Wetlands, Wetlands_veg, c("Cyperus latifolius", "Cyperus exaltatus"))

## Set a species group
Wetlands <- set_group(Wetlands, Wetlands_veg, group_id="Cyperus papyrus", group=c(
  "Cyperus papyrus",
  "Cyclosorus interruptus",
  "Lepistemon owariense"))

## Set a formula
Wetlands <- set_formula(Wetlands, Wetlands_veg, formula_id="HE1", formula="groups:'Cyperus papyrus' | species:'Cyperus papyrus > 50'")

## Summaries
summary(Wetlands)
summary(Wetlands, Wetlands_veg)
```

---

**match_names**

*Search matchings between character and taxlist objects.*

**Description**

Names provided in a character vector will be compared with names stored in slot 'taxonNames' of an object of class `taxlist` by using the function `stringsim`.

**Usage**

```r
## S4 method for signature 'character,vegtable'
match_names(x, object, ...)
```
Arguments

- **x**: A character vector with names to be compared.
- **object**: An object of class `taxlist` to be compared with.
- **...**: Further arguments passed to `match_names`.

Details

This method is applied to the slot `species` in the input `vegetable` object.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

See Also

- `match_names`, `stringsim`.

Examples

```r
## No example at the moment
```

merge_taxa, taxa2samples

*Merge concepts.*

Description

Merge taxon concepts form into single ones or insert accepted names to slot samples.

Usage

```r
## S4 method for signature 'vegetable,numeric,missing'
merge_taxa(object, concepts, level, ...)

## S4 method for signature 'vegetable,missing,character'
merge_taxa(object, concepts, level, ...)

## S4 method for signature 'vegetable'
taxa2samples(object, merge_to, add_traits, ...)
```

Arguments

- **object**: Object of class `vegetable`.
- **concepts**: Numeric (integer) vector including taxon concepts to be merged.
- **level, merge_to**: Character value indicating the level to which the taxa have to be merged.
- **add_traits**: A character vector indicating variables in the slot `taxonTraits` to be added in slot `samples`.
- **...**: Further arguments passed to `merge_taxa` (`taxlist` method).
Details

This method is applied to a function defined in the package `taxlist` and only modify the slot 'species' in the input 'object'.

The use of 'taxa2samples' with 'merge_to' argument will produce a similar result as using 'merge_taxa' with 'level' argument, but 'taxa2samples' will replace the records in slot samples by the respective accepted names without any modification in slot species. Additionally taxon concept IDs will be addes as columns in samples and taxon traits if indicated in argument add_traits.

Value

An object of class `vegtable`.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

Examples

```r
library(vegtable)
data(Kenya_veg)

## Merge Olea capensis into one
summary(subset(Kenya_veg@species, grepl("Olea capensis", TaxonName),
  slot="names"), "all")
Kenya_veg <- merge_taxa(Kenya_veg, c(52041,50432,50235))

## Check Olea capensis again
summary(subset(Kenya_veg@species, grepl("Olea capensis", TaxonName),
  slot="names"), "all")

## Effect of taxa2samples by counting taxa
count_taxa(Kenya_veg, level="genus")
Kenya_veg <- taxa2samples(Kenya_veg, merge_to="genus")
count_taxa(Kenya_veg, level="genus")
```

---

**names,dimnames**

Retrieve names of `vegtable` and `coverconvert` objects.

**Description**

Quick access to column names in slot header and names of conversion codes.
Usage

```r
## S4 method for signature 'coverconvert'
names(x)

## S4 method for signature 'vegtable'
names(x)

## S4 method for signature 'vegtable'
dimnames(x)
```

Arguments

- `x` An object of class `vegtable` or `coverconvert`.

Details

These methods provide a quick display of the contents in `coverconvert` and `vegtable` objects.

Value

Either a vector or a list (in the case of `dimnames`) with the names of variables.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

Examples

```r
library(vegtable)
data(Kenya_veg)

## All possibilities shown in 'Usage'
names(Kenya_veg@coverconvert)
names(Kenya_veg)
dimnames(Kenya_veg)
```

Description

This function imports vegetation tables exported from `Juice` ([http://www.sci.muni.cz/botany/](http://www.sci.muni.cz/botany/juice)).

Usage

```r
read_juice(file, encoding="LATIN-1", sep=";", na="", ...)```
**read_juice**

**Arguments**

- **file**  
  Character value indicating the name of the file exported from 'Juice'.

- **encoding**  
  Argument passed to `readLines`.

- **sep**  
  Separator used to split rows into columns.

- **na**  
  Character used as not available values.

- **...**  
  Further arguments passed to `readLines`.

**Details**

For a properly import, you may strictly follow the export steps in 'Juice':

- Menu File -> Export -> Table -> to Spreadsheet Format File
- Check the option Export covers in %

In the header (see **Value**), the first column (juice_nr) corresponds to the plot number assigned by 'Juice' at import, while the column db_nr is the number originally assigned to the plot (e.g. 'Turboveg' ID).

**Value**

A list with two elements:

- **cross_table** A data frame of species by plot.
- **header** A data frame with header data.

**Author(s)**

Miguel Alvarez (<kamapu78@gmail.com>).

**Examples**

```r
library(vegtable)

## Installed 'Juice' version of 'Wetlands_veg'
Veg <- file.path(path.package("vegtable"), "juice", "Wetlands_juice.txt")
Veg <- read_juice(Veg)
names(Veg)
```
shaker-class  

Class containing Cocktail algorithms.

Description

Objects used for collecting Cocktail definitions.

Details

These objects work as expert systems for recognition of defined vegetation units among plots of a vegtable object. A 'shaker' object will be always dependent on a vegtable object, which is called 'companion'. Since modifications in the companion may affect the functionality of the 'shaker' object, it will be recommended to create the last during a session by a source script instead of recycling them from old R images.

Slots

- **pseudos**  List containing IDs of taxa that will be merged into pseudo-species.
- **groups**  List containing IDs of taxa belonging to the same Cocktail group.
- **dominants**  data frame including lists of species used as dominant species in Cocktail algorithms, as well as operators and cover values used in the formulas.
- **formulas**  List with formulas that will be used as definitions for vegetation units.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

See Also

- set_pseudo, set_group, set_formula, make_cocktail.

Examples

```r
library(vegtable)
showClass("shaker")
```
subset functions for vegtable objects.

Description

Produce subsets of vegtable objects.

Usage

```r
## S4 method for signature 'vegtable'
subset(x, subset, slot="header", keep_children=FALSE, keep_parents=FALSE, ...)
```

Arguments

- `x` A vegtable object for subset.
- `subset` Logical vector or operation for subset.
- `slot` Slot to be applied for subset.
- `keep_children` Argument passed to subset.
- `keep_parents` Argument passed to subset.
- `...` Further arguments passed from or to other methods.

Details

This function generate subsets of vegtable objects through logical operations. Such operations can be applied either to the plots, or the relations, which are the main slots in that class.

This method can be referred to the slot species the same way as `subset`, then the rest of the data will include only references to the subset of species list.

Value

A S4 object of class vegtable.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

Examples

```r
library(vegtable)
data(dune_veg)
summary(dune_veg)

## Select plots used as pastures
Pastures <- subset(dune_veg, Use == "Pasture", slot="header")
summary(Pastures)
```
Description

Display summaries for `vegtable` objects.

Usage

## S4 method for signature 'vegtable'
summary(object, units="Kb", ...)

## S4 method for signature 'coverconvert'
summary(object, ...)

## S4 method for signature 'shaker'
summary(object, companion, authority=FALSE, ...)

Arguments

- `object`: Object to be summarized.
- `units`: Units used for object size (passed to `format`).
- `companion`: Companion object (either a `taxlist` or a `vegtable` object).
- `authority`: Logical value indicating whether authors should be displayed or not.
- `...`: further arguments to be passed to or from other methods.

Details

Those methods are implemented for objects of the classes `vegtable`, `coverconvert` and `shaker`. The method for class 'vegtable' retrieves the metadata, the size of the object, its validity and additional statistics on the content of input object.

For objects of class `shaker`, the function 'summary' will either retrieve general statistics when 'companion' is missing, or a more detailed display when accompanied by a `taxlist` or `vegtable` object.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

Examples

library(vegtable)

## Summary for 'vegtable' objects
data(Wetlands)
summary(Wetlands_veg)
## Summary for 'coverconvert' objects
```r
data(braun_blanquet)
summary(braun_blanquet)
```

## Summary for 'shaker' objects (alone and with companion)
```r
summary(Wetlands)
sample(Wetlands, Wetlands_veg)
```

---

### Description

Calculation of statistics and proportions of taxon traits for plot observations or groups of observations, considering data relationships, taxonomic ranks and the handling of not available values.

The function `trait_stats()` calculates statistics for numeric variables, while the function `trait_proportion()` may be used for categorical variables. In the first case, a column with the name of the variable and a suffix will be generated, while in the second case, one additional column per selected trait level will be calculated.

Both mentioned functions offer the alternative weighted and unweighted calculations (e.g. calculations weighted by the abundance of species). In the particular case of `trait_stats()`, customized functions have to be defined as `foo(x,w,...)`, where `w` is the weight.

With the arguments 'taxon_level' and 'merge_to' the used taxonomic ranks can be defined, where the first one indicates which ranks have to be considered in the calculations and the second one determine the aggregation of taxa from a lower level to a parental one.

Formula methods allow for the calculation of multiple variables at once. The formulas have to be written as `trait_1 + ... + trait_n ~ head_var`.

### Usage

```r
## S4 method for signature 'character,vegtable'
trait_stats(trait, object, FUN, head_var, taxon_level, merge_to, weight, suffix="_stats", in_header=FALSE, ...)

## S4 method for signature 'formula,vegetable'
trait_stats(trait, object, weight, suffix="_stats", in_header=FALSE, ...)

## S4 method for signature 'character,vegetable'
trait_proportion(trait, object, head_var, trait_level, taxon_level, merge_to, include_nas=TRUE, weight, suffix="_prop", in_header=FALSE, ...)

## S4 method for signature 'formula,vegetable'
trait_proportion(trait, object, in_header=FALSE, ...)
```
trait_stats, trait_proportion

Arguments

trait
Either a character value indicating the name of trait variable or a formula including both arguments, 'trait' and 'head_var'.

object
A vegtable object.

FUN
A function usually defined as 'foo(x,...)' or as 'foo(x,w,...)' for weighted statistics.

head_var
Character value, the name of the variable at slot header to be used as aggregation level for the calculation of statistics or proportions. If not provided, the function will use ReleveID by default.

trait_level
Character vector indicating a selection of levels from a trait, in the case that some levels should be ignored in the output. Trait levels that are skipped at output will be still used for the calculation of proportions. This argument gets only applied for the character method.

taxon_level
Character value indicating a selected taxonomic rank for the output.

merge_to
Character value indicating the taxonomic rank for aggregation of taxa. All ranks lower than the one indicated here will be assigned to the respective parents at the required taxonomic rank.

include_nas
Logical value indicating whether NAs should be considered for the calculation of proportions or not.

weight
Character value indicating the name of the variable at slot samples used as weight for the proportions. Usually the numeric abundance.

suffix
A suffix added to the name of the trait variable or to the levels of categorical trait variables. It is meant to avoid homonymous variables within the same object.

in_header
Logical value indicating whether the output should be inserted in the slot header or provided as data frame.

...
Further arguments passed among methods. In the case of the formula method, arguments are passed to the character method.

Value
A data frame with the proportions of traits levels or statistics for the trait variable, or an object of class vegtable including those results at the slot header.

Author(s)
Miguel Alvarez, <kamapu78@gmail.com>.

Examples

## Cocktail classification of plots
data(Wetlands)
Wetlands_veg@header <- make_cocktail(Wetlands, Wetlands_veg, cover="percen")

## Calculation of proportion of Cyperaceae species in the plot
Wetlands_veg <- trait_proportion("FAMILY", Wetlands_veg, trait_level="Cyperaceae", weight="percen", include_nas=FALSE, in_header=TRUE)
```r
## Display of proportions per plant community
boxplot(Cyperaceae_prop ~ Syntax, Wetlands_veg@header, col="grey")
```

## Description
Convert cover scales to percent cover.

## Usage

### S4 method for signature 'character,coverconvert'
```r
transform(x, conversion, from=NULL, rule="top", zeroto=0.1, ...)
```

### S4 method for signature 'factor,coverconvert'
```r
transform(x, conversion, ...)
```

### S4 method for signature 'vegtable,missing'
```r
transform(x, to, replace=FALSE, rule="top", zeroto=0.1, ...)
```

## Arguments
- **x**
  - Either a factor or character vector, or a `vegtable` object.
- **conversion**
  - An object of class `coverconvert`.
- **from**
  - Scale name of values in 'x' as character value.
- **to**
  - Name of the column in slot 'samples' for writing converted values.
- **replace**
  - Logical value indicating whether existing cover values should be replaced or not.
- **rule**
  - Rule applied for the conversion (see details).
- **zeroto**
  - Value used to replace levels with bottom at 0% cover.
- **...**
  - Further arguments passed from or to other methods.

## Details
This function requires as input a `coverconvert` object which contains the conversion tables. In the case of `vegtable` objects, the conversion is already embedded in the slot 'coverconvert'.

Three rules are implemented for transformation, either 'top' (values transformed to the top of the range), 'middle' (transformation at the midpoint), and 'bottom' (conversion at the lowest value of the range). In the later case, transformation ranges starting at 0% of cover can be set to a different value by the argument 'zeroto'.

When 'replace=FALSE', existing values of cover in the `vegtable` object will be maintained. Since there is not a standard naming of cover values, in the transformation the name of cover variable should be indicated in the argument 'to'.
Value
Either a vector or a `vegtable` object.

Author(s)
Miguel Alvarez (<kamapu78@gmail.com>).

Examples
```r
library(vegtable)
data(Kenya_veg)

## Check the available scales
summary(Kenya_veg@coverconvert)

## Conversion by default 'top' rule
Kenya_veg <- transform(Kenya_veg, to="percent")
summary(as.factor(Kenya_veg@samples$percent))

## Conversion by 'middle' rule
Kenya_veg <- transform(Kenya_veg, to="percent", rule="middle", replace=TRUE)
summary(as.factor(Kenya_veg@samples$percent))

## Conversion by 'bottom' rule
Kenya_veg <- transform(Kenya_veg, to="percent", rule="bottom", replace=TRUE)
summary(as.factor(Kenya_veg@samples$percent))
```

---

**tv2coverconvert**

*Importing conversion tables from 'Turboveg' databases.*

**Description**
This function reads the content of cover conversion tables stored in ‘Turboveg’ and attempts to reformat them in a more comprehensive structure.

**Usage**
tv2coverconvert(file, as.is=TRUE)

**Arguments**
- **file** A connection to a DBF file containing conversion table in ‘Turboveg’.
- **as.is** A logical value passed to `read.dbf`

**Details**
This function is used by `tv2vegtable` to import the respective conversion table from ‘Turboveg’ databases. Note that conversion tables in ‘Turboveg’ have only stored the middle point for each cover class in a scale, thus it will be recommended to rebuild the ‘coverconvert’ slot or use `braun_blanquet`.
**Value**

A `coverconvert` object.

**Author(s)**

Miguel Alvarez (<kamapu78@gmail.com>).

**See Also**

tv2vegtable, read.dbf.

**Examples**

```r
library(vegtable)

## Installed 'Turboveg' version of "Fujiwara et al. (2014)"
TV_Home <- file.path(path.package("vegtable"), "tv_data", "popup", "Swea")
Table <- tv2coverconvert(file.path(TV_Home, "tvscale.dbf"))

## First scale have to be deleted from conversion table
Table@value <- Table@value[-1]
Table@conversion <- Table@conversion[-1]
summary(Table)

## Compare the 'Turboveg' version with a vegtable version
data(braun_blanquet)
summary(Table$br_bl)
summary(braun_blanquet$br_bl)
```

---

**tv2vegtable**  
*Import vegetation data from a 'Turboveg' databases.*

**Description**

`tv2vegtable` imports vegetation data sets from ‘Turboveg’ data bases.

**Usage**

```r
tv2vegtable(db, tv_home=tv.home(), skip_empty_relations=TRUE, skip_scale, clean=TRUE)
```

**Arguments**

- **db**  
  Name of ‘Turboveg’ data base as character value.
- **tv_home**  
  ‘Turboveg’ installation path as character value.
- **skip_empty_relations**  
  Logical value indicating whether empty relations may be excluded from imported database or not.
skip_scale  Character value indicating scales to be excluded in slot 'coverconvert'.
clean    Logical value indicating whether output object should be cleaned or not.

Details
Import function for ‘Turboveg’ databases into an object of class vegtable. Most of the contents of ‘Turboveg’ databases are included in DBF files and therefore imported by the function read.dbf. The automatic setting of database path will be done by the function tv.home but it can be customised by the argument 'tv_home'.
The species list will be imported by using the function tv2taxlist and therefore formatted as a taxlist object. Similarly, conversion tables will be handled as coverconvert objects.
Empty columns in the header will be deleted in the imported object.

Value
A vegtable object.

Author(s)
Miguel Alvarez (<kamapu78@gmail.com>).

See Also
tv2taxlist, tv2coverconvert, tv.home.

Examples
library(vegtable)

## Installed 'Turboveg' version of 'Fujiwara et al. (2014)'
TV_Home <- file.path(path.package("vegtable"), "tv_data")
Veg <- tv2vegtable("Fujiwara_2014", TV_Home)
summary(Veg)

---

used_synonyms Retrieve synonyms used in the data set.

Description
Plots records are rather linked to plant names than plant taxon concepts. This function provides a quick report about synonyms used in a data set (a vegtable object) and their respective accepted name.

Usage
## S4 method for signature 'vegtable'
used_synonyms(x, ...)
used_synonyms

Arguments

x A vegetable object.

... Further arguments to be passed from or to another methods.

Details

This function will only retrieve synonyms that are used in plot records.

Value

A data frame with following columns:

- **SynonymsID**  Usage ID of synonyms.
- **Synonym**   The synonym itself.
- **SynonymAuthor**  Author of synonym.
- **TaxonConceptID**  ID of the taxon concept.
- **AcceptedNameID**  Usage ID of the accepted name.
- **AcceptedName**  The respective accepted name.
- **AcceptedNameAuthor**  The author of the accepted name.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

See Also

accepted_name.

Examples

```r
library(vegetable)
data(Kenya_veg)

## Synonyms used in the Kenya_veg
Synonyms <- used_synonyms(Kenya_veg)
head(Synonyms)
```
vegtable-class

Class vegtable.

Description

Class holding vegetation-plot data sets. Designed to content all information stored in ‘Turboveg’ databases in just one object.

Details

This class was designed to include information of relevés, header data and species in just one object. Objects can be created by calls of the form `new("vegtable",...)`.

Slots

description A named character vector containing metadata.
samples A data frame with samples list.
header A data frame with plots data.
species Species list as a `taxlist` object.
layers A list including strata within samples as data frames.
relations A list including popup lists as data frames.
coverconvert A scale conversion object of class `coverconvert`.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

See Also

tv2vegetable.

Examples

```r
library(vegtable)
showClass("vegetable")
```
Mapping of plot observations.

Description

This function is a wrapper of \texttt{kml} producing and displaying KML files.

Usage

\begin{verbatim}
## S4 method for signature 'data.frame'
vegtable2kml(obj, file, coords=~ Longitude + Latitude, 
srs=CRS("+proj=longlat +datum=WGS84"))

## S4 method for signature 'vegetable'
vegtable2kml(obj, file, coords=~ LONGITUDE + LATITUDE, 
srs=CRS("+proj=longlat +datum=WGS84"))
\end{verbatim}

Arguments

- \textit{obj} Input object containing coordinate values.
- \textit{file} Character value with the name of output file (including file extension).
- \textit{coords} Either a character vector or a formula indicating the names of coordinate values.
- \textit{srs} Spatial reference system as \texttt{proj4string}.

Details

Georeferenced plots can be quickly displayed in \texttt{Google Earth} using this function.

Value

A KML file, which will be automatically opened in \texttt{Google Earth}.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

Examples

\begin{verbatim}
library(vegetable)
data(Kenya_veg)

## Plots containing Podocarpus observations
Kenya_veg@species <- subset(Kenya_veg@species, grepl("Podocarpus", TaxonName), slot="names")

Kenya_veg <- subset(Kenya_veg, TaxonUsageID %in% Kenya_veg@species@taxonNames$TaxonUsageID, slot="samples")

## Not run: vegtable2kml(Kenya_veg, "Podocarpus.kml")
\end{verbatim}
vegtable_stat  General statistics from vegtable objects.

Description
This function calculates general statistics of local ‘Turboveg’ databases as required by GIVD (Global Index of Vegetation-Plot Databases, https://www.givd.info).

Usage
vegtable_stat(vegtable)

Arguments
vegtable An object of class vegtable.

Details
This function is based on a script delivered by GIVD for summarising statistics required in the descriptions of databases (see meta data in the page of the Global Index for Vegetation-Plot Databases).

Author(s)
GIVD. Adapted by Miguel Alvarez (<kamapu78@gmail.com>).

Examples
library(vegtable)
data(Kenya_veg)

## Statistics for GIVD
vegtable_stat(Kenya_veg)

veg_relation,relation2header
Retrieve or replace relations in vegtable objects.

Description
Tables providing information about levels of categorical variables in the header of a ‘Turboveg’ database are called ‘popups’ in ‘Turboveg’, but ‘relations’ in vegtable. Such variables will be converted into factors in the slot ‘header’ according to the levels and their sorting in the respective relation.
Usage

```r
## S4 method for signature 'vegtable,character'
veg_relation(vegtable, relation, match_header=FALSE, ...)
```

```r
## Replacement method
veg_relation(vegtable, relation) <- value
```

```r
## S4 method for signature 'vegtable,data.frame'
relation2header(vegtable, relation, by, vars, ...)
```

```r
## S4 method for signature 'vegtable,character'
relation2header(vegtable, relation, ...)  
```

Arguments

- `vegtable`: An object of class `vegtable`.
- `relation`: A character value indicating the relation table to be retrieved or replaced.
- `by`: Character value indicating the name of the common column used as index for inserting values in slot 'header'.
- `vars`: A character vector with the names of variables to be inserted in slot header.
- `match_header`: A logical vector, whether only levels occurring in slot 'header' should be considered or all.
- `value`: A data frame containing the new `veg_relation`.
- `...`: Further arguments to be passed among methods.

Value

This function retrieves and object of class 'data.frame'. In the replacement method, an object of class `vegtable` including 'value' in the slot 'relations'.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

Examples

```r
library(vegtable)
data(Kenya_veg)

## overview of references
veg_relation(Kenya_veg, "REFERENCE")
```
**Wetlands-data**  
*Vegetation-plots from Tanzania.*

**Description**  
A subset of SWEA-Dataveg with plots sampled in Tanzania.

**Usage**  
```r
data(Wetlands)
```

**Format**  
An object of class `shaker` ('Wetlands') and the respective companion as `vegtable` object ('Wetlands_veg').

**Author(s)**  
Miguel Alvarez (<kamapu78@gmail.com>), Michael Curran (<currmi01@gmail.com>).

**Source**  
[http://www.givd.info/ID/AF-00-006](http://www.givd.info/ID/AF-00-006).

**References**  

**Examples**  
```r
library(vegtable)
data(Wetlands)

summary(Wetlands)
summary(Wetlands_veg)
```

---

**write_juice**  
*Exporting Tables for ‘Juice’.*

**Description**  
This function produce txt files as input formats for ‘Juice’ ([http://www.sci.muni.cz/botany/juice/](http://www.sci.muni.cz/botany/juice/)).
write_juice

Usage

## S4 method for signature 'vegtable,character,formula'
write_juice(data, file, formula, FUN, db_name="Plot Observations", header, coords, ...)

Arguments

data An object of class vegtable.

file Character value indicating the name of output files (without file extension).

db_name Name for data set displayed in import wizard.

formula A formula passed to crosstable.

FUN Function passed to crosstable.

header Variables of header to be exported.

coords Names of coordinate variables in header of 'data'.

... Further arguments passed to the function crosstable.

Details

This function produces two output files to be imported into a 'Juice' file: A vegetation table produced by crosstable and a header table. Both tables share the file name plus a suffix (table for the vegetation table and header for the header).

For the import in 'Juice', you may start with the table following in the menu 'File -> Import -> Table -> from Spreadsheet File (e.g. EXCEL Table)' and then follow the wizard. You further import the header table following in the menu 'File -> Import -> Header Data -> From Comma Delimited File'. Notice that the vegetation is a semi-colon delimited file, while the header is a comma delimited file.

Author(s)

Miguel Alvarez (<kamapu78@gmail.com>).

Examples

library(vegtable)
data(Kenya_veg)

## Only first 20 observations
Kenya_veg <- Kenya_veg[1:20,]

## Not run:
write_juice(Kenya_veg, "SWEA", FUN=mean)

## End(Not run)
Index

+Topic **classes**
  coverconvert-class, 9

+Topic **datasets**
  aspect_conv-data, 6
  braun_blanquet-data, 6
  dune_veg-data, 13
  Kenya_veg-data, 15
  Wetlands-data, 38

+Topic **methods**
  add_releves, 2
  aggregate, 4
  as.list, 5
  clean, 7
  count_taxa, 8
  crosstable, cross2db, 10
  df2vegtable, 11
  Extract, 13
  header, 14
  make_cocktail, set_pseudo, set_group, set_formulas, 17
  match_names, 19
  merge_taxa, taxa2samples, 20
  names, dimnames, 21
  subset, 25
  summary, 26
  transform, 29
  transform, 29
  used_synonyms, 32
  veg_relation, relation2header, 36
  vegtable2kml, 35
  write_juice, 38
  [, vegtable, ANY, ANY, ANY-method (Extract), 13
  [, vegtable, ANY, ANY-method (Extract), 13
  [, vegtable-method (Extract), 13
  [<- (Extract), 13
  [<-, vegtable, ANY, ANY, ANY-method (Extract), 13
  $, coverconvert-method (Extract), 13
  $, vegtable-method (Extract), 13
  $<-(Extract), 13
  $<-, coverconvert, list-method (Extract), 13
  $<-, coverconvert, list-method (Extract), 13
  $<-, vegtable, ANY-method (Extract), 13
  $<-, vegtable-method (Extract), 13
  accepted_name, 33
  add_releves, 2
  add_releves, vegtable, data.frame-method (add_releves), 2
  aggregate, 4, 4, 10, 16
  aggregate, formula-method (aggregate), 4
  as.list, 5
  as.list, coverconvert-method (as.list), 5
  as.list, vegtable-method (as.list), 5
  aspect_conv (aspect_conv-data), 6
  aspect_conv-data, 6
  braun_blanquet, 10, 30
  braun_blanquet (braun_blanquet-data), 6
  braun_blanquet-data, 6
  clean, 7
  clean, vegtable-method (clean), 7
  count_taxa, 8
  count_taxa, formula-method (count_taxa), 8
  count_taxa, vegtable-method (count_taxa), 8
  coverconvert, 5–7, 13, 22, 26, 29, 31, 32, 34
  coverconvert (coverconvert-class), 9
  coverconvert-class, 9
  cross2db, 3
  cross2db (crosstable, cross2db), 10
  crosstable, 4, 16, 39
  crosstable (crosstable, cross2db), 10
  crosstable, cross2db, 10
  crosstable, formula, data.frame-method (crosstable, cross2db), 10
INDEX

41

crosstable, formula, vegtable-method
   (crosstable, cross2db), 10

data.frame, 4, 11

df2vegtable, 11

df2vegtable.data.frame, numeric, missing-method
   (df2vegtable), 11

dimnames (names, dimnames), 21

dimnames, vegtable-method
   (names, dimnames), 21

dune, 13

dune_veg (dune_veg-data), 13

dune_veg-data, 13

Extract, 13, 14

format, 26

header, 14

header<-(header), 14

header<-, vegtable.data.frame-method
   (header), 14

Kenya_veg (Kenya_veg-data), 15

Kenya_veg-data, 15

kml, 35

layers2samples, 16

layers2samples, vegtable, character, character-method
   (layers2samples), 16

layers2samples, vegtable, character, missing-method
   (layers2samples), 16

make_cocktail, 24

make_cocktail
   (make_cocktail, set_pseudo, set_group, set_formula), 17

relation2header
   (veg_relation, relation2header), 36

set_formula, 24

set_formula
   (make_cocktail, set_pseudo, set_group, set_formula), 17

set_formula, shaker, taxlist, character-method

match_names, 19, 20

match_names, character, vegtable-method
   (match_names), 19

merge_taxa, 20

merge_taxa (merge_taxa, taxa2samples), 20

merge_taxa, vegtable, character-method
   (merge_taxa, taxa2samples), 20

merge_taxa, vegtable, missing, character-method
   (merge_taxa, taxa2samples), 20

merge_taxa, vegtable, numeric, missing-method
   (merge_taxa, taxa2samples), 20

names (names, dimnames), 21

names, coverconvert-method
   (names, dimnames), 21

names, vegtable, 21

names<-(names, dimnames), 21

names<-, coverconvert-method
   (names, dimnames), 21

names<-, vegtable-method
   (names, dimnames), 21

read.dbf, 30–32

read_juice, 22

relation2header
   (veg_relation, relation2header), 36

relation2header, vegtable, character-method
   (veg_relation, relation2header), 36

relation2header, vegtable, data.frame-method
   (veg_relation, relation2header), 36

set_formula, 24

set_formula
   (make_cocktail, set_pseudo, set_group, set_formula), 17

set_formula, shaker, taxlist, character-method

match_names, 19, 20

match_names, character, vegtable-method
   (match_names), 19

merge_taxa, 20

merge_taxa (merge_taxa, taxa2samples), 20

merge_taxa, vegtable, character-method
   (merge_taxa, taxa2samples), 20

merge_taxa, vegtable, missing, character-method
   (merge_taxa, taxa2samples), 20

merge_taxa, vegtable, numeric, missing-method
   (merge_taxa, taxa2samples), 20

names (names, dimnames), 21

names, coverconvert-method
   (names, dimnames), 21

names, vegtable, 21

names<-(names, dimnames), 21

names<-, coverconvert-method
   (names, dimnames), 21

names<-, vegtable-method
   (names, dimnames), 21

read.dbf, 30–32

read_juice, 22

readLines, 23

relation2header
   (veg_relation, relation2header), 36

relation2header, vegtable, character-method
   (veg_relation, relation2header), 36

relation2header, vegtable, data.frame-method
   (veg_relation, relation2header), 36
set_group, shaker, vegetable, character-method
(make_cocktail, set_pseudo, set_group, set_formula)

set_pseudo, 24
(set_pseudo, 24)

transform, 7, 9, 29
(transform, 29)

set_pseudo, shaker, taxlist, character-method
(make_cocktail, set_pseudo, set_group, set_formula)

shaker, 17, 18, 26, 38
(shaker-class, 24)

stringsim, 19, 20

subset, 16, 25, 26
(subset, vegtable-method (subset), 25
summary, 26
(summary, coverconvert-method (summary), 26
(summary, shaker-method (summary), 26
(summary, vegtable-method (summary), 26

taxa2samples (merge_taxa, taxa2samples), 20
taxa2samples, vegtable-method
(merge_taxa, taxa2samples), 20
taxlist, 8, 17-21, 26, 32, 34
trait_proportion (trait_stats, trait_proportion), 27
(trait_proportion(), 27
(trait_proportion, character, vegtable-method
(trait_stats, trait_proportion), 27
(trait_proportion, formula, vegtable-method
(trait_stats, trait_proportion), 27
(trait_stats (trait_stats, trait_proportion), 27
(trait_stats(), 27
(trait_stats, trait_proportion, 27
(trait_stats, character, vegtable-method
(trait_stats, trait_proportion), 27
(trait_stats, formula, vegtable-method
(trait_stats, trait_proportion), 27

veg_relation
(veg_relation, relation2header), 36
(veg_relation, relation2header, 36
(veg_relation, vegtable, character-method
(veg_relation, relation2header), 36
(veg_relation<-
(veg_relation, relation2header), 36
(veg_relation<-, vegtable, character, data.frame-method
(veg_relation, relation2header), 36

vegan, 10, 13
(vegtable, 2-5, 7-18, 20-22, 24-26, 28-30,
32, 33, 36-39
(vegtable (vegtable-class), 34
(vegtable-class, 34
(vegtable2kml, 35
(vegtable2kml, data.frame-method
(vegtable2kml), 35
(vegtable2kml, vegtable-method
(vegtable2kml), 35

vegtable_stat, 36

Wetlands, 18
(Wetlands-data, 38
(Wetlands-data, 38
Wetlands_veg (Wetlands-data), 38
Wetlands_veg-data (Wetlands-data), 38
write_juice, 38
write_juice, vegtable, character, formula-method
(write_juice). 38